

ТРУДЫ МОЛОДЫХ УЧЕНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ
НАУК ВГУ

Выпуск 1

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет компьютерных наук

**ТРУДЫ МОЛОДЫХ УЧЁНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ
НАУК ВГУ**

В ы п у с к 1

Под редакцией кандидата технических наук Д. Н. Борисова

Воронеж
ООО «ВЭЛБОРН»
2021

УДК 004.65 + 004.438.5
ББК Ч481(2)22
Т78

*Рекомендовано к опубликованию
Ученым советом факультета компьютерных наук ВГУ*

Т78 **Труды молодых учёных факультета компьютерных наук ВГУ.** Выпуск 1 / под ред. Д. Н. Борисова ; Воронежский государственный университет. – Воронеж : ООО «ВЭЛБОРН», 2021. – 491 с.

ISBN 978-5-6045486-3-9

В сборник включены научные работы студентов бакалавриата и магистратуры факультета компьютерных наук ВГУ, выполненные в 2020–2021 гг. под руководством преподавателей факультета, представленные в виде докладов и рекомендованные к опубликованию оргкомитетами студенческих научных конференций.

УДК 004.65 + 004.438.5
ББК Ч481(2)22

ISBN 978-5-6045486-3-9

© Воронежский государственный университет, 2021
© ООО «ВЭЛБОРН», 2021

Труды студентов бакалавриата

Обзор алгоритма monocular SLAM для определения позиционирования объекта в пространстве

В.М. Беспалов

Студент бакалавр

Д. И. Соломатин

Старший преподаватель

Введение

В современном мире широко применяются разнообразные мобильные робототехнические системы для выполнения различных задач. Примерами таких систем являются автономные роботы, такие как роботы пылесосы, роботы доставщики, планетоходы и так далее. Эти машины объединяет то, что для корректной работы им необходимо ориентироваться в пространстве, в котором они находятся.

На текущий момент, задача ориентации в пространстве является одной из ключевых в мобильной робототехнике. Для работы в неизвестном пространстве робот должен достаточно точно определять свое положение относительно тех объектов, которые его окружают, строить свой маршрут, а также определять траекторию своего движения.

Ввиду повсеместного распространения камер в робототехнике, одним из путей решения данной задачи является визуальная локализация и построение карты (англ. visual SLAM).

Данная статья посвящена методу позиционирования объекта с использованием лишь одной камеры (monocular SLAM [1]), который может применяться на небольших роботах, где использование двух камер не имеет смысла ввиду малых габаритов или низкой общей цены.

1. Общая идея алгоритма

Алгоритм получает на вход видеопоток, на первом кадре которого обнаруживаются ключевые точки, а каждом последующем – выполняется последовательность действий: для каждой ключевой точки предыдущего кадра ищется положение на текущем, после чего составляется список пар точек, по которому, при помощи алгоритма RANSAC [2] находится существенная матрица, при помощи которой вычисляется матрица поворота и вектор смещения текущего кадра относительно предыдущего. Именно по данным этих двух матриц

определяется текущее положение камеры относительно начальной точки.

2. Выбор ключевых точек

На текущий момент существует множество алгоритмов нахождения ключевых точек: детектор Харриса, ORB, SIFT, FAST, Shi-Tomasi и прочие. В данной реализации используется один из самых быстрых алгоритмов в своём классе – FAST.

В процессе выбора алгоритма было проведено сравнение времени работы разных алгоритмов, при условии, что на каждом кадре было обнаружено 300-400 ключевых точек. Время работы данных алгоритмов для одной и той же последовательности кадров, приведено в таблице ниже.

Таблица

Сравнение времени выполнения для разных алгоритмов

Алгоритм	Среднее время выполнения для изображения 1368x768, мс
Fast	2
ORB	6
Детектор Харриса	10
SIFT	110
Shi-Tomasi	12

Fast (англ. Features From Accelerated Segment Test) [3] был разработан Ростоном и Драммондом в 2005 году. В алгоритме рассматривается окружность из 16 пикселей, окружающая точку-кандидата P . Точка считается ключевой, если для нее существуют N смежных пикселей на окружности, интенсивность которых больше, чем $I_p + t$ или, наоборот меньше, чем $I_p - t$, где I_p – интенсивность точки P , а t – пороговая величина. На рис. 1 представлен пример работы данного алгоритма для одного из пикселей изображения.

В данном случае используется $N = 9$ и $t = 16$. Данные значения взяты для того, чтобы получать примерно по 300 – 400 на каждый кадр изображения 1368x768.

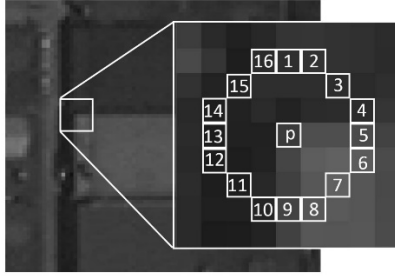


Рис. 1. Окрестность пикселя Р при использовании FAST детектора

3. Поиск пар точек на двух изображениях

Для нахождения расположения точки предыдущего кадра на текущем используется реализация алгоритма Лукаса-Канаде [4] для вычисления оптического потока из библиотеки OpenCV. Он получает на вход оба кадра, а также набор точек текущего кадра, возвращает набор векторов оптического потока, которые соответствуют каждой ключевой точке переданного на вход набора. При суммировании координат точки и соответствующего ей вектора вычисляется набор точек на текущем кадре. Данный набор сравнивается с набором, полученном при помощи алгоритма FAST и возвращается пересечение данных наборов. Таким образом создаются пары точек, которые образуются при переходе от предыдущего кадра к текущему.

4. Задача поиска перемещения и поворота камеры

При помощи полученных пар точек необходимо найти перемещение камеры в мире.

При использовании модели, представленной на рис. 2, известно расположение камеры O_1 , точек p , p' , откуда необходимо найти $\overline{O_1O_2}$.

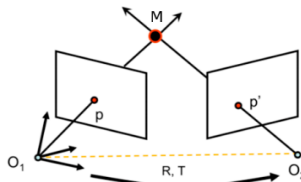


Рис. 2. Упрощенное представление задачи поиска перемещения и поворота камеры

При решении данной задачи подразумевается, что линии O_1p и O_2p' пересекаются в точке M . По трём точкам: O_1, O_2 и M возможно построить плоскость, которая называется эпиполярной плоскостью. Отсюда вытекает уравнение:

$$p^T \cdot [T_x]Rp' = p^T Ep' = 0 \quad (1)$$

где E – матрица 3×3 , которая называется существенной.

Учитывая, что $p = [x, y, 1]^T$, $p' = [x', y', 1]^T$, так как они находятся на плоскости кадра, уравнение (1) сводится к системе (2)

$$\begin{pmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1x'_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ x_2x'_2 & x_2y'_2 & x_2 & y_2x'_2 & y_2y'_2 & y_2 & x'_2 & y'_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_8x'_8 & x_8y'_8 & x_8 & y_8x'_8 & y_8y'_8 & y_8 & x'_8 & y'_8 & 1 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \end{pmatrix} = 0 \quad (2)$$

Далее, при решении системы, вычисляется существенная матрица E , составленную из коэффициентов e_i .

Данное решение будет верным только в том случае, если количество пар точек будет равно 8. Для поиска существенной матрицы для большего числа пар точек использовался алгоритм RANSAC, который получает на вход множество пар точек на каждом изображении и максимальную погрешность.

При помощи данного итеративного алгоритма вычисляется существенная матрица, которая подходит для наибольшего числа пар точек. На каждом шаге RANSAC случайно выбираются 8 точек, относительно которых решается уравнение 2, далее полученная матрица и каждая пара точек из набора подставляются в уравнение 1. Если полученное значение по модулю больше погрешности, то пара точек считается выбросом. Алгоритм возвращает такую матрицу, которая имеет наименьшее число таких выбросов.

Используя сингулярное разложение полученной матрицы E , а также при помощи библиотеки NumPy, вычисляются матрицы U, V^T .

Тогда возможно достаточно легко определить матрицу поворота и вектор переноса [5].

$$R = UR_Z^T \left(\pm \frac{\pi}{2} \right) V^T, t = UR_Z \left(\pm \frac{\pi}{2} \right) \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} U^T \quad (3)$$

где R_Z – матрица поворота на определенный угол вокруг оси z , R, t – поворот и перенос камеры при переходе из точки O_1 в точку O_2 .

В итоге получено 4 разных решения для поворота и переноса, из них выбираем то, при котором все точки будут расположены перед камерой (оно будет только одно).

5. Проверка результатов работы

При проверке результатов работы алгоритма были использованы записи автомобильного видеорежистратора, для построения пройденного пути и сервис Yandex Maps для сравнения с реальной местностью. При проверке пройденного пути с реальной картой в городской или сельской местности была получена погрешность 2-3 м при на 200-300 м пути (при этом количество точек на кадр составляет 350-400 в среднем). Небольшой отрезок пути, наложенный на реальную карту сельской местности показан на рис. 3.

При этом, в степной местности погрешность может сильно меняться в зависимости от наличия каких-либо выделяющихся объектов. На местности такого типа количество точек падает до 100-150, что в среднем негативно сказывается на точности работы.

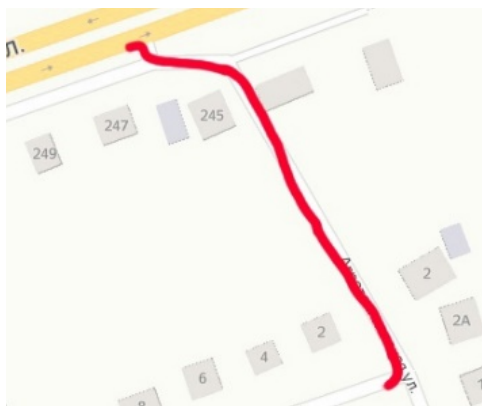


Рис. 3. Пример работы алгоритма

Также, при работе алгоритма происходит визуализация положения точек и их расположения на экране, представленная на рис. 4. Отображается расположение точек на каждом кадре, а также их перемещение относительно предыдущего.



Рис. 4. Визуализация расположения ключевых точек

На данном рисунке красным цветом обозначены сами ключевые точки текущего кадра, а желтым – линии, соединяющие положение точки на предыдущем кадре с расположением на текущем.

Заключение

Таким образом, получилось создать алгоритм, позволяющий строить карту местности без использования каких-либо средств, помимо камеры. По сравнению с PTAM, данная реализация отличается повышенной скоростью работы, за счёт использования более быстрых алгоритмов выбора ключевых точек. К сожалению, такой подход приводит к повышению ошибки позиционирования.

В будущем планируется продолжить работу над текущим алгоритмом с целью повышения точности и быстродействия.

Список литературы

1. Hyon Lim Real-Time 6-DOF Monocular Visual SLAM in a Large-Scale Environment / Hyon Lim, Jongwoo Lim. and H. Jin Kim // 2014 IEEE International Conference on Robotics and Automation (ICRA) (Hong Kong, China, 31 May – 7 June 2014) – Hong Kong, 2014 – P. 1532-1539.

2. T. Botterill Refining essential matrix estimates from RANSAC / T. Botterill, S. Mills, R. Green // 2011 International Conference on Digital Image Computing: Techniques and Applications (DICTA) (Noosa, Queensland Australia, 6-8 December 2011) – IEEE, 2011 – P. 561-566
3. Rosten, E. Machine Learning for High-speed Corner Detection / E. Rosten, T. Drummond // Computer Vision – ECCV 2006 (Graz, Austria, May 7-13, 2006) – Graz, 2006 – P. 430-443.
4. Lucas, B. An Iterative Image Registration Technique with an Application to Stereo Vision / B. Lucas, T. Kanade // Proceedings of the Seventh International Joint Conference on Artificial Intelligence (Vancouver, Canada, 24-28 August 1981) – Vancouver, 1981 – P. 674-679.
5. An Invitation to 3-D Vision: From Images to Models / Yi Ma [et al.] – New York: Springer, 2004. – 528 p.

Использование нейронных сетей для определения местоположения радужной оболочки глаза на изображениях

Е. П. Булавина

Студент бакалавр

Д. И. Соломатин

Старший преподаватель

Введение

В настоящее время большому количеству приложений требуется определение местоположения глаз, а в частности, радужной оболочки. Применение этому находится в таких направлениях, как вычислительная фотография (например, изменение таких параметров радужки как цвет, размер) или же дополненная реальность (создание виртуальных аватаров).

Одним из способов решения данной задачи является использование нейронных сетей, что позволяет создать быстродействующую и достаточно точную модель, обладающую способностью находить закономерности в таком типе данных как изображения.

В данной работе рассматривается задача определения местоположения радужной оболочки глаза с использованием сверточных нейронных сетей.

1. Общая идея

Так как необходимо местоположение только радужки, нет смысла использовать в нейронной сети все изображение. Поэтому, стоит предварительно обработать изображение. Для начала, на изображении находятся лица, откуда можно вычислить координаты, ограничивающие участок с глазом. А уже обрезанное по координатам изображение передается нейронной сети, которая в виде результата выдает на выходе предугаданные координаты ключевых точек радужной оболочки. Зная координаты вырезанного региона, можно перенести предугаданные точки на исходное изображение.

Так как нейронная сеть будет работать только с обрезанным изображением, это позволит оптимизировать её работу, так как сети не придется еще заниматься задачей обнаружения глаза на фотографии.

2. Подготовка данных

Для того чтобы обучить нейронную сеть, предугадывающую местоположения ключевых точек радужки глаза, необходимо было подготовить набор данных (датасет), который включает в себя изображения глаз и координаты размеченных точек.

Изображения для набора данных собирались из публичных источников, таких как Unsplash, Pexels или же датасет Flickr-Faces [1]. Изначально они представляли собой фотографии людей, из которых в последствии с помощью каскада Хаара [2] были извлечены участки с глазами человека. Таким образом было собрано более 1200 цветных изображений.

Так же, поскольку качество обучения нейронной сети напрямую зависит от количества исходных данных при тренировке, была произведена аугментация данных. Это позволило искусственно увеличить объем датасета в 6 раз за счет таких операций как отражение и изменение гаммы.

Так как обучение нейронной сети производится методом обучения с учителем, необходимо было так же подготовить разметку данных. Предложенная модель состоит из 12 точек, составляющих эллипс. Такое количество точек делает разметку более пригодной для возможной последующей задачи фиттинга контура радужки. Меньшее количество точек может послужить поводом неправильного вписывания эллипса.

Для ручной разметки изображений был дополнительно создан модуль, упрощающий данную задачу. Он позволяет по отмеченным на изображении точкам генерировать xml файл, который используется при передаче данных в загрузчик.

3. Реализация

Таким образом, для подготовки и обучения нейронной сети было создано 3 отдельных модуля (рис. 1).

Модуль первичной обработки отвечает за выделение участков глаз на изображении. На вход поступает фотография, на которой с помощью каскада Хаара [2] находятся необходимые фрагменты. Квадратные участки обрезаются и преобразуются до размера в 224 пикселя.

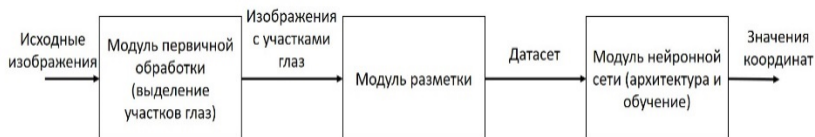


Рис. 1. Схема последовательности работы модулей

Далее, полученные изображения передаются в модуль разметки, где перед обучением сети формируется датасет из набора фотографий и координат точек. Датасет сохраняется в xml-формате и передается в модуль нейронной сети.

Перед передачей данных для обучения в сеть, значения цветов изображения, как и координаты точек нормализуются. Результатом работы модуля являются нормированные значения.

Данные модули были написаны на языке программирования Python, с использованием фреймворка PyTorch и библиотек NumPy, OpenCV и Pillow для вычислений и работы с изображением.

4. Архитектура сети и вспомогательные функции

В качестве архитектуры рассматривались такие, как ResNet, VGG AlexNet, Inception и другие. В связи с используемым аппаратным обеспечением накладывается ограничение в виде легковесности сети. Поэтому была выбрана AlexNet (рис. 2), как показавшая один из лучших результатов среди оставшихся вариантов.

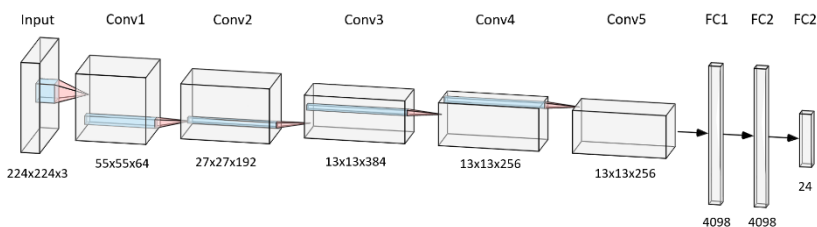


Рис. 2. Архитектура сети AlexNet

AlexNet – это сверточная нейронная сеть, состоящая из 8 слоев. Пять из них являются сверточными и 3 последних – полносвязными. Как функция активации используется ReLU (линейный выпрямитель), которая применяется после каждого сверточного и полносвязного слоя. А перед первым и вторым слоем применяется дропаут, который позволяет частично решить проблему переобучения [3].

На вход нейронной сети поступают нормализованные изображения с размерностью 224x224x3. После прохождения через сеть на выходе результатом являются 24 значения, соответствующие нормированным значениям координат точек, которые необходимо было найти.

Перед обучением нейронной сети необходимо было выбрать функции оптимизации и потери. Функция оптимизации отвечает за достижение минимального результата функции потерь, когда сама

функция потерь показывает, насколько хорошо или же плохо текущие веса нейронной сети справляются с задачей.

В качестве функции оптимизации был выбран алгоритм Adam с параметрами $\beta_1 = 0.9$, $\beta_2 = 0.999$ и $\alpha = 0.0001$. Он объединяет в себе преимущества других алгоритмов, таких как AdaGrad или же RMSProp. Adam высчитывает 2 момента. Первый отвечает за скорость изменения весов, а второй используется для оптимизации [4].

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2. \quad (1)$$

В качестве функции ошибки была выбрана MSELoss (среднеквадратичная ошибка), которая вычисляется как среднее из квадратов разностей предсказанных и истинных значений [5]. Из-за этого, она всегда принимает неотрицательное значение.

5. Обучение

Обучение нейронной сети происходило методом обучения с учителем (рис. 3). Модель, зная значение на выходе, должна найти зависимость, которая позволит получить такой результат. Для проверки правильности нахождения зависимости исходный датасет был поделен на тренировочный и проверочный. После чего было произведено обучение нейронной сети в течение 50 эпох. Обучение для данного датасета заняло около 7 часов на одной видеокарте GTX 1050 TI.

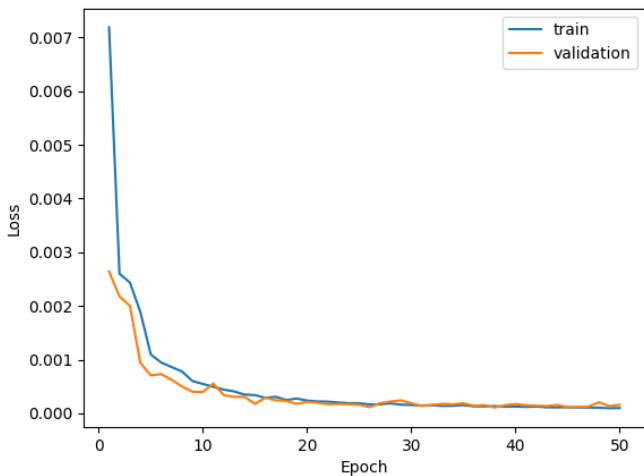
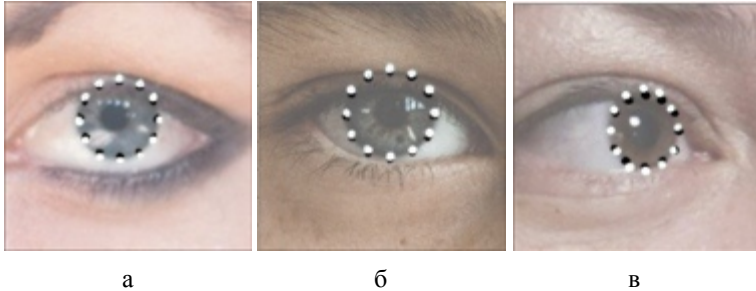


Рис. 3. График обучения нейронной сети

6. Результаты

Обученная нейронная сеть достаточно точно определяет местоположение радужной оболочки глаза. Но наблюдаются исключения в виде случаев, когда лицо человека направлено вбок. Например, на рис. 4в, где ошибка возрастает, по сравнению с другими примерами. Это объясняется недостаточным количеством подобных примеров в исходном датасете, так как люди на фотографиях стараются смотреть в направлении камеры.



a – 0.0000206, б – 0.0000546, в – 0.000157

белым цветом отмечены размеченные точки, черным – предугаданные

Рис. 4. Результат работы нейронной сети вместе со значениями функции ошибки

Заключение

В данной работе было рассмотрено использование сверточных нейронных сетей для определения местоположения радужной оболочки глаз. Для этого были реализованы модули для создания и разметки датасета, включая обучение сети по нему. Дополнительно был собран датасет, предназначенный для данной архитектуры нейронной сети. Данные модули могут использоваться как по отдельности, так и как часть приложения, ответственная за нахождение радужки.

Список литературы

1. Karras, T. A Style-Based Generator Architecture for Generative Adversarial Networks / T. Karras, S. Laine, T. Aila // 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (Long Beach, CA, USA, 15-20 June 2019). – Long Beach, 2019. – P. 4396-4405.
2. OpenCV: Cascade Classifier [Электронный ресурс]. – Режим доступа: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html
3. Krizhevsky, A. ImageNet classification with deep convolutional neural networks / A. Krizhevsky, I. Sutskever, G E. Hinton // Proceedings of

the 25th International Conference on Neural Information Processing Systems (NIPS) (Lake Tahoe, Nevada, USA, 3-8 December 2012). – Lake Tahoe, 2012. – Vol. 2. – P. 1097-1105.

4. Kingma, D. P. Adam: A Method for Stochastic Optimization / D. P. Kingma, J. Ba // Proceedings of the 3rd International Conference on Learning Representations (ICLR) (San Diego, CA, USA, 7-9 May 2015). – San Diego, 2015. – 15p.

5. MSELoss – PyTorch 1.8.1 documentation [Электронный ресурс]. – Режим доступа : <https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html>

Разработка программного модуля бюджетирования предприятия для базовой конфигурации «1С: Бухгалтерия 8.3»

И. А. Герасимов

Студент бакалавр

Н. В. Абрамова

Ассистент

Введение

Разработка дополнительных модулей и надстроек к различным конфигурациям платформы «1С Предприятие» является популярной задачей и даже целым направлением в сфере информационных технологий [1]. Актуальность и востребованность программистов, способных грамотно интегрировать дополнительные элементы в имеющуюся в наличии конфигурацию, объясняется тем, что многим предприятиям очень часто недостаточно предоставляемого фирмой «1С» функционала, включенного в «коробочную» версию поставляемой программы. Вследствие этого компании прибегают к услугам разработчиков для реализации дополнительного функционала. Конфигурация «1С:Бухгалтерия 8.3» - очень популярное решение на рынке для ведения бухгалтерии и сдачи отчетности. Эта конфигурация легкая, маленькая, недорогая, хорошо отлажена разработчиком.

Но на современном предприятии уже недостаточно ведения только бухгалтерского учета. Одним из наиболее востребованных средств ведения бизнеса предприятия является бюджетирование. Бюджетирование – это процесс планирования и распределения ресурсов предприятия, результаты которого оформляются системой бюджетов [2]. Как правило, тема бюджетирования решается с помощью различных электронных таблиц. Но возникает закономерный вопрос - конфигурация «1С:Бухгалтерия 8.3» фиксирует данные о всех хозяйственных операциях, происходящих на предприятии, таким образом содержит все данные, которые требуются для анализа любых бюджетов организации. Эти данные являются фактическими для бюджетирования (т.е. являются Фактом для бюджетирования). Почему бы не воспользоваться этим и не создать дополнительный блок

бюджетирования - требуется только взять данные «1С:Бухгалтерия 8.3» и представить в нужной форме?

В ходе представленной работы был реализован «Бюджет доходов и расходов» по заданным статьям. Так как планирование не является предметом учета конфигурации «1С:Бухгалтерия 8.3», то реализован новый функционал - ввод, хранение плановых данных. Факт же собирается из имеющихся данных в базе конфигурации «1С:Бухгалтерия 8.3».

1. Постановка задачи

Бюджет доходов и расходов представляет собой план доходов и расходов предприятия на определенный промежуток времени – бюджетный период, в котором отражается финансовый результат работы компании. В рамках данной работы были реализованы инструменты ввода плановых показателей доходов и расходов предприятия, а также сбора данных из регистров «1С:Бухгалтерия 8.3» для сравнения и анализа запланированных и фактических расходов по выделенным статьям бюджетирования предприятия.

В результате, в конфигурации «1С:Бухгалтерия 8.3» был реализован упрощенный вариант модуля бюджетирования предприятия, содержащий в себе необходимый функционал. Были реализованы следующие дополнительные объекты конфигурации: подсистема Бюджетирование, документ ввода плановых показателей БДРПлан, документ сбора фактических показателей БДРФакт, регистр накопления БДРДвиженияПоСтатьям и отчет БДРСравнениеПланФакт [3]. Диаграмма последовательности использования внедренного модуля представлена на рис. 1.

2. Документ ввода плановых показателей

Документ БДРПлан служит для ввода пользователем плановых показателей затрат по статьям расходов и прочим статьям доходов и расходов на годовой период ежемесячно. Для реализации этого функционала в документе были созданы табличная часть и форма документа. Элементами табличной части являются Счет, отображающий код соответствующей статьи, Статья (бюджетирования), Сумма, а также двенадцать элементов числового типа по месяцам.

В обработке события ПриСозданииНаСервере формы документа происходит вызов процедуры ЗаполнитьТабличнуюЧасть, в результате работы которой при создании документа в его табличную часть автоматически добавляются новые строки с заполненными значениями полей Счет и Статья, данные для заполнения которых берутся из справочников Статьи затрат и Прочие доходы и расходы.

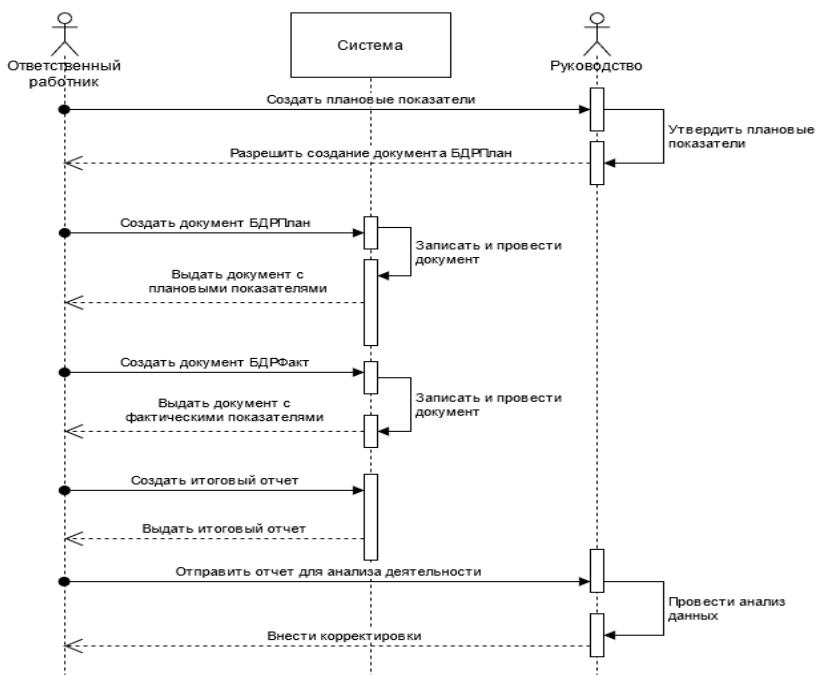


Рис. 1. Диаграмма последовательности

Таким образом, при создании нового документа в нем автоматически появятся строки, содержащие в себе все статьи бюджетирования, отсортированные по их кодам. Пользователю требуется лишь заполнить значения планируемых показателей затрат в ячейках на пересечении статей бюджетирования и соответствующих месяцев. Результат выполнения процедуры ЗаполнитьТабличнуюЧасть представлен на рис. 2.

При внесении, удалении или изменении данных в ячейках полей, относящихся к месяцам, срабатывает соответствующая обработка события ПриИзменении, которая вызывает процедуру РассчитатьСумму для строки, в которой произошли изменения. Благодаря этому, при каждом изменении данных в ячейках таблицы происходит автоматический перерасчет суммы по строке. При проведении документа его строки записываются в регистр накопления БДРДвиженияПоСтатьям с видом движения Приход для более удобной и быстрой выборки данных для итогового отчета.

N	Счет	Статья
1	00-000001	Оплата труда
2	00-000002	Оплата труда (ЕНВД)
3	00-000003	Страховые взносы
4	00-000004	Страховые взносы (ЕНВД)
5	00-000005	Взносы в ФСС от НС и ПЗ
6	00-000006	Взносы в ФСС от НС и ПЗ (ЕН
7	00-000007	Имущественные налоги
8	00-000008	Торговый сбор
9	00-000009	Услуги комиссионеров
10	00-000010	Амортизация
11	00-000011	Списание материалов
12	00-000012	Списание НДС
13	00-000013	Списание НДС (ЕНВД)
14	00-000014	Амортизационная премия
15	00-000015	Прочие затраты
16	2.01	Материалы/Драг. мет.

Рис. 2. Процедура ЗаполнитьТабличнуюЧасть

3. Документ сбора фактических показателей

Перед формированием итогового отчета пользователю нужно создать и провести документ БДРФакт для попадания в отчет фактических показателей затрат по соответствующим статьям за период. Параметром в этом документе является ссылка на проведенный документ БДРПлан. По своей структуре этот документ полностью повторяет документ БДРПлан. Табличные части документов одинаковы, исключением является то, что все поля в документе сбора фактических показателей недоступны пользователю для изменения.

На форме БДРФакт содержатся все те же процедуры и обработчики событий, что и на форме документа БДРПлан. В отличие от БДРПлан, была добавлена процедура ЗаполнитьФакт и обработчик события изменения параметра ДокументПлана, результатом работы которого является вызов ЗаполнитьФакт. В процедуре ЗаполнитьФакт происходит запрос к регистру бухгалтерии ХозрасчетныйОстаткиИОбороты, в выборку попадают Субkonto1 проводки и ее сумма СуммаОборот за год, взятый из параметра Дата выбранного документа ввода плановых показателей. При совпадении значений Субkonto1 и Статьи в строке заполняются ячейки строки с отбором по дате, после чего пересчитывается сумма строки. Таким образом, в ячейки табличной части записываются суммы оборотов статей за соответствующий месяц.

При проведении документа, как и в случае с БДРПлан, его строки записываются в регистр накопления БДРДвиженияПоСтатьям, но с видом движения Расход. Благодаря этому корректно подсчитывается

сумма всех движений по конкретной статье из регистра БДРДвиженияПоСтатьям.

4. Итоговый отчет

Отчет БДРСравнениеПланФакт состоит из Схемы компоновки данных, содержащей в себе один набор данных, в котором происходит запрос к регистру накопления БДРДвиженияПоСтатьямОбороты. В выборку попадают измерения регистра СтатьяОборотов и Код, реквизит Период, ресурс СуммаОборот, а также все доступные ресурсы по каждому месяцу, содержащие в себе суммы по движениям с типами Приход и Расход. В итоговый отчет попадают только те движения регистра, Период которых находится между датами введенных параметров НачалоПериода и КонецПериода.

На вкладке Настройки отчета в конфигураторе были добавлены две группировки – иерархическая группировка СтатьяОборотов.Родитель, благодаря которой итоговые записи оказываются сгруппированы по группам статей в соответствии с записями справочников и могут быть свернуты. В созданную группировку включаются Детальные записи – группировка без иерархии и поля группировки, содержащие в себе все записи, попадающие в отчет. Структура отчета на вкладке Настройки отображена на рис. 3.

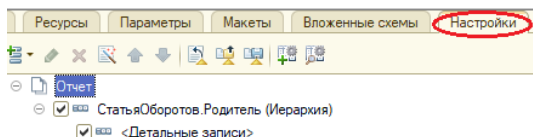


Рис. 3. Вкладка Настройки Схемы компоновки данных

На вкладке Сортировка группировки СтатьяОборотов.Родитель выставлено условие сортировки данных СтатьяОборотов.Родитель.Код, при использовании которого выводимые статьи и их группы оказываются распределены последовательно по их кодам. На вкладке Выбранные поля отчета для вывода были включены поля Код, Статья, Обороты приходов и расходов для каждого месяца, а также СуммаОборот. Для удобства использования отчета и упрощения восприятия его итогов в Выбранных полях были созданы группы с названиями каждого из двенадцати месяцев, включающие в себя соответствующие суммы оборотов.

Результат формирования отчета БДРСравнениеПланФакт представлен на рис. 4.

Январь		Февраль		Март		Апрель	
Эталон	Факт	Эталон	Факт	Эталон	Факт	Эталон	Факт
597,00		127,00		261,00		11,00	
542,00		54,00		5,00			
				85,00			
54,00		7,00		67,00			
		6,00					
				85,00			
						6,00	
		54,00		7,00		5,00	
1,00				7,00			
		6,00		5,00			
68,00		6,00		45,00		71,00	
68,00		6,00		45,00		6,00	
						65,00	
2,00							
1,00							
1,00							
			5 000,00				
			5 000,00				
667,00		133,00	5 000,00	306,00		82,00	

Рис. 4. Отчет БДРСравнениеПланФакт

Заключение

Данная статья посвящена разработке и реализации программного модуля бюджетирования для конфигурации «1С:Бухгалтерия 8.3» платформы «1С:Предприятие 8.3». Описаны логика, структура и создание самого разрабатываемого программного решения. В итоге данной разработки получился полностью рабочий программный модуль, реализованный в виде дополнительных элементов конфигурации. Реализованный модуль является расширяемым, его функционал может и дальше дорабатываться и изменяться при необходимости. Использование данного решения позволяет вести бюджет доходов и расходов на предприятии с возможностью записать, просмотреть и изменить плановые показатели, а также сравнить показатели плановых и фактических затрат предприятия при помощи отчета. Реализованное приложение адаптировано к изменению, добавлению и удалению статей бюджетирования компании и может быть использовано в работе предприятий с любым набором статей бюджетирования.

Список литературы

1. Habr.com [Электронный ресурс] : Что такое 1С. О сложной системе простыми словами. – Режим доступа : <https://habr.com/ru/company/trinion/blog/250893/>
2. Панов М. М. Постановка системы бюджетного управления или три координаты бизнеса: БДР, БДДС, ББЛ / М. М. Панов. – Инфра-М, 2014. – 5 стр.
3. Радченко М. Г. 1С:Предприятие 8.2. Практическое пособие разработчика. Примеры и типовые приемы / М. Г. Радченко. – ООО «1С-Паблишинг», 2013. – 207 с.

Разработка приложения для автоматизации работы с данными в области клинической фармакологии

М. С. Гончаров

Студент бакалавр

С. В. Борзунов

Доцент

Введение

В организации деятельности учреждений медицинского профиля большое значение имеют вопросы учета лекарственных средств, хранения и обработки данных о суммах затрат на курсы лечения пациентов, контроль ошибок, которые могут возникнуть при заполнении документации, а также создание отчетов за тот или иной период. Современный подход к решению указанных вопросов заключается в автоматизации процессов работы с данными, которая может быть достигнута с помощью применения специализированного программного обеспечения.

Целью работы является создание десктопной программы, позволяющей решать ряд задач автоматизации анализа данных в области клинической фармакологии.

1. Основные требования к разрабатываемой программе

Требования к составу и параметрам технических средств разработки обусловлены тем, что программа будет использоваться в различных учреждениях здравоохранения. В связи с этим принимались во внимание следующие требования:

- реализация в формате десктопного приложения;
- формирование базы данных, в которой должны храниться сведения о лекарственных средствах;
- возможность интегрируемости базы данных для нескольких медицинских учреждений, которые нуждаются в сборе и обработке данных;
- возможность добавлять сведения в существующую базу данных;
- загрузка и получение информации с помощью файлов формата табличного процессора.

2. Реализация приложения

Требование необходимости работы программы без привлечения дополнительно устанавливаемых модулей и библиотек определило язык программирования, выбранный для реализации, – C#. Для работы с файлами табличного процессора используется стандартная C#-библиотека Microsoft.Office.Interop.Excel. В качестве базы данных используется СУБД SQLite [1, 2].

Блок-схема алгоритма работы программы представлена на рис. 1. Существуют следующие режимы ее функционирования.

Первый режим: пользователь загружает сведения о затратах на закупку лекарственных средств в базу данных медицинских препаратов с помощью форм ввода данных или с помощью файлов формата *.xls после чего производится запрос к ней.

Второй режим: пользователь делает запрос к уже сформированной базе данных. В конце работы программы результат по запросу можно получить в виде файла формата *.xls или же отобразить на экране монитора для быстрого просмотра.

Загрузка данных о пациентах, принимающих различные препараты, а именно: количество таблеток, предусмотренных для одного приема, номер курса терапии, дозы препаратов, стоимость лечения и др. – осуществляется также двумя способами: самостоятельно или с помощью файлов табличного процессора.

Заполнение базы данных происходит с помощью Microsoft.Office.Interop.Excel-библиотеки. При этом для того, чтобы обратиться к ячейке файла формата *.xls, необходимо последовательно создать переменные [1].

Ниже приводится пример инициализации переменных, необходимых для обращения к ячейке файла Excel, (листинг 1) и записи в базу данных информации из ранее заполненной ячейки файла (листинг 2) [1, 2].

Листинг 1

Инициализация переменных

```
...
// Объявляем приложение
Excel.Application ex = new Excel.Application();
// Открываем книгу
Excel.Workbook workbook = ex.WorkBooks.Open(NameExcel, 0,
false, 5, "", "", false, Excel.XlPlatform.xlWindows, "", true,
false, 0, true, false, false);
// Получаем нужный лист документа (для примера возьмем первый)
Excel.WorkSheet sheet =
(Excel.Worksheet)ex.Worksheets.get_Item(1);
...
```

Запись в базу данных

```

...
SQLiteConnection Con = new SQLiteConnection("Data
Source=|DataDirectory|DB.sqlite");
SQLiteConnection com;
String query;
Con.Open();
for (int i = 0; i < data.Rows; i++)
{
    for (int j = 0; j < data.Columns; j++)
    {
        query = "INSERT INTO Patient (ID, FullName) values ('" +
sheet.Cells[i, j] + "'";
    }
}
Con.Close();
...

```

База данных представлена двумя сущностями, ее схема изображена на рис. 2.

Заметим, что в программе обеспечена устойчивость к наличию ошибок при вводе данных.

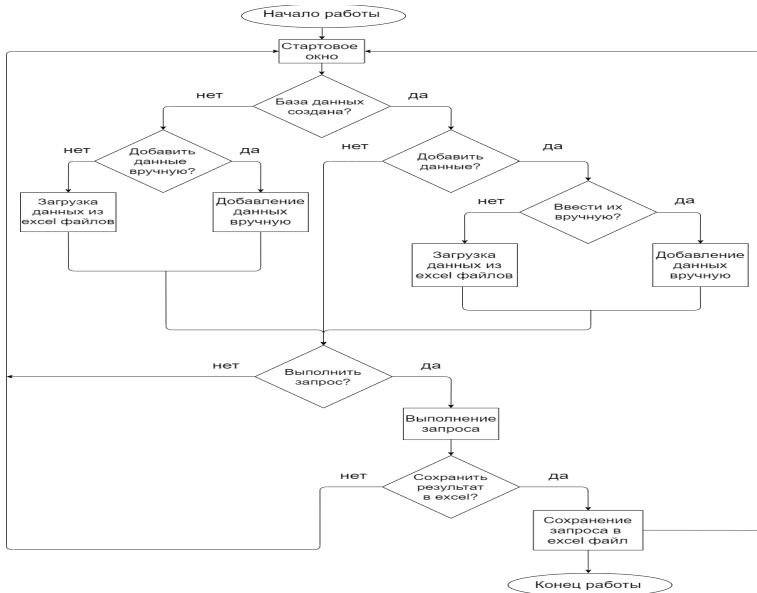


Рис. 1. Блок-схема разработанной программы

Пациент			
id	integer		
ФИО	string		
Add field			

Препарат			
id	integer		
Количество таблеток в 1 приём	float		
Курс терапии	integer		
Доза в таблетке	float		
Суточная доза	float		
Стоимость препарата	float		
Add field			

Рис. 2. Визуализация сущностей базы данных

3. Интерфейс пользователя

Приложение имеет многостраничный интерфейс. В качестве примера рассмотрим отображение выбора нужных параметров, характеристик для выполнения запроса (см. рис. 3).

В интерфейсе реализован выбор нужных характеристик запросов, которые необходимы для составления экономического отчета.

На рис. 3 показаны параметры, по которым можно запрашивать информацию, такие как стоимость курса терапии, стоимость одного дня, название препарата и прочие.

На форме присутствуют две кнопки для сохранения результатов. Одна из них позволяет увидеть данные непосредственно, другая же сохраняет их в отдельные файл формата *.xls.

Рис. 3. Интерфейс запроса в разработанной программе

Заключение

Создана десктопная компьютерная программа, обеспечивающая формирование базы данных, содержащей информацию о стоимости лекарственных препаратов. В процессе реализации были решены поставленные задачи: создана реляционная база данных, которая предусматривает возможность выполнения запросов для создания отчетных листов. Также предусмотрена возможность интегрируемости данных для нескольких профильных учреждений.

Список литературы

1. Албахари, Дж. С# 6.0. Справочник. Полное описание языка / Дж. Албахари, Б. Албахари – М. : «Вильямс», 2018. – 1040 с.
2. Шелдон, Р. MySQL 5: базовый курс = Beginning MySQL / Р. Шелдон, Дж. Мойе. – М. : «Диалектика», 2007. – 880 с.

Разработка системы генерации естественно-языковых поисковых запросов по структуре онтологии для обогащения ее новыми фактами

В. С. Господарикова

Студент бакалавр

В. В. Гаршина

Доцент

Введение

Каждый день в мире формируется и записывается множество новых данных. Они могут сохраняться в базах данных, документах и статьях, распространяясь при помощи сайтов новостей и других средств массовой информации. Рост количества данных приводит к затруднению в их обработке и структуризации, поиске в них нужной информации.

Онтологии способны решить проблему в структуризации данных для последующего их использования и обработки, так как являются относительно понятной, выстроенной по связям формой представления. Однако, когда онтологии разрастаются в графы с более чем тысячами вершинами, человеку уже трудно самостоятельно поддерживать, обновлять и дополнять их, потому что для нас непривычен и практически не осуществим процесс оперирования таким огромным количеством данных и их связей [1]. Тогда становится необходимой автоматический поиск фактов и дальнейшая генерация запросов для уточнения, дополнения и обновления онтологии.

В данной статье мы рассматриваем алгоритм поиска по онтологии тех фактов, которые требуют уточнения, дополнения или обновления, а, следовательно, генерации запроса.

1. Постановка задачи

Проблема поиска по онтологии фактов, требующих дополнения, уточнения или обновления возникла в реально существующей онтологии. Она была создана посредством кроулинга новостных сайтов. Направленность онтологии предполагает нахождение информации о договорах, компаниях, их сотрудниках и т.п. Из новостей и статей, которые были найдены, система сбора данных для онтологии извлекает факты и отношения при помощи поиска контекстно-свободной

грамматики и специально обученной нейронной сети. Полученные таким образом данные добавляются в онтологию, которая хранится и функционирует в среде семантической графовой базы данных «GraphDB» разработанной ontotext [2].

К сожалению, факты в такой онтологии зачастую неполные, поскольку в одной новостной статье может не содержаться всей информации о договоре или компании, например. Тогда возникает задача поиска и добавления в онтологию дополнительной информации. Но перед этим проблема поиска фактов, в которых этой информации не хватает.

С другой стороны, несколько статей может писать об одном и том же договоре, а значит, такие данные могут добавиться в базу не один раз, образовав избыточность, от которой также следует избавиться.

И наконец, даже полные и не избыточные данные нуждаются в перепроверке со временем, поскольку имеют свойство устаревать и терять свою ценность. Так, например, сотрудник может сменить должность или место работы, а в базе останется запись о его старых данных.

Таким образом, необходимо разработать алгоритм, способный учитывая заданные особенности добавления и изменения данных искать факты, для которых необходимо сгенерировать запрос.

2. Алгоритм поиска фактов для дополнения по онтологии

В данном разделе описывается алгоритм, разработанный для поиска по онтологии фактов, требующих генерации запроса на уточнение, дополнения или обновления. Далее на рис. 1 показана блок-схема этого алгоритма.

Как видно из схемы первым этапом для поиска по онтологии фактов, требующих генерации запроса является выборка из самой базы объектов, которые проверки. Для этого необходима структура онтологии, которая содержится в самой онтологии, а также возможно добавление фильтра, под которым подразумевается, например, конкретный тип класса объектов, требующих проверки.

Далее производится выборка соответствующих объектов из графовой базы данных, в которой содержится онтология. Также для описываемого алгоритма возможна замена базы данных на простой файл описания онтологии (.owl/.rdf).

После получения выборки фактов начинается итерирование по ним, чтобы проверить по отдельности каждый и составить список сгенерированных запросов. Важно, что для каждого запроса обязательно указание ссылки на конкретный объект, так как добавление и изменение информации в онтологии может проводиться только с этим условием.



Рис. 1. Блок-схема алгоритма

Теперь рассмотрим цикл, который для каждого отдельного объекта проверяет необходимость генерации запроса, а также его тип.

В данном алгоритме есть три типа запроса:

- Поиск недостающей информации;
- Запрос на удаление дубликатов;
- Запрос на проверку актуальности информации.

По ходу выполнения производится проверка на необходимость генерации запроса сначала для поиска недостающей информации. Если предполагается, что информация достаточно полная, то алгоритм проходит к поиску дубликатов, иначе будет произведена генерация запроса и выход из цикла, потому что дальнейшие проверки не имеют смысла и полезного действия.

С последующими двумя проверками: поиск дубликатов и проверка актуальности информации (или её срока годности), - происходит аналогичная обработка.

Сама логика алгоритма предполагает, что для каждого объекта за один проход алгоритма возможен максимум один запрос, а значит, для полной проверки базы необходим тройной запуск алгоритма.

Хотелось бы обратить особое внимание на второй вариант запроса – запрос на удаление дубликатов, так как он при дальнейшей обработке требует наличия встречного запроса от объектов, которые были признаны дубликатами для того, который прикреплен к запросу.

Это необходимо, так как алгоритм предполагает за каждую итерацию всего один запрос для записи. Тогда, дубликатом может быть признан объект, который находится на стадии дополнения и может ещё измениться, получив существенные дополнения, что может привести к ошибочному удалению данных.

В конце алгоритма сформированные запросы с прикрепленными к ним ссылками или метками объектов передаются в качестве результата на выход. В дальнейшем, они могут быть обработаны, как обычные поисковые запросы, как информационной системой поиска, так и человеком, поскольку формирование всех запросов предполагается на естественном языке.

При этом для самих записей предполагается отмечать, на какой стадии проверки они закончили в предыдущий запуск алгоритма, чтобы не перепроверять одни и те же запросы. Так изначально все записи начинают в нулевой стадии. После формирования запросов на дополнение сам алгоритм переводит их в первую. Запрос на дубликаты может быть отмечен, как пройденный, только самим обработчиком запросов, в случае, если запрос сформирован, так как дубликат может

быть в стадии дополнения, или алгоритмом, если дубликатов нет. Последняя проверка не изменяет стадию и совершается всегда.

Заключение

Этот алгоритм был представлен в виде блок-схемы, а также его нюансы работы описаны в описании работы алгоритма.

Подводя итоги, хотелось бы отметить, что данный алгоритм во многом является универсальным.

Во-первых, он не зависит от предметной области самой онтологии, потому что предполагает работу с её структурой без опоры на семантическую составляющую.

Также, данный алгоритм был разработан для реализации на языке Java, однако он может также быть запрограммирован под любой другой язык, т.к. выбранное средство реализации хранилища онтологии имеет удобный и универсальный API для запросов, а сам алгоритм не завязан на определённых свойствах и возможностях конкретного языка.

Таким образом, в результате проделанной работы был разработан универсальный алгоритм поиска по онтологии фактов, требующих обогащения новыми фактами, а также удаления избыточности и поддержания актуальности информации, для генерации по ним запросов естественного языка для дальнейшей обработки их поисковыми системами.

Список литературы

1. Митрофанова, О. А. Онтологии как системы хранения знаний / О. А. Митрофанова, Н.С. Константинова – СПбГУ, 2008. – 54 с.
2. GraphDb documentation [Электронный ресурс]: документация – Режим доступа : <https://graphdb.ontotext.com/documentation/free/>

Программный модуль для работы с торговыми наименованиями лекарственных препаратов

А. В. Данилова

Студент бакалавр

С. В. Борзунов

Доцент

Введение

Проведение непрерывного анализа экономических параметров деятельности учреждений является важной задачей их работы, и ее решение требует привлечения автоматизированных средств обработки данных. В частности, в рамках работы медицинских учреждений возникает следующая проблема: лекарственные препараты с одинаковым действующим веществом могут иметь разные торговые наименования, при этом анализ нужно проводить по международному непатентованному наименованию (МНН) – уникальному наименованию действующего вещества лекарственного средства. Однако исходные данные, которые поступают медицинским работникам, содержат только торговые наименования препаратов. В связи с этим, возникает необходимость устанавливать соответствие между МНН и торговым наименованием препарата.

Описанная выше проблема возникает, например, при проведении ABC/VEN-анализа. ABC-анализ позволяет разделить все используемые учреждением лекарственные препараты на три класса в зависимости от объемов их потребления (объем потребления определяется путем умножения стоимости одной упаковки препарата на количество упаковок). Проведение вместе с ABC-анализом и другого вида анализа, называемого VEN-анализом, позволяет оценить уровень рационального использования бюджета медицинского учреждения для приобретения различных категорий лекарственных препаратов.

В настоящей работе рассмотрено решение проблемы установления соответствия между МНН и торговым наименованием препарата в рамках информационной системы для проведения ABC/VEN-анализа.

1. Проектирование информационной системы

Информационная система для проведения ABC/VEN-анализа состоит из четырех логических частей (рис. 1):

- библиотеки EconomicAnalysisLibrary, содержащей функции для проведения необходимых расчетов;
- библиотеки Excel для работы с файлами формата *.xlsx;
- базы данных;
- десктопного приложения, которое предоставляет пользователю возможность работать с базой данных лекарственных средств (добавлять, редактировать и удалять медицинские препараты), а также проводить ABC/VEN-анализ и выгружать результаты в формате .xlsx

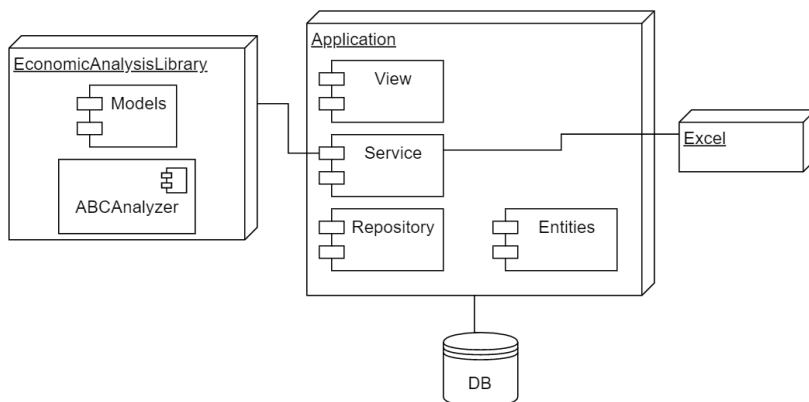


Рис. 1. Архитектура информационной системы

Рассмотрим схему базы данных. В ней содержится информация о МНН, соответствующих им торговых наименованиях препаратов и их категориях VEN (рис. 2).

Как видно из рисунка, в базе данных содержится две сущности: Medicine и INN.

Сущность Medicine описывает лекарственные препараты и содержит следующие поля:

- идентифицирующий номер;
- идентифицирующий номер МНН, которое соответствует данному лекарственному средству;
- название;
- категория VEN.

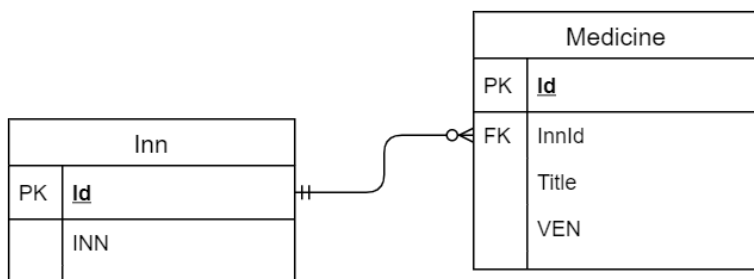


Рис. 2. Схема базы данных

Сущность INN является описанием МНН и содержит два поля: идентифицирующий номер и МНН.

Данные сущности связаны отношением один-ко-многим: одному и тому же МНН может соответствовать несколько лекарственных препаратов, но каждому препарату соответствует только одно МНН.

Категория VEN, к которой относится лекарственное средство, является полем сущности Medicine, потому что препараты, соответствующие одному и тому же МНН, могут принадлежать разным категориям VEN.

Для реализации информационной системы были выбраны следующие технологии [1, 2]:

- в качестве языка программирования использован язык C#, который обеспечивает полную поддержку объектно-ориентированного программирования (включая инкапсуляцию, полиморфизм и наследование);
- для написания десктопного приложения принято решение применить Windows Presentation Foundation (WPF), так как он позволяет создавать визуально привлекательный интерфейс;
- в качестве базы данных взята LiteDB – легковесная и быстродействующая документ-ориентированная база данных (БД), предназначенная для .NET.

2. Реализация программного модуля для работы с торговыми наименованиями лекарственных препаратов

Модуль, который устанавливает соответствие между МНН и торговым наименованием препарата, расположен в слое Service десктопного приложения (рис. 1). Блок-схема алгоритма установления соответствия приведена на рис. 3.

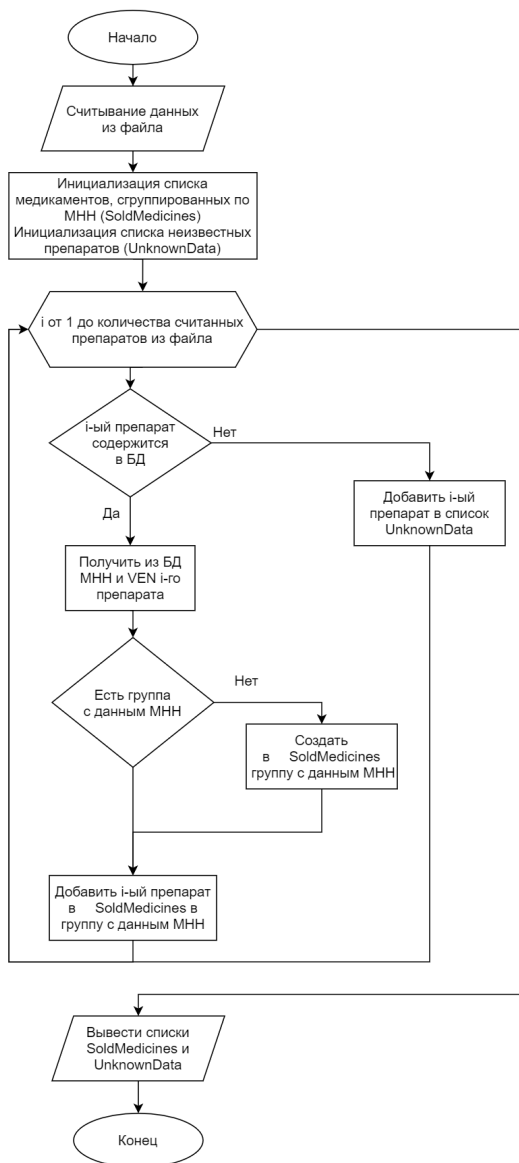


Рис. 3. Алгоритм установления соответствия между МНН и торговым наименованием препарата

Рассмотрим более подробно этапы этого алгоритма.

1. Считывание данных из файла. На данном шаге с помощью функций из библиотеки Excel производится считывание торговых наименований лекарственных препаратов из указанного пользователем файла.

2. Инициализация списка препаратов, сгруппированных по МНН (SoldMedicines); инициализация списка неизвестных препаратов (UnknownData).

3. Цикл по считанным из файла лекарственным препаратам. Если препарат с данным торговым наименованием, содержится в базе данных (в таблице Medicine), то из базы данных считывается информация о его МНН и VEN, и он добавляется в список препаратов, сгруппированных по МНН. Если рассматриваемого лекарственного средства не оказалось в базе данных, то он добавляется в список неизвестных препаратов.

4. Вывод списка препаратов, сгруппированных по МНН, и списка неизвестных препаратов. Далее, если список неизвестных препаратов не будет пустым, то пользователю будет предложено внести информацию об этих лекарственных средствах. Список препаратов, сгруппированных по МНН, будет использован далее при проведении расчетов для ABC/VEN-анализа.

Заключение

В настоящей статье рассмотрено решение проблемы установления соответствия между МНН и торговым наименованием препарата в рамках информационной системы для проведения ABC/VEN-анализа. Разработанный программный модуль позволит работникам медицинских учреждений существенно сократить объем ручного труда при проведении данных видов анализа. Кроме того, разработанный модуль обладает высоким уровнем переносимости и может быть использован в других информационных системах.

Список литературы

1. Андерсон, К. Основы Windows Presentation Foundation / К. Андерсон. – СПб. : БХВ-Петербург, 2008. – 432 с.
2. Язык программирования C#. Классика Computers Science / Хейлсберг А. [и др.]. – 4-е изд. – СПб. : «Питер», 2012. – 784 с.

Разработка программного обеспечения для решения проблемы восстановления лица в формате 3D модели

В. В. Дмитриев

Студент бакалавр

Е. Ю. Митрофанова

Доцент

Введение

В настоящее время 3D-моделирование является одним из наиболее интенсивно развивающихся направлений компьютерной графики. 3D-модели лиц активно используются для создания персонажей игр и фильмов с высокой детализацией лиц, для использующих лицо мобильных приложений и, конечно, биометрических систем. Таким образом, повсеместное использование 3D-моделей лиц, становится не только необходимостью, но и реальностью [1]. Это обусловлено мощным скачком в развитии компьютерных технологий – увеличением памяти и быстродействия компьютерных систем, эксплуатацией новейших методов обработки данных и цифровых изображений на основе математического моделирования, а также появлением новых способов и технологий сканирования 3D-объектов и получения их цифровых моделей.

Однако даже новейшие способы сканирования не являются полностью идеальными. Одной из основных проблем при создании 3D-модели лица является проблема появления пустот на результирующей поверхности, которая может возникать в результате использования неточно подобранного освещения, некачественного оборудования или устаревших программных средств. Например, плохое освещение сканируемого лица, наверняка отрицательно повлияет на итоговый результат, на резких перепадах, таких как кончик носа, надбровные дуги и т. д., могут образоваться «провалы» [2, 3]. Для решения этой проблемы возможно применение метода реконструкции 3D-модели лица, на основе которого и будет строиться реализация прикладного программного обеспечения, которому и посвящена данная работа.

1. Предобработка данных и формирование «Range Image»

«Range Image» – спроектированный ортографически на регулярную сетку и пересчитанный набор координат x_l, y_l, z_l облака точек. Может быть представлен в виде изображения и графика, пример представлен на рис. 1.

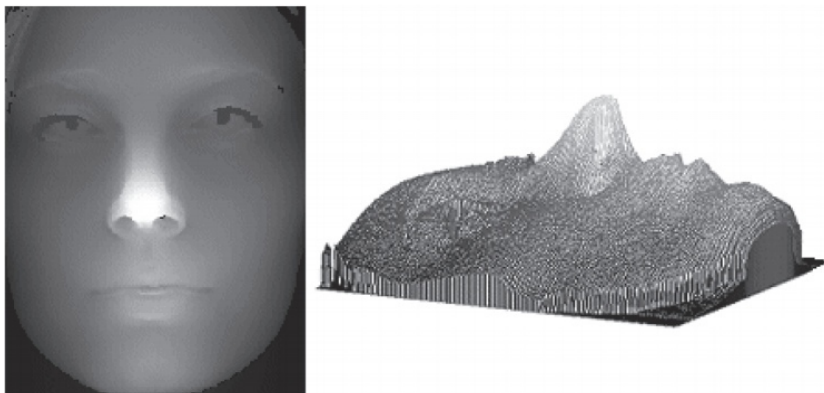


Рис. 1. Пример видов «Range Image»

«Range Image» необходим нам для вывода промежуточных и итоговых результатов работы алгоритмов. Он создается в несколько этапов.

Этап первый – получим облако точек в результате сканирования лица человека сканером-камерой или из готовой 3D-модели. В результате получим облако точек p_l , где $l = 1, 2, 3, \dots, L$. Параметр L может варьироваться от нескольких сотен до нескольких десятков тысяч и каждая точка имеет свои координаты x_l, y_l, z_l .

Этап второй – выполним адаптивную триангуляцию этих точек. Свяжем каждые три координаты z_l в треугольник, вершины которых соединим с ближайшими соседями, образуя новые треугольники. В результате получим триангуляционную сетку.

Этап третий – перенесем координаты x_l, y_l на регулярную сетку и выполним пересчет координат z_l . Каждый треугольник обработаем, так, чтобы вычислить все значения на области этого треугольника и спроектируем ортографически полученные значения на регулярную сетку.

Этап четвертый – полученный результат выровняем по всем направлениям и приведем в шкалу GRAY. Таким образом получаем «Range Image».

2. Восстановление поверхности 3D-модели с помощью алгоритма переноса на основе метода обратных расстояний

Восстановление 3D-модели по MOP подразумевает под собой алгоритм реконструкции 3D-модели лица, в случае появления на ней артефактов – провалов, пустот. Данная проблема возникает при создании «Range Image» или 3D-объекта из некачественно созданного облака точек. Пример проблемы представлен на рис. 2.

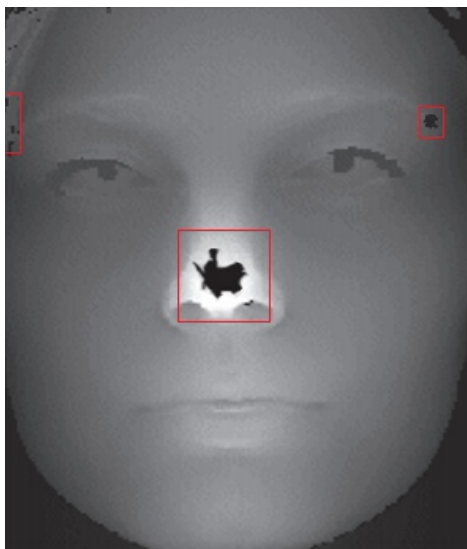


Рис. 2. Пример проблемы появления пустот на «Range Image» после обработки некачественного облака точек

Алгоритм переноса по методу обратных расстояний состоит из двух этапов.

На первом этапе выполняются следующие действия.

1. Получим на вход облако точек p_l .
2. Представим координаты x_l, y_l вектором K в комплексной форме следующим образом:

$$K = X + jY, j = \sqrt{-1}. \quad (1)$$

3. Сформируем матрицу K_m путем повторения вектора K L раз:

$$K_m = [K \dots K], \quad (2)$$

где размер матрицы: $L \times L$.

4. Вычислим дистанции между точками:

$$D = \text{abs}(K - K^T). \quad (3)$$

5. Вычислим вектор B параметров регрессии

$$B = D^{-1}Z. \quad (4)$$

На втором этапе выполняются следующие действия.

1. Заданные x_l, y_l , запишем в векторы X и Y и рассчитаем границы координат. Определим шаг дискретизации и значения координат для необходимого размера «Range Image», а затем создадим рабочий массив размера $M \times N$. Все это выполняется в рабочей среде MATLAB.

2. Для всех значений координат сформируем вектор X_Y , состоящий из L строк:

$$X_Y = \begin{pmatrix} X_r(n) + jY_r(m) \\ \vdots \\ X_r(n) + jY_r(m) \end{pmatrix} \quad (5)$$

3. Вычислим расстояния между исходными и новыми координатами:

$$D_{new} = \text{abs}(X_Y - K) \quad (6)$$

4. Вычислим значения для каждого пикселя «Range Image»:

$$I(m, n) = D_{new}^T B \quad (7)$$

Суть алгоритма состоит в реконструкции поврежденных облака точек путем переноса с нерегулярной сетки на регулярную с интерполяцией по методу обратных расстояний. Данный алгоритм для рабочей среды MATLAB представлен на рис. 3.

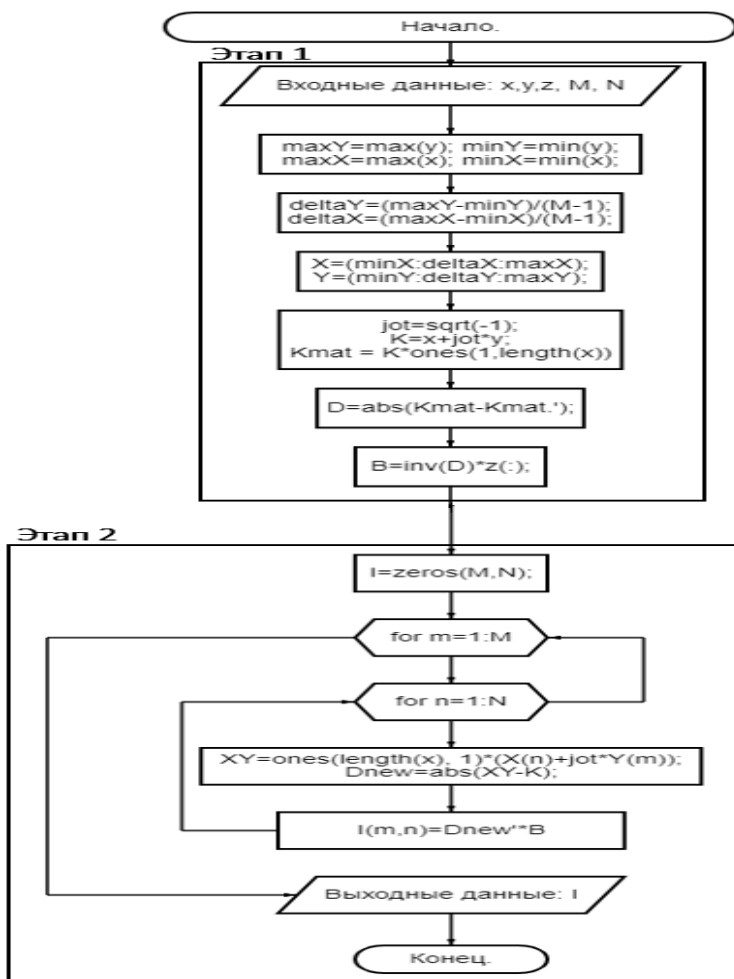


Рис. 3. Блок-схема алгоритма переноса по MOP

В результате работы алгоритма получаем обработанный «Range Image» без пустот.

Пример обработанного «Range Image» представлен на рис. 4.



Рис. 4. Обработанный алгоритмом «Range Image»

3. Тестирование разработанного программного обеспечения

В результате разработки было создано приложение для восстановления поврежденных 3D-моделей лиц. Схема работы приложения предоставлена на рис.5.

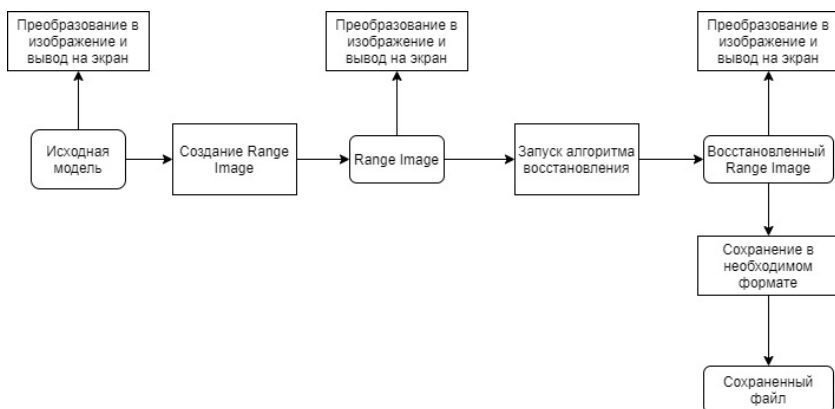


Рис. 5. Схема работы приложения

Пользователь может выбрать файл для восстановления, просмотреть промежуточный «Range Image» данного файла и при

наличии повреждений запустить алгоритм восстановления. После этого реконструированный «Range Image» будет выведен на экран и с согласия пользователя восстановленный файл будет сохранен.

Заключение

Данная статья посвящена разработке программного обеспечения для решения проблемы восстановления поверхности лица в формате 3D-модели. Был описан алгоритм реконструирования поврежденной модели с помощью метода обратных расстояний, была составлена его блок-схема для рабочей среды MATLAB. В итоге процесса разработки получилось полностью рабочее, протестированное, приложение, с интегрированным алгоритмом восстановления, пригодное к автономной работе на любой платформе.

Список литературы

1. Кухарев Г. А. Методы обработки и распознавания изображений лиц в задачах биометрии / Г. А. Кухарев, Е. И. Каменская, Ю. Н. Матвеев, Н. Л. Щеголева; под ред. М. В. Хитрова. – СПб. : Политехника, 2013. – 388 с.: ил.
2. Dense 3D Point Cloud Reconstruction Using a Deep Pyramid Network / Priyanka Mandikal [и др.] [Электронный ресурс]. – Режим доступа : <https://ieeexplore.ieee.org/abstract/document/8659320>
3. Learning Efficient Point Cloud Generation for Dense 3D Object Reconstruction / Chen-Hsuan Lin [и др.] [Электронный ресурс]. – Режим доступа : <https://ojs.aaai.org/index.php/AAAI/article/view/12278>

Теорема Шеннона–Котельникова и формула суммирования Пуассона

Е. В. Дунаева

Студент бакалавр

Н. Б. Кожевникова

Студент бакалавр

Д. М. Серадская

Студент бакалавр

Л. А. Минин

Доцент

Введение

Теорема Шеннона-Котельникова (теорема отсчётов) является одним из ключевых утверждений в современной цифровой обработке сигналов. Фундаментальная работа В. А. Котельникова «О пропускной способности «эфира» и проволоки в электросвязи» была в 1933 году напечатана в одном из отраслевых сборников [1]. Попытка опубликовать эту работу, содержащую математическое обоснование возможности цифровой передачи информации, в 1936 году в журнале «Электричество» оказалось неудачной. Причиной отказа, по мнению редакции журнала, стали «перегруженность портфеля журнала и узкий интерес его статьи».

Данную проблему затрагивал в 1929 г. английский радиоинженер Г. Найквист. Он одним из первых выразил мысль, что выборки сигнала должны различаться интервалами времени, равными приблизительно обратной полосе его спектральной ширины. Однако рассуждения Найквиста относились к проблеме неискаженной передачи телеграфного (цифрового) сигнала.

Теорема отсчётов стала широко известной лишь после того, как американский математик К. Шеннон в 1948 г. доказал ее вновь [2]. Позднее было обнаружено, что похожую теорему в 1915 г. получил английский математик Э. Уитакер при исследовании проблем

интерполяции целых функций [3]. С 1977 г. теорему отсчётов стали называть WKS-теоремой – теоремой Whittaker-Kotelnikov-Shannon.

В настоящей статье показывается, что теорема отсчётов является частным случае формулы суммирования Пуассона. С учётом того, что эта формула была получена С. Пуассоном в 1827 г., вопрос о временных рамках WKS-теоремы снова становится открытым.

Перейдём к строгим формулировкам и доказательству нашего утверждения.

1. Формулировка теоремы отсчётов

Рассмотрим преобразование Фурье функции $f(x)$ в следующей нормировке

$$(Ff)(\xi) = \widehat{f}(\xi) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ix\xi} dx \quad (1)$$

Важную роль играют функциональные классы, которым принадлежат $f(x)$ и $\widehat{f}(\xi)$. Наложим на эти функции, для упрощения изложения, несколько завышенные требования гладкости. Будем предполагать, что $f(x)$ и $\widehat{f}(\xi)$ непрерывны на всей числовой оси и удовлетворяют оценкам

$$|f(x)| \leq \frac{C_1}{(1+|x|)^{1+\varepsilon_1}}, x \in \mathbb{R}, |\widehat{f}(\xi)| \leq \frac{C_2}{(1+|\xi|)^{1+\varepsilon_2}}, \xi \in \mathbb{R} \quad (2)$$

с некоторыми константами $C_1, C_2, \varepsilon_1, \varepsilon_2 > 0$.

В цифровой обработке сигналов вводится специальная функция отсчётов $\text{sinc}(x) = \sin(x)/x$, для которой справедливы следующие свойства:

$$\int_{-\infty}^{\infty} \text{sinc}(\pi x) dx = \int_{-\infty}^{\infty} \frac{\sin \pi x}{\pi x} dx = 1, \text{sinc}(\pi m) = \delta_{om} = \begin{cases} 1, m = 0 \\ 0, m \neq 0 \end{cases}$$

Кроме того, целочисленные сдвиги ортогональны друг другу

$$\int_{-\infty}^{\infty} \text{sinc}(\pi(x-k)) \cdot \text{sinc}(\pi(x-m)) dx = \delta_{km}, k, m \in \mathbb{Z}$$

Образ Фурье функции $f(x)$ часто интерпретируется как спектр сигнала. Говорят, что $f(x)$ является функцией с ограниченным спектром, если $\widehat{f}(\xi) = 0$ при $|\xi| > b$. Положительное число b называют

шириной спектра. Для таких функций справедлива теорема Шеннона-Котельникова:

Теорема 1. Пусть $f(x)$ является функцией с ограниченным спектром с шириной спектра b . Пусть выполнено условие Найквиста

$$0 < h \leq \frac{\pi}{b}.$$

Тогда справедливы следующие формулы:

$$1) f(x) = \sum_{k=-\infty}^{\infty} f(kh) \cdot \operatorname{sinc}\left(\frac{\pi}{h}(x-kh)\right); \quad (3)$$

$$2) \hat{f}(\xi) = \frac{h}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} f(kh) e^{-ikh\xi}, \text{ где } \xi \in \left[-\frac{\pi}{h}, \frac{\pi}{h}\right]. \quad (4)$$

Опишем схему доказательства теоремы 1. Функция $\hat{f}(\xi)$ раскладывается в ряд Фурье на отрезке $[-\pi/h, \pi/h]$, коэффициенты ряда Фурье выражаются через значения $f(kh)$. Так получается равенство (4), а для получения (3) надо применить к (4) обратное преобразование Фурье.

2. Формула суммирования Пуассона

Перейдем к формуле Пуассона. Справедлива

Теорема 2. Пусть функция $g(x)$ удовлетворяет условиям (2). Тогда имеет место формула

$$\sum_{k=-\infty}^{\infty} g(x+2\pi k) = \frac{1}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \hat{g}(k) e^{ikx}, x \in \mathbb{R}. \quad (5)$$

Доказательство теоремы 2 приводится, например, в [4, глава 2]. Ключевым моментом является разложение в ряд Фурье суммы из левой части (5), представляющей собой периодизацию функции $g(x)$ по отрезку длины 2π .

Следующая теорема представляет собой основной результат нашей статьи.

Теорема 3. Формула (4) из теоремы 1 является следствием формулы (5) из теоремы 2.

Доказательство. Предположим, что $g(x) = 0$ при $|x| \geq \pi$. Тогда в левой части (5) остаётся только одно ненулевое слагаемое

$$\sum_{k=-\infty}^{\infty} g(x + 2\pi k) = g(x) = \frac{1}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \hat{g}(k) e^{ikx}, x \in [-\pi, \pi].$$

Сделаем замену переменной $x = -\xi h, h > 0$. Получим равенство

$$g(-\xi h) = \frac{1}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \hat{g}(k) e^{-ik\xi h}, \xi \in \left[-\frac{\pi}{h}, \frac{\pi}{h}\right].$$

Введём функцию ... Тогда $\hat{g}(k) = f(kh)$,

$$\hat{f}(\xi) = \left(\hat{g}(t/h)\right)^{\wedge}(\xi) = hg(-\xi h).$$

Последнее равенство есть следствие двух свойств преобразования Фурье:

$$\left(\hat{g}(t)\right)^{\wedge}(\xi) = g(-\xi), \left(g(\xi h)\right)^{\wedge}(t) = \frac{1}{h} \hat{g}\left(\frac{t}{h}\right).$$

Таким образом,

$$\hat{f}(\xi) = hg(-\xi h) = \frac{h}{\sqrt{2\pi}} \sum_{k=-\infty}^{\infty} \hat{g}(k) e^{-ik\xi h}, \xi \in \left[-\frac{\pi}{h}, \frac{\pi}{h}\right]. \quad (6)$$

Теорема доказана.

Заключение

Формула суммирования Пуассона является важным звеном в теории всплесков и фреймов и в современных методах цифровой обработки сигналов. Она не подменяет собой теорему Шеннона-Котельникова, а даёт возможность более эффективного применения её в различных разделах физики.

Недостатком формулы (4) теоремы Шеннона-Котельникова с точки зрения практического применения является то, что ряд в правой части является бесконечным. В реальности же всегда обходятся конечными суммами:

$$\hat{f}(\xi) \approx \frac{h}{\sqrt{2\pi}} f(kh) e^{-ihk\xi}. \quad (7)$$

Применение конечных сумм эквивалентно переходу от интегрального преобразования Фурье к дискретному преобразованию Фурье (ДПФ).

При доказательстве теоремы 1 для разложения $\hat{f}(\xi)$ на отрезке $[-\pi, \pi]$ в ряд Фурье использовалась система функций $e^{-ik\xi}$, отвечающая периодическим краевым условиям. При других краевых условиях можно получить другие варианты этой теоремы. Зависимость ДПФ от выбора краевых условий описана, например, в [5; 6, глава 4]. Когда же речь идет о теореме Шеннона-Котельникова, используются только периодические краевые условия. На наш взгляд, этот вопрос требует дальнейшего изучения, поскольку учет краевых условий обеспечивает большую скорость сходимости рядов из теоремы 1, а значит, и более точные формулы при переходе от непрерывных моделей к дискретным.

Список литературы

1. Научная сессия Отделения физических наук Российской академии наук, посвященная памяти академика Владимира Александровича Котельникова (22 февраля 2006 г.) / Ю. В. Гуляев [и др.] // УФН. – 2006. – Т. 176, № 7. – С. 751-792.
2. Shannon, C. E. A Mathematical Theory of Communication / C. E. Shannon // Bell System Technical Journal. – 1948. – Vol. 27. – P. 379-423.
3. Whittaker, E. T. On the functions which are represented by the expansions of the interpolation-theory / E. T. Whittaker // Proceedings of the Royal Society of Edinburgh. – 1915. – V. 35. – P. 181-194.
4. Чуи, Ч. Введение в вэйвлеты / Ч. Чуи. – Москва : Мир, 2001. – 412 с.
5. Strang, G The discrete cosine transform / G Strang // SIAM Review. – 1999. – Issue 1. – V. 41. – P. 135-147.
6. Самарский, А. А. Методы решения сеточных уравнений / А. А. Самарский, Е. С. Николаев. – М. : Наука, 1978. – 592 с.

Реализация и исследование методов слежения за движущимися объектами в видеопотоке

М. А. Дынин

Студент бакалавр

В. А. Степанцов

Доцент

Введение

В настоящее время во многих социальных и производственных областях человеческой деятельности получили распространение различные системы, ориентированные на видеонаблюдение как статических, так и динамических объектов. Они находят широкое применение в сферах обеспечения общественной безопасности, безопасности дорожного движения, мониторинга поверхности земли с воздуха, контроля качества изделий на этапах процесса их производства и т.д. Для этих областей актуально решение задачи слежения за движущимися объектами на движущемся фоне.

1. Основная часть

Трекинг – это процесс отслеживания движения объекта в кадре. По окончании трекинга программа «запоминает» и сохраняет траекторию движения, чтобы потом применить её к объекту, которого изначально не было на видео: подписи, изображению, иконке, маске – буквально любому графическому элементу.

В случае слежения за объектом выбирается нужный объект путем его выделения. Для этого необходимо задать его образ и характеристики.

Существует несколько способов представления образа объекта:

– Точки. При данном подходе объект представляет собой набор из одной или нескольких точек.

– Примитивные геометрические фигуры. В данном случае перемещения объекта обычно моделируются как аффинное или проективное преобразование.

– Контур или силуэт объекта. Контур является границей объекта. Силуэт объекта – это площадь внутри контура.

– Сочлененные геометрические фигуры. В этой модели части объекта связаны вместе. Например, тело человека или животного можно представить, как набор соединенных геометрических фигур, представляющих различные части тела.

– Скелетная модель. При этом подходе извлекается скелет. Модель применима как для твердых, так и для подвижных объектов.

2. Задачи, решаемые системами видеонаблюдения

– Распознавание

Для того чтобы распознать модель, необходимо знать, как выглядит объект слежения.

Проблемы: Распознать модель бывает достаточно проблематично в условиях плохой видимости или при наличии перекрытий. Например, когда объект выходит из зоны видимости камеры или перекрывается другим объектом.

– Слежение

Слежение за объектом производится на основе информации о нахождении объекта в предыдущих кадрах, благодаря которым можно прогнозировать следующее положение.

Проблемы: Исчезновение объекта из зоны видимости в большинстве случаев приводит к его потере.

3. Алгоритм TLD

Метод надёжного длительного сопровождения заранее неизвестных объектов в естественной среде. Метод выдерживает разрывы между кадрами, быстрое движение камеры, полное исчезновение, а затем появление объекта. Подход, который использован в данном методе, называется Сопровождение – Обучение–Обнаружение (Tracking–Learning–Detection (TLD)), он сочетает адаптивное сопровождение объекта с обучением детектора объекта в процессе распознавания. Схема алгоритма изображена на рис. 1.

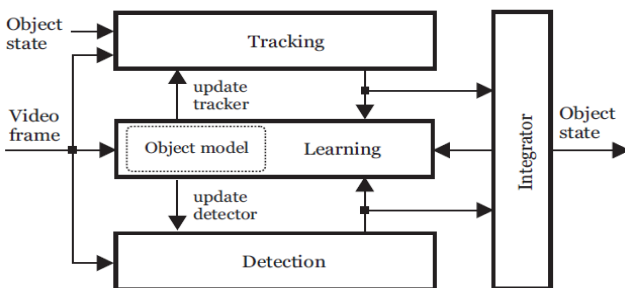


Рис. 1. Схема алгоритма TLD

После того как объект был захвачен при помощи какого-либо метода захвата, траектория объекта начинает наблюдаться двумя процессами (расширяющее и урезающее события). Они строят детектор объекта на лету. Оба процесса делают ошибки, стабильность системы достигается отменой событий. Метод TLD является методом без учителя, но процесс обучения всё же имеется. Обучение происходит «на лету», т. е. в процессе обнаружения объекта, и классификация производится при помощи рандомизированного леса [1].

Объект сопровождается при помощи краткосрочного трекера. Траектория в пространстве признаков анализируется двумя событиями, которые непрерывно пытаются расширить или уменьшить пространство, описываемое моделью. L_t расширяется измерениями, которые вероятнее всего содержат сопровождаемый объект. Эти измерения определяются при помощи расширяющего события. Из L_t удаляются измерения, которые определены как не содержащие объект при помощи обрезающих событий. События работают параллельно, стремясь достичь $L_t \rightarrow L^*$.

Расширяющие события в одиночку приводят к высокому уровню примеси и, следовательно, к детектору низкой точности. Обрезающие события служат отрицательной обратной связью: чем выше уровень примесей, тем больше измерений в модели определяются как неверные и удаляются. Динамическое взаимодействие расширяющих и обрезающих событий является решающим в придании системы стабильности.

Краткосрочный трекер основывается на методе Лукаса–Канаде. (P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features. – 2001. – 3-7 с) Вначале множество ключевых точек извлекается из прямоугольной решётки внутри описанного вокруг сопровождаемого объекта прямоугольника. Затем трекер сопровождает точки от одного кадра до другого, строя разреженное поле движения. Основываясь на поле движения, смещение и изменение масштаба ограничивающего прямоугольника могут быть надёжно оценены как средние значения по распределению. В каждом новом кадре сопровождается новый набор точек, это делает трекер адаптивным. Детектор объекта основан на двухбитных бинарных шаблонах [4]. Эти признаки измеряют ориентацию градиента внутри определённой зоны, квантуют её и выдают 4 возможных кода (рис. 2.)

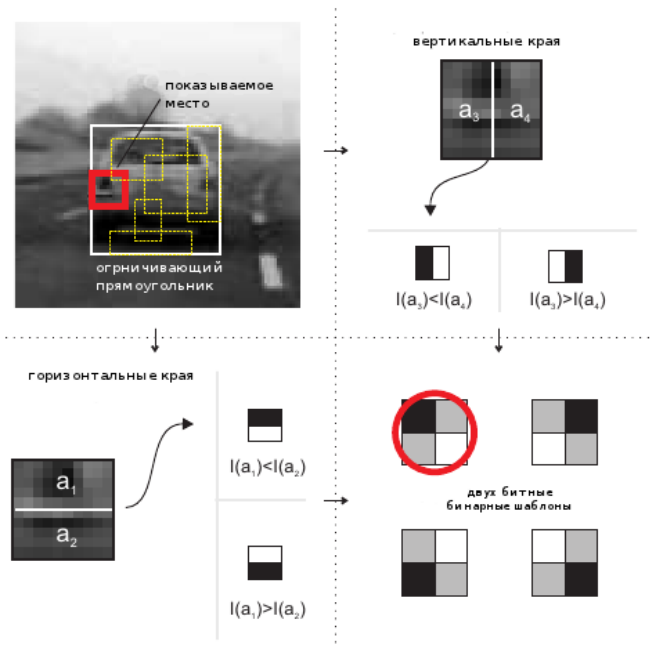


Рис. 2. Принцип работы алгоритма

Таким образом, комбинированные методы имеют высокий уровень обнаружения, позволяют объединять достоинства разных методов поиска [3].

4. Математическое описание алгоритма

Пусть F_t и B_t – кадр видео ряда и описанный прямоугольник сопровождаемого объекта в момент времени t . Пиксели внутри прямоугольника B_t описываются вектором признаков x_t , который содержит информацию о наличии объекта сопровождения. Множество последовательных описанных прямоугольников определяет трек

$$T_t = \{B_0 + B_1 + \dots + B_t\} \quad (1)$$

длины $t+1$, который определяет траекторию объекта в пространстве изображений, T_t^f описывает соответствующую траекторию в пространстве признаков U , которое является подпространством L^* . L^* представляет все возможные состояния наблюдаемого объекта. L^* неизвестно в момент начала сопровождения, когда выбрано первое

измерение $x_0 \in L^*$. Это первое измерение описывает начальное состояние модели в момент времени

$$t = 0: L_0 = \{x_0\}. \quad (2)$$

Компоненты алгоритма взаимодействуют, как описано ниже. Объект сопровождается при помощи краткосрочного трекера. Траектория в пространстве признаков анализируется двумя событиями, которые непрерывно пытаются расширить или уменьшить пространство, описываемое моделью. L_t расширяется измерениями, которые скорее содержат сопровождаемый объект. Эти измерения определяются при помощи расширяющего события. Из L_t удаляются измерения, которые определены как не содержащие объект. Эти измерения определяются при помощи обрезающих событий. События работают параллельно, стремясь достичь $L_t \rightarrow L^*$

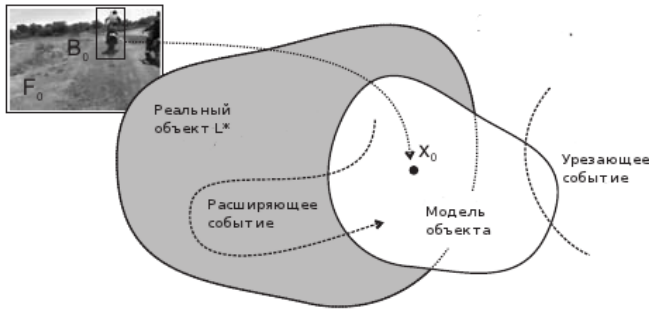


Рис. 3. Модель объекта

Главная цель построения L_t – это предоставить алгоритму память, чтобы создать детектор объекта, который непрерывно обновляется и улучшается. Он сканирует входное изображение F_t и выдаёт на выход множество описанных прямоугольников, которые содержат измерения, входящие в L_t . Эти прямоугольники описывают альтернативные гипотезы к позиции, возвращённой трекером. Слияние гипотез производится взятием позиции, которая минимизирует расстояние до L_t . Отсюда следует, что если положение от трекера очень близко к L_t , то ложные отклики детектора не влияют на трекинг (пока они не станут ещё ближе к L_t , чем позиция от трекера). Минимальное

расстояние до L_t становится очками недоверия к результату, выданному алгоритмом. Опираясь на эти очки, алгоритм решает, виден объект или нет.

Ранее было сказано о событиях, которые наблюдают краткосрочный трекер [3]. Рассмотрим как они работают.

Мы различаем две части L_t :

– правильную часть

$$L_t^c \subset L^* \quad (3)$$

– ошибочную часть

$$L_t^c \not\subset L^*, L_t^c \cup L_t^e = L_t, L_t^c \cap L_t^e = 0 \quad (4)$$

Покрытие является долей измерений с объектом уже открытых неконтролируемым процессом обучения, т. е.

$$C(L_t) = \frac{|L_{ct}|}{|L^*|} \quad (5)$$

Примесь называется доля L_t , которая неверна, т. е.

$$I(L_t) = \frac{|L_t^e|}{|L^*|} \quad (6)$$

Оператор $\|$ обозначает размер множества.

Расширяющие события. В момент времени t краткосрочный трекер создал траекторию

$$T_t^f = \{x_0 + x_1 + \dots + x_t\} \quad (7)$$

Расширяющее событие сначала выбирает определённую часть траектории, которая рассматривается положительной, $P \subset T_t^f$. Модель объекта L_t затем обновляется, т. е.

$$L_t - L_t - 1 + P \quad (8)$$

После этого обновления покрытие модели увеличивается, если P содержит хотя бы одно измерение из L^* . Стратегию выбора части траектории обсудим несколько ниже.

Обрезающие события. Невозможно придумать стратегию выбора, которая бы выбирала только верные измерения. По этой причине примесь L_t постоянно увеличивается. Так как нашей целью является условие $L_t \rightarrow L^*$ то события, уменьшающие примесь модели, являются необходимыми. Если модель характеризуется определённым уровнем

примеси, обрезающее событие необходимо, чтобы идентифицировать подмножество N , которое рассматривается как некорректное и удалить его из модели, т. е.

$$L_t = L_t - 1 - N. \quad (9)$$

Расширяющие события в одиночку приводят к высокому уровню примеси и, следовательно, к детектору низкой точности. Обрезающие события служат отрицательной обратной связью: чем выше уровень примесей, тем больше измерений в модели определяются как неверные и удаляются. Динамическое взаимодействие расширяющих и обрезающих событий является решающим в придании системе стабильности, это будет показано эмпирически в экспериментальном разделе.

5. Вычислительный эксперимент.

Алгоритм был запущен без использования урезающих событий, что должно привести к максимально полной модели. В результате точность распознавания очень быстро упала со временем. Урезающие события служат отрицательной обратной связью. Удаляя неверные измерения из модели, они стабилизируют систему (рис.4). Динамическая устойчивость системы достигается подбором параметров событий.

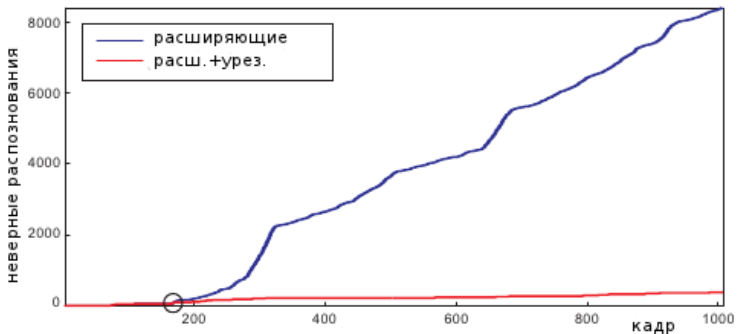


Рис. 4. Число неверных распознаваний быстро растёт со временем, если не используются урезающие события

Заключение

В данной работе был рассмотрен вопрос слежения за объектами в видеопотоке, определены задачи, которые должны решать системы видеонаблюдения, проблемы, которые могут возникнуть во время трекинга и выделен и разобран алгоритм TLD.

Список литературы

1. Tracking-Learning-Detection – [Электронный ресурс]. – https://www.researchgate.net/publication/51866277_Tracking-Learning-Detection (дата обращения: 04.05.2021).
2. Szeliski, R. Computer Vision: Algorithms and Applications. – 2010. – 979 p.
3. Zdenek, K. Tracking-Learning-Detection / K. Zdenek, K. Mikolajczyk, M. Jiri – 2014. – 4-12 p.
4. Viola, P. Rapid object detection using a boosted cascade of simple features / P. Viola, M. Jones – 2001. – P. 3-7
5. Javed, O. Online detection and classification of moving objects using progressively improving detectors / O. Javed, S. Ali, M. Shah – 2005. – 4 p.

Методы и алгоритмы обнаружения и оценки количества объектов в видеопотоке

А. С. Зверева

Студент бакалавр

А. В. Швырева

Ассистент

Введение

С ростом информатизации современного мира стали активно применяться цифровые технологии, связанные с обработкой изображений или видеопотока. В различных сферах деятельности человека возникает потребность подсчета каких-либо объектов в целях повышения производительности труда путем автоматизации процесса работы. Для решения данных проблем все чаще используются системы компьютерного зрения. Алгоритмы распознавания образов позволяют идентифицировать объекты, определять их тип, предоставлять необходимую и качественную информацию.

Видеоаналитика связана с извлечением данных из необработанного видео, которые будут использоваться для дальнейшей обработки в таких методах, как обнаружение, выделение и слежение за объектом в последовательности кадров [1] реальном времени или в виде записей с камер [2].

1. Проблемы, связанные с отслеживанием объектов

Можно выделить следующие основные проблемы, возникающие при слежении за объектом:

1. Повороты объекта. Объект поворачивается относительно всех осей трехмерного пространства.
2. Изменение яркости. Разные области изображения могут изменять свою яркость, одни становиться ярче, другие темнее.
3. Масштабируемость. Возможно приближение или удаления объекта относительно камеры.
4. Перекрывание объекта. детектируемый объект могут перекрывать другие объекты.

5. Шум на изображении. Шум на изображении может появляться вследствие влияния естественных причин (снег, дождь, ветер) и искусственных (наложение фильтров, перекодирование).

6. Изменение цветовой гистограммы. На разных сторонах тела объекта могут быть гистограммы разного цвета.

7. Интенсивность движения. Производительность зависит от интенсивности и сложности движения на видеоряде, поэтому точную алгоритмическую сложность подсчитать невозможно.

2. Методы распознавания объектов

Методы детектирования объектов можно разбить на четыре группы:

- детерминированные, среди которых можно выделить методы поиска оптического потока, поиска особых точек и поиска по шаблону;

- вероятностные, в которых на основе всех предыдущих измерений состояния объекта делается предположение о его следующем состоянии;

- нейросетевые методы основаны на использовании нейронных сетей;

- комбинированные - методы, состоящие из комбинаций различных видов методов.

- В работе будут рассмотрены метод опорных векторов и система YOLO.

1. Support Vector Machine

Метод опорных векторов относится к детерминированным методам распознавания, которые основаны на отказе от статистических моделей данных и постулировании некоторых гипотез относительно метрических свойств классов и образов, представляющих эти классы в многомерном классовом пространстве [3].

Главной целью SVM классификатора является поиск уравнения разделяющие гиперплоскости, которая разделила бы два класса некоторым оптимальным способом.

Общий вид классификатора представлен в (1).

$$g'(x) = \text{sign}(x^T w - b_0) \quad (1)$$

где $x \in R^n, x(x_1, x_2, \dots, x_n)$.

Задача линейно разделяемых классов формулируется в виде (2):

$$\begin{cases} \frac{1}{2} w^T w \rightarrow \min \\ d^{(i)}(x^{(i),T} w - b_0) \geq 0, \quad i = 1, N \end{cases} \quad (2)$$

где w - вектор весовых коэффициентов, b_0 - вспомогательная константа.

Случай линейной разделимости классов с ошибками представлен в формуле (3)

$$\begin{cases} \frac{1}{2} w^T w \rightarrow \min \\ d^{(i)}(x^{(i)T} w - b_0) \geq 1 - \xi, \quad i = 1, N \\ \xi_i \geq 0 \end{cases} \quad (3)$$

где ξ_i - набор дополнительных переменных, характеризующих величину ошибки, C - управляющий параметр, который позволяет находить компромисс между максимизацией разделяющей полосы и минимизацией суммарной ошибки.

В качестве алгоритма нахождения вектора признаков используется HOG - дескрипторы (Histogram of Oriented Gradients), алгоритм вычисления которых представлен ниже.

1. Вычисление градиента для каждого пикселя.
2. Вычисление гистограмм ячеек изображения. Деление на ячейки фиксированного размера и для каждой расчет преобладающих направлений градиента.
3. Формирование и нормирование блоков дескрипторов. Ячейки группируются в блоки для нормирования значений градиентов, учитываются контрастность и яркость каждой ячейки.
4. Классификация HOG-дескрипторов с помощью системы обучения с учителем. В данной работе выбран SVM.

2. YOLO

YOLO (You Only Look Once) – это система обнаружения объектов, использующая сверточную нейронную сеть Darknet-53 для обнаружения динамических объектов. Главным преимуществом системы перед другими архитектурами является скорость работы [4].

На вход системы подается отдельный кадр из видеопотока. Далее с помощью предобученной сверточной нейронной сети формируется карта признаков, после чего происходит агрегирование информации с предыдущих слоев. Последним слоем является прогнозирование:

1. Изображение делится на ячейки размером $a \times a$. Каждая ячейка становится «якорем» для прикрепления прямоугольника, позиции и размеры которого определяются пользователем или автоматически на основе имеющегося набора данных;

5. Кадр прогоняется через нейронную сеть, в которой происходит выделение объекта в единственный прямоугольник. Для этого просматривается каждая ячейка, и если в ней находится объект, то к ячейке прикрепляется «якорь». Далее происходит вычисление характеристики пересечения по формуле (4)

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2}. \quad (4)$$

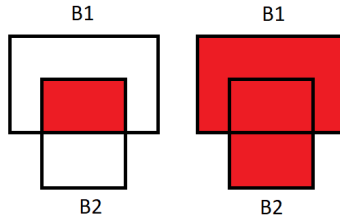


Рис. 1. Пример пересечения прямоугольников

Если характеристика пересечений становится выше некоторого заданного значения, то эти прямоугольник остаются, иначе они удаляются.

Оценка уверенности того, что нужный объект находится внутри прямоугольника, для каждого прямоугольника вычисляется по формулам (5)-(9):

$$b_x = \sigma(t_x) + c_x \quad (5)$$

$$b_y = \sigma(t_y) + c_y \quad (6)$$

$$b_w = p_w e^{t_w} \quad (7)$$

$$b_h = p_h e^{t_h} \quad (8)$$

$$P(obj) \cdot IoU(b, obj) = \sigma(t_0) \quad (9)$$

где c_x, c_y – координаты верхнего левого угла нужной клетки, $\sigma(t_x), \sigma(t_y)$ – координаты точки-якоря внутри клетки, p_w, p_h – ширина и высота прямоугольника начальные, b_x, b_y, b_w, b_h – параметры предсказанного прямоугольника.

$\sigma(t_0)$ – искомая оценка уверенности, которая ищется для каждого прямоугольника. Каждая IoU -характеристика пересечения умножается на вероятность того, что это определенный объект. После этого выбирается лучшая оценка. Если лучший вариант превышает некоторый

порог, то прямоугольник остается. Все еще может оставаться несколько прямоугольников вокруг искомого объекта. Чтобы остался один используется алгоритм подавления немаксимумов.

Для работы с видеопотоком используется технология DeepSORT. Она применяет фильтр Калмана и расстояние Махаланобиса для «переноса» информации из одного кадра в другой. Это необходимо для того, чтобы перекрытый в одном моменте объект и открытый в следующем не считался новым объектом.

3. Оценка количества объектов

В методе SVM количество объектов в кадре подсчитывается путем увеличения счетчика при выявлении нового объекта, и уменьшении счетчика при исчезновении.

В методе YOLO подсчет количества людей происходит путем слежения за пересечением объектов центральной линии - движение вверх и вниз. Данную линию можно расположить произвольным образом: вертикально при изменении направления движения.

В результате проведения сравнения описанных методов было определено, что метод SVM вычисляет ошибочное количество людей, так как при обработке следующего кадра алгоритм может распознать объект как новый или не распознать вовсе. Эта проблема решается технологией DeepSort в методе YOLO.

4. Практическая реализация

Для оценки точности детектирования была разработана программа, реализующая описанные в работе алгоритмы.

Результат работы алгоритма SVM представлен на рисунке 2.

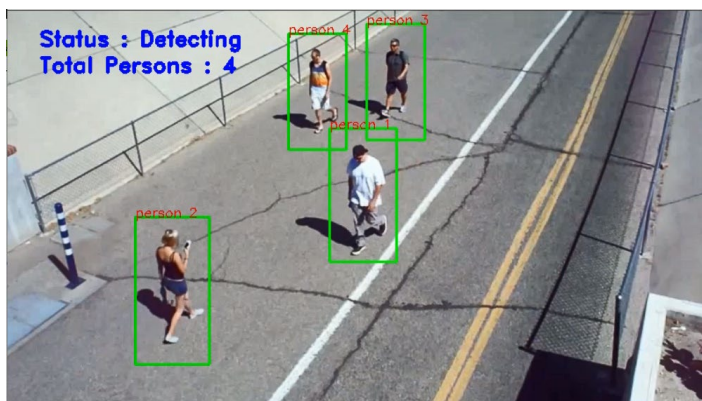


Рис. 2. Работа алгоритма SVM

Результат работы системы YOLO представлен на рисунке 3.

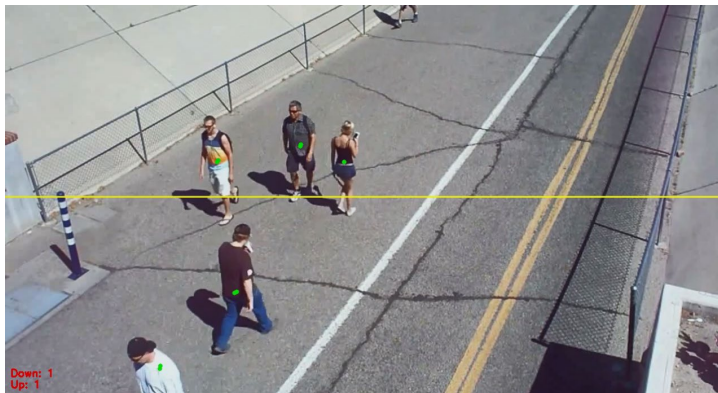


Рис. 3. Работа системы YOLO

Заключение

В результате выполнения данной работы был проведен анализ задачи детектирования объектов в видеопотоке, выявлены основные проблемы при решении данной задачи, а также рассмотрены и реализованы алгоритмы детектирования людей в видеопотоке.

В качестве алгоритмов выделения и слежения за объектами были выбраны и рассмотрены метод SVM, в котором вектор признаков рассчитывается с помощью HOG – дескрипторов и система YOLO.

В результате сравнения двух алгоритмов система YOLO продемонстрировала лучшую скорость выполнения, а также была более эффективна при оценке числа объектов за счет использования технологии DeepSort.

Список литературы

1. Методы автоматического обнаружения и сопровождения объектов. Обработка изображений и управление / Б. А. Алпатов [и др.] // М. : Радиотехника, 2008. – 176 с.
2. Анштедт, Т. Видеоаналитика: мифы и реальность / Т. Анштедт, И. Келлер, Х. Лутц // М. : Security Focus, 2012. – 176 с.
3. Сирота, А.А. Методы и алгоритмы данных и их моделирование в MATLAB: учеб. пособие / А. А. Сирота // БХВ-Петербург – 2016. – 384 с.
4. Survival Strategies for the Robot Rebellion [Электронный ресурс]. – Режим доступа : <https://pjreddie.com>

Подходы к оптимизации выравнивания последовательностей нуклеиновых кислот по алгоритму Нидлмана–Вунша на примере биоинформационного анализа последовательностей вируса бешенства

Н. А. Конищева

Студент бакалавр

П. Д. Меркутова

Студент специалист

Е. В. Манжула

Студент специалист

Я. А. Туровский

Доцент

Введение

Развитие современных вычислительных систем позволило существенно продвинуться в понимании путей патогенеза, то есть путей развития заболевания, и обусловило значительное совершенствование методов исследований в области биоинформатики биологических последовательностей.

Существует обширная группа заболеваний, называемых инфекционными, которые вызываются теми или иными специфическими патогенными возбудителями и передаются от зараженного организма к здоровому. Инфекционные заболевания развиваются за счет сложного биологического процесса, определяющегося взаимодействием патогенного микроорганизма с восприимчивым организмом при определенных условиях. В частности, к этой группе заболеваний относятся инфекционные болезни, вызываемые вирусами. Одним из перспективных методов борьбы с заболеваниями этой группы является анализ последовательностей аминокислот и, в особенности, кодирующих их последовательностей ДНК и РНК возбудителей данных болезней. Анализ данных заключается в прослеживании родственных связей между биологическими последовательностями различных организмов, а также воздействия

метаболических путей возбудителя, с целью предотвращения заболевания и противодействия ему.

Следует подчеркнуть, что классические биологические методы изучения инфекционных заболеваний не всегда считаются приемлемыми по материальным, аппаратным и, в значительной степени, этическим причинам проведения исследований. К таким ограничениям можно отнести невозможность проведения клинических экспериментов сразу на людях, высокую стоимость аппаратуры и материалов, необходимых для проведения экспериментальных исследований. Решение данной проблемы лежит в аспекте разработки ряда методов, основанных на информационных технологиях, позволяющих обрабатывать значительные массивы данных, в том числе биологических. Для этого используется совокупность методов под названием Big Data, которая подразумевает работу с большим объемом информации, так как эти методы позволяют анализировать огромные выборки данных. За счет этого появляется возможность получать более точные результаты, что определяет значительную важность для медицины, в частности, на основе такой области, как биоинформатика.

Биоинформатика определяет серию задач, направленных на решение значительных биологических проблем, в большей степени касающихся манипуляций существенными объемами информации. Таким образом, эта область акцентирует свое внимание на изучении и разработке компьютерных методов, направленных на получение, анализ, хранение, организацию и визуализацию биологических данных.

Одним из базовых направлений, на которых биоинформатика фокусирует свое внимание, является биоинформатика последовательностей. Этот раздел занимается разработкой экспериментальных методов эффективного определения и анализа белковых и нуклеотидных последовательностей. В частности, таким методом биоинформатики является метод выравнивания биологических последовательностей.

На сегодняшний день одним из серьезных инфекционных заболеваний является бешенство. Интерес представляет ситуация циркуляции вируса бешенства в организме носителя заболевания, так как данный механизм остается недостаточно изученным. Таким образом, актуальным представляется изучение посредством цифровых технологий последовательностей нуклеиновых кислот вируса с целью определения возможности его персистенции, то есть длительного нахождения в среде носителя, который может представлять опасность. Для этих целей считается целесообразным использование биоинформатического метода выравнивания последовательностей.

Выравнивание как метод биоинформатики заключается в сравнении нуклеотидных или белковых последовательностей с целью определения сходств и различий между генотипами или белками исследуемых биологических объектов. Таким способом осуществляется оценка генетического материала инфекционного носителя и агента, восприимчивого организма, что, в свою очередь, необходимо для определения механизма патогенеза инфекционного заболевания.

1. Описание математического аппарата

Для выравнивания биологических последовательностей воспользуемся алгоритмом Нидмана–Вунша [1]. Его сложность равна $O(nm)$, где n, m - размеры первой и второй последовательностей соответственно. Это подходит для быстрого вычисления геномов бешенства (их размер около 5000 элементов).

Итак, пусть S_1, S_2 - первая и вторая последовательности, которые необходимо выровнять. Имея две последовательности, необходимо построить матрицу выравнивания M , заполненную весами выравнивания. Для этого введем штраф за разрыв d и матрицу похужести S . В самом простом случае она выглядит так, хотя ее можно настраивать как угодно, главное – чтобы она была симметричной относительно главной диагонали:

$$S = \begin{pmatrix} & a & g & c & t \\ a & 1 & -1 & -1 & -1 \\ g & -1 & 1 & -1 & -1 \\ c & -1 & -1 & 1 & -1 \\ t & -1 & -1 & -1 & 1 \end{pmatrix}$$

Теперь можно приступать к построению матрицы выравнивания:

2. Построим так называемый базис матрицы (то есть заполним первую строку и первый столбец) $M_{i,0} = i * d, M_{0,j} = j * d$.

3. Заполним все остальные элементы по формуле:

$$M_{i,j} = \max(M_{i-1,j-1} + S(A_i, B_j), M_{i,j-1} + d, M_{i-1,j} + d)$$

С помощью такой матрицы и строится выравнивание. Выбираем в последнем столбце ячейку с самым большим весом выравнивания, от нее мы можем перейти вверх (разрыв), влево (вставка) и по диагонали (совпадение или несовпадение), но переходим мы также в ячейку с самым большим весом выравнивания [2]. Так до конца матрицы. В итоге получаются две выровненные последовательности одинаковой длины.

2. Описание алгоритма

Для реализации алгоритма построим три функции: создание матрицы, выравнивание последовательностей и вычисление оценки. Их блок-схемы изображены на рис. 1-рис. 3.

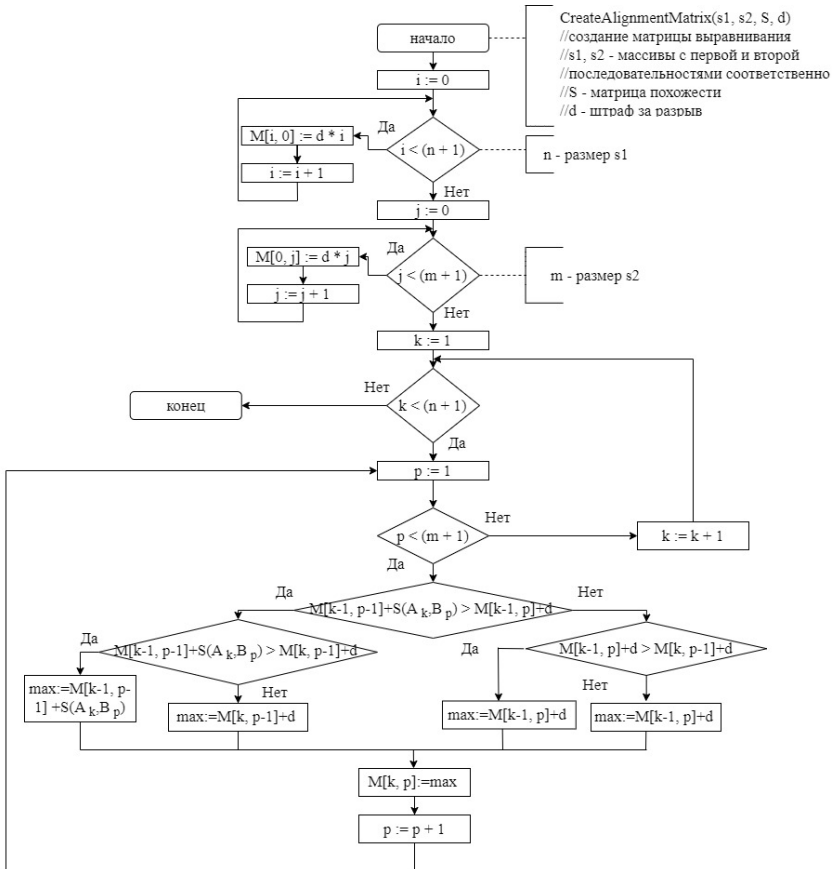


Рис. 1. Блок-схема функции создания матрицы выравнивания

3. Программная реализация алгоритма

На вход программы подаются две последовательности, штраф за разрыв и матрица похожести. При запуске программы открывается форма, представленная на рис. 4.

«Заполнить матрицу поожести», при необходимости пользователь может менять значения в матрице.

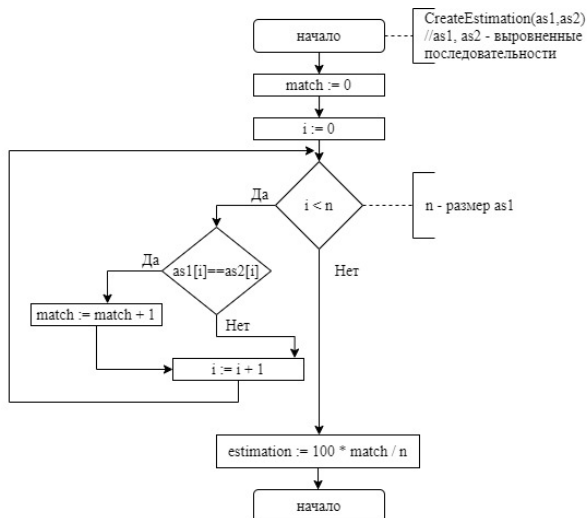


Рис. 3. Блок-схема функции вычисления оценки выравнивания

4. Оптимизация работы с большими последовательностями

Большую последовательность удобнее сравнивать с так называемой эталонной короткой последовательностью, чем с такой же огромной последовательностью, поэтому пусть теперь у нас есть короткая и длинная последовательности.

От каждого элемента длинной последовательности будем откладывать короткую и сравнивать по алгоритму, изображенному на рис. 3. Пройдя так по всей последовательности, получим зависимость оценки выравнивания от индекса элемента.

Затем необходимо выбрать индексы, значение по которым выше заданного порога. После этого необходимо изменить длинную последовательность: остаются все индексы, которые максимально совпадают с короткой последовательностью и по 20% элементов в начале и в конце преобразуемой последовательности. Таким образом, на выходе получается: оценка выравнивания, короткая последовательность и преобразованная длинная.

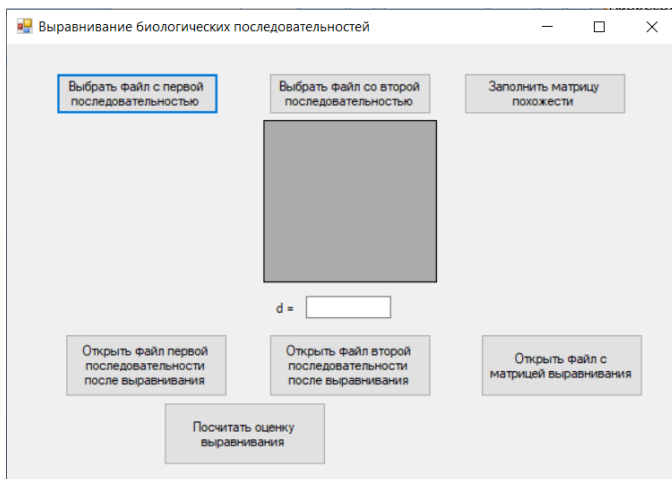


Рис. 4. Вид формы при запуске

5. Результаты

Для проведения эксперимента с целью оценки генетического материала инфекционного носителя и восприимчивого организма были взяты различные биологические последовательности организмов, являющихся переносчиками, резервуарами или возбудителями бешенства. В качестве эталонной биологической последовательности была взята нуклеотидная последовательность гена нуклеопротеида вируса бешенства. В качестве последовательности сравнения был выбран геном собаки, так как данный организм является одним из наиболее часто встречаемых в повседневной жизни переносчиков бешенства.

При программной реализации алгоритма выравнивания была подсчитана оценка выравнивания исследуемых биологических последовательностей, значение которой изображено на рис. 5.

Заключение

В ходе проведенной работы была написана программа, выполняющая алгоритм выравнивания биологических последовательностей, позволяющая обрабатывать большой объем информации. Данная программа позволяет провести оценку родственных связей между организмами, геном которых был использован для проведения исследования.

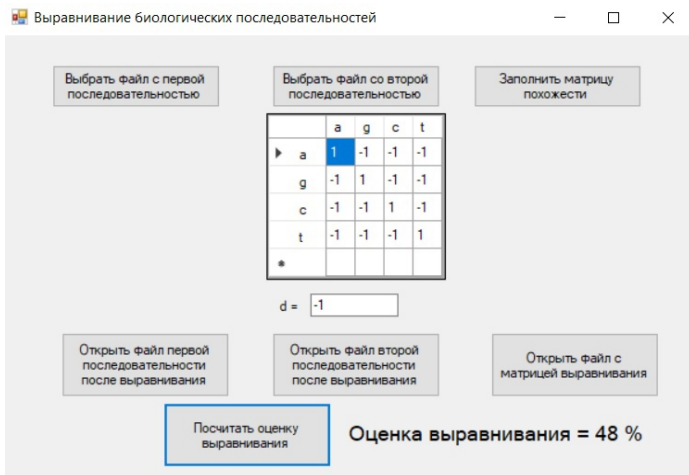


Рис. 5. Результат оценки последовательностей

Важно отметить, что построение программы осуществлялось в рамках совокупности методов Big Data, что позволило решить ряд материальных, аппаратных и этических проблем при исследовании и оценке биологических последовательностей. Такой подход имеет огромное значение при анализе геномов инфекционных заболеваний и восприимчивых к патогенам организмов с целью последующих исследований в области метаболических путей, переселении и циркуляции патогенного микроорганизма в восприимчивом организме. Полученные данные позволяют значительно продвинуться в дальнейшем анализе патогенеза вируса бешенства.

Необходимо также отметить, что выведенные решения на основе информационных технологий в биоинформатике открыли новые возможности анализа и оценки больших массивов биологических данных.

Литература

1. Needleman, Saul B. A general method applicable to the search for similarities in the amino acid sequence of two proteins / Saul B. Needleman, Christian D. Wunsch // Journal of Molecular Biology. – 1970. – Vol. 48. – № 3 – P. 443-453.
2. Sankoff, D. Matching sequences under deletion/insertion constraints / D. Sankoff // Proceedings of the National Academy of Sciences of the United States of America. – 1972. – Vol. 69. – № 1 – P. 4-6.

Автоматизированная система выращивания тканей

О. И. Коротков

Студент бакалавр

Я. А. Туровский

Доцент

Введение

Современные решения в IT-сфере позволяют вывести тканевую инженерию на новый уровень. Благодаря симбиозу кибернетики и медицины, появилась возможность автоматизировать процессы выращивания тканей, что позволяет производить биоматериал в больших количествах с точным поддержанием необходимых для благоприятного размножения клеток параметров среды.

Трансплантация ткани – пересадка здоровой ткани донора на место поврежденных участков ткани реципиента. Долгое время проблемой трансплантации тканей была возможность отторжения реципиентом донорского материала. Но даже в случаях успешного принятия тканей такой подход не решал всех проблем. Пересаженные имплантаты могут устранить только физические и механические недостатки поврежденных тканей, но не восстановить биологические (метаболические) функций, т. е. регенерацию тканей. На помощь в решении данной проблемы пришла тканевая инженерия.

Тканевая инженерия – это процесс искусственного воссоздания тканей на основе клеточного материала. Преимуществом данного подхода является то, что донором клеточного материала может быть сам реципиент, что в значительной степени снижает риск отторжения пересаживаемых тканей. Помимо этого, пересаженные таким образом ткани позволяют восстановить регенерацию поврежденных участков.

Большое количество статей по с данной тематике публикуются в журнале «Вестник трансплантологии и искусственных органов». В нем можно найти полезные и информативные работы [1, 2] на тему тканевой инженерии.

В качестве клеток-предшественников костной ткани используют мезенхимальные стволовые клетки (mesenchymal stem cells, MSC), которые находятся, например, в костном мозге каждого человека. Эти клетки в соответствующих условиях могут дифференцироваться в

клетки разных типов: жировые, костные, хрящевые, мышцы, клетки соединительной ткани [3].

Создание тканеинженерного имплантата (графта) включает несколько этапов:

1. отбор и культивирование собственного или донорского клеточного материала;

2. разработка специального носителя для клеток (матрицы) на основе биосовместимых материалов;

3. нанесение культуры клеток на матрицу и размножение клеток в биореакторе со специальными условиями культивирования;

4. непосредственное внедрение графта в область пораженного органа или предварительное размещение в области, хорошо снабжаемой кровью, для дозревания и формирования микроциркуляции внутри графта (префабрикация) [4].

В данной статье речь пойдет о создании автоматизированной системы выращивания тканей, способной обеспечить благоприятные параметры среды для размножения клеток.

2. Основные компоненты системы

На схеме рис. 1 представлены основные компоненты системы, а также способы их интеграции.

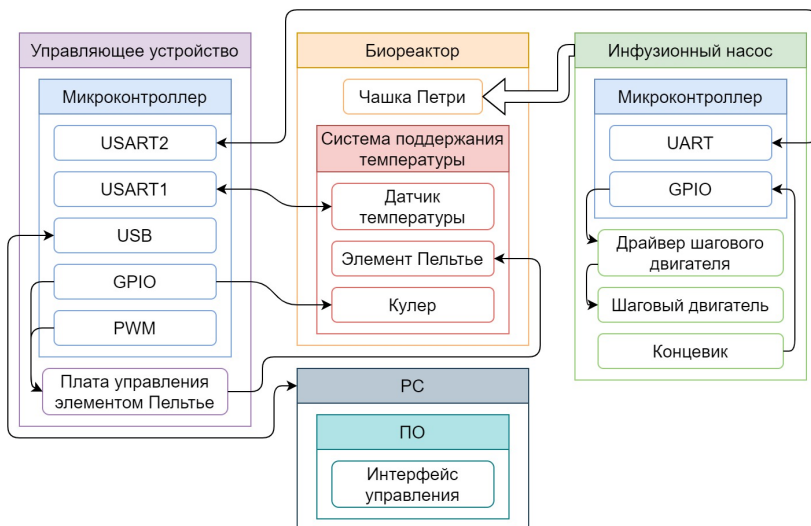
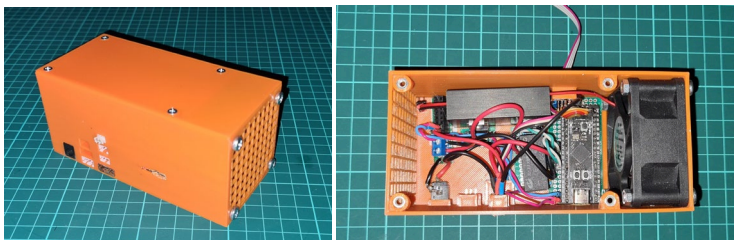


Рис. 1. Схема устройства

3. Управляющее устройство

Центром данного проекта является управляющее устройство (рис. 2) на базе микроконтроллера STM32F401CEU6. Данное устройство отвечает за управление периферией, обработку показаний, полученных с датчиков, взаимодействие с пользовательским интерфейсом.



а

б

а – фотография устройства в собранном виде, б – фотография внутренних компонентов устройства

Рис. 2. Управляющее устройство на базе микроконтроллера STM32F401CEU6

Устройство подключается ПК посредством USB-интерфейса. На компьютере запускается ПО, в котором пользователь задает параметры среды, необходимые для благоприятного протекания химических и биологических процессов. Данные параметры передаются в управляющее устройство, которое запускает процесс поддержания заданных параметров посредством подключённых к нему периферийных устройств. Также управляющее устройство передает в пользовательский интерфейс данные о текущих показателях и сопутствующую информацию.

Микроконтроллер STM32F401CEU6 был выбран за счет наличия обширного множества интерфейсов передачи данных, необходимых для подключения периферийных устройств.

4. Терморегулирование

Одной из основных задач данного проекта является поддержание заданной температуры в зоне биореактора. Для выполнения этой задачи необходимы датчик температуры для получения сведений о текущей температуре, термоэлектрический преобразователь, обеспечивающий управление температурой в биореакторе, а также контроллер, управляющий термоэлектрическим преобразователем на основе сведений о текущей температуре.

Для измерения текущей температуры использован популярный цифровой датчик температуры DS18B20 компании Dallas. Данный датчик имеет невысокую стоимость, но при этом обеспечивает довольно хорошую точность измерений (погрешность $\pm 0,5^{\circ}\text{C}$) и широкий диапазон рабочих температур (от -55°C до 125°C).

Передача данных между датчиком и микроконтроллером происходит по протоколу 1-Wire с использованием однопроводной двунаправленной шины. Однако в данном проекте для подключения датчика был задействован интерфейс USART микроконтроллера, использующий для подключения две линии: TX и RX (см. рис. 3).

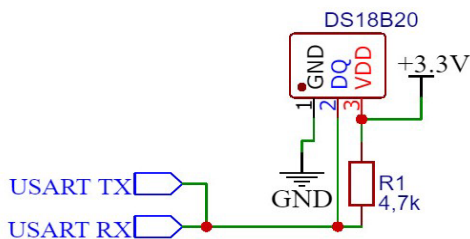


Рис. 3. Схема подключения датчика DS18B20 к микроконтроллеру

Такой подход к передаче информации дает возможность задействовать аппаратные средства микроконтроллера, что позволяет не тратить процессорное время на передачу данных между микроконтроллером и датчиком. С учетом того, что рассматриваемый процесс имеет постоянный циклический характер, данное решение позволяет значительно оптимизировать процедуру получения текущей температуры.

В качестве термоэлектрического преобразователя был выбран элемент Пельтье. Данный элемент основан на эффекте Пельтье и представляет собой набор полупроводниковых материалов, которые при пропускании через себя тока переносят тепловую энергию от одного контакта к другому, тем самым нагревая одну сторону элемента и охлаждая другую. Меняя полярность протекающего через элемент тока, можно менять нагреваемую и охлаждаемую стороны. Достоинствами данного элемента являются малые габариты, а также возможность как нагревать, так и охлаждать необходимые объекты, что позволяет поддерживать необходимую температуру независимо от температуры окружающей среды. В летнее время, когда комнатная температура выше температуры, необходимой для благоприятного

протекания процессов, можно охлаждать рабочую зону биореактора, а в зимнее время, когда температура окружающей среды ниже необходимой, — нагревать его. Для переключения полярности питания элемента Пельтье собрана схема (см. рис. 4) с использованием реле с двумя переключающимися группами контактов.

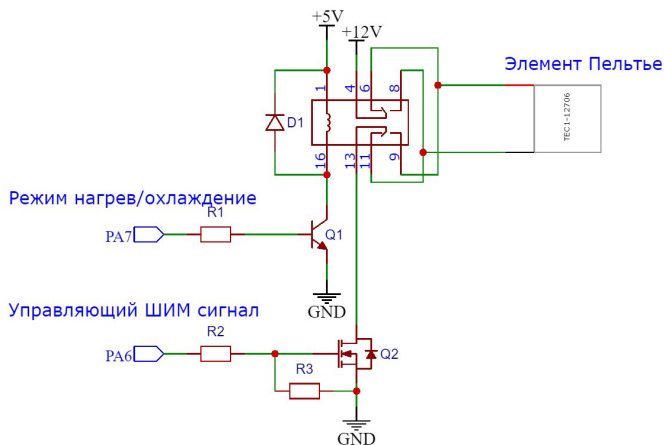


Рис. 4. Схема подключения элемента Пельтье к микроконтроллеру

Для обеспечения хорошего охлаждения от обратной нагреваемой стороны необходимо отводить выработанное тепло. Для этих целей смоделирован и собран воздушный охладитель (см. рис. 5).

Так как данная система, в силу большой теплоемкости, обладает высокой тепловой инерцией, управление элементом Пельтье осуществляется при помощи подачи на него ШИМ-сигнала высокой частоты. Это позволяет варьировать мощность нагрева/охлаждения элемента путем изменения скважности ШИМ-сигнала. Благодаря этому при выставлении заданной температуры мы можем снижать мощность пропорционально приближению к заданной отметке, тем самым избежав перескакивания через заданное значение, а также в процессе поддержания заданной температуры выставлять необходимую постоянную мощность, позволяющую установить равновесие тепловой системы и избежать возникновения автоколебаний.

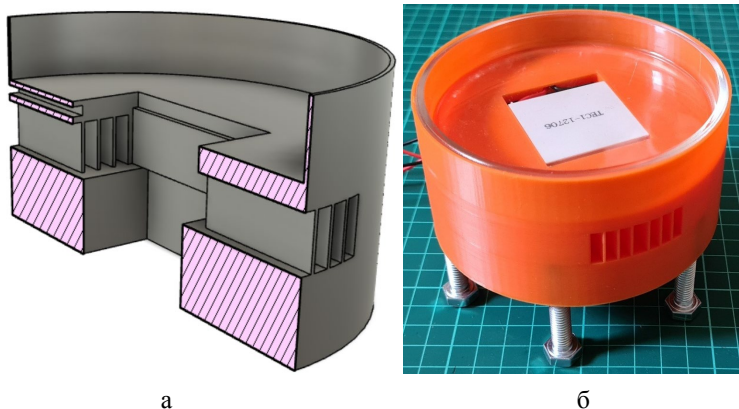


Рис. 5. Системы поддержания температуры под чашкой Петри на основе элемента Пельтье с воздушным охлаждением: а – модель в разрезе, б – внешний вид

За управление скважностью ШИМ-сигнала отвечает пропорционально-интегрально-дифференцирующий регулятор (далее ПИД-регулятор). Значение текущей температуры, полученное с датчика, передаётся в ПИД-регулятор, где на его основе, а также на основе значения целевой температуры по представленной ниже формуле вычисляется скважность ШИМ-сигнала, подаваемого на элемент Пельтье:

$$u(t) = P + I + D = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de(t)}{dt}, \quad (1)$$

где P , I , D — пропорциональная, интегральная и дифференциальная составляющие, соответственно, $e(t)$ — текущая ошибка, вычисляемая как разность текущей и заданной температуры, K_p , K_i , K_d — пропорциональный, интегральный, дифференциальный коэффициенты, соответственно. Данные коэффициенты влияют на степень воздействия соответствующей составляющей регулятора и подбираются экспериментальным путём для достижения наилучшей работы регулятора.

5. Поддержание ионного состава

Не менее важной задачей является поддержание ионного состава окружающей ткани жидкости.

Регулирование ионного состава в данной системе осуществляется путем изменения концентрации раствора с заранее известными параметрами в общем объеме окружающей ткани жидкости.

Для этих целей сконструирован инфузионный насос (см. рис. 6), способный вводить и откачивать жидкости в заданном объеме с заданной скоростью.

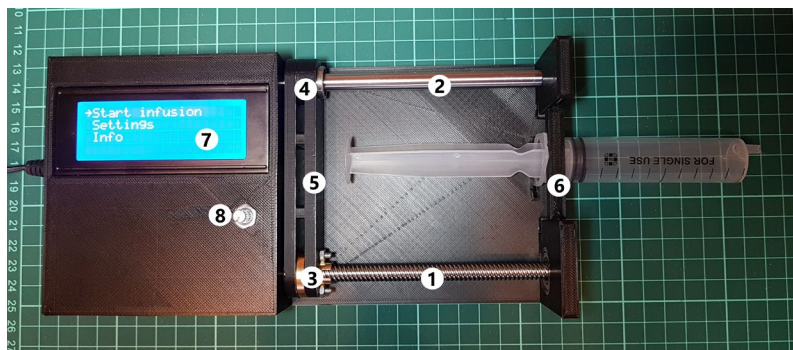


Рис. 6. Инфузионный насос

В качестве механизма, выполняющего нажатие поршня шприца, был выбран шаговый двигатель, так как он может обеспечить точное и плавное вращение с заданной скоростью. Для преобразования вращательного движения в поступательное был использован ходовой винт (элемент 1, рис. 6) с перемещающейся по нему гайкой ходового винта (элемент 3, рис. 6), соединенной подвижной кареткой (элемент 5, рис. 6) с направляющим валом (элемент 2, рис. 6). Для фиксации шприца используется пружинный фиксатор (элемент 6, рис. 6).

Для введения нужного количества жидкости вычисляется необходимое расстояние перемещения каретки по известным заранее параметрам установленного в данный момент шприца. Далее на основе данного расстояния находится количество шагов двигателя, которые необходимо совершить для перемещения каретки на заданное расстояние. Зная, что при совершении одного шага двигатель вращает ходовой винт на угол, равный $0,1125^\circ$ ($1,8^\circ$ — собственный шаг двигателя, 16 — делитель драйвера, управляющего шаговым двигателем), вычисляем, что полный оборот в 360° двигатель делает за 3200 шагов. Исходя из того, что шаг резьбы используемого в данном проекте ходового винта равен 8 мм, получаем расстояние продольного перемещения каретки при совершении одного шага двигателя, равное 0,0025 мм. В силу приведенных выше вычислений имеем формулу:

$$n = \frac{L}{0,0025}, \quad (2)$$

где n — необходимое количество шагов двигателя для перемещения каретки на расстояние L .

Получив искомое количество шагов и зная заданную скорость введения раствора (переведенную из мл/ч в мм/с по составленной заранее таблице, основанной на параметрах шприцов), вычисляем время одного шага путем деления количества шагов на скорость введения раствора. После чего настраиваем прерывание таймера на вычисленное время, в течение которого совершаем шаг двигателя.

Заключение

На данный момент реализован механизм для поддержания заданной температуры биореактора в пределах от 0°C до 50°C, а также возможность ручного управления инфузионными насосом для поддержания ионного состава.

Ведется работа по внедрению в проект анализатора ионного состава, который позволит автоматизировать процесс управления инфузионными насосом.

Список литературы

1. Создание и биологические испытания новых матрицков из биорезорбируемых материалов / В. Н. Василец [и др.] // Вестник трансплантологии и искусственных органов. – 2010. – Т. 12, №2. – С. 67–73.
2. Севастьянов, В. И. Технологии тканевой инженерии и регенеративной медицины / В. И. Севастьянов // Вестник трансплантологии и искусственных органов. – 2014. – Т. 16 №3. – С. 93–108.
3. Трусов, Л. А. Нанотехнологии в тканевой инженерии [Электронный ресурс] / Л. А. Трусов ; Нанометр. Нанотехнологическое сообщество. – Режим доступа: www.nanometer.ru/2007/10/16/tkanevaa_inzheneria_4860.html
4. Народицкий, Б. С. Тканевая инженерия / Б. С. Народицкий, Л. Н. Нестеренко // Словарь нанотехнологических и связанных с нанотехнологиями терминов [Электронный ресурс]. – Режим доступа : <https://thesaurus.rusnano.com/wiki/article1799>

Формирование отчета деятельности наркоконтроля по Воронежской области средствами офисных приложений

А. Н. Кропачев

Студент бакалавр

Е. А. Копытина

Старший преподаватель

Введение

Для успешной работы правоохранительных органов необходимо не только обеспечивать порядок и спокойствие граждан, но также и вести отчетность о проделанной работе. Именно поэтому так важно обеспечить удобство работы с БД и средствами составления отчетов. Пакет офисных приложений «Microsoft Office» позволяет полноценно вести работу, связанную с организацией БД, и делать все необходимые отчеты, соответствующие требованиям, как показано в [1-3].

В данной статье рассматривается разработка способа формирования отчетов деятельности наркоконтроля по Воронежской области средствами офисных приложений.

1. Постановка задачи

К описываемому в статье отчету по деятельности наркоконтроля по Воронежской области были выдвинуты следующие требования:

- Удобство работы с формой отчета.
- Расположение требуемых параметров в строгом порядке.
- Возможность быстрого редактирования отчета по параметрам.
- Добавление в отчет данных из других таблиц.

2. Реализация логики

Реализуемый отчет формируется средствами офисных приложений и имеющейся БД. Для доступа к БД и возможности формирования отчета необходимо использовать пароль и профиль администратора. После осуществления входа в приложение, открывается окно главной формы, на котором предоставлены несколько стандартизированных отчетов. Каждый такой отчет формирует SQL-запрос к БД. Выбрав необходимый отчет, необходимо нажать на кнопку «Открыть отчетную форму». Обработчик событий отправит заранее сформированный SQL-

запрос к БД, которая вернет требуемые данные. Эти данные будут расположены на форме в определенном порядке. Таким образом создается пользовательская форма отчета. Для формирования отчета соблюдая все требования, необходимо использовать четыре таблицы, включенных в базу данных: «Медучреждение», «Лица на учете», «Адрес», «Места отбывания наказания» (рис. 1).



Рис. 1. ER-диаграмма

Таким образом, используя четыре выбранных таблицы, можно построить запрос, который будет содержать всю требуемую информацию о лице на учете, медучреждении, адресе или месте отбывания наказания. Выбранную информацию необходимо сгруппировать таким образом, чтобы она отображала все требования, предоставленные организацией. Средство реализации Microsoft Access позволяет сохранить именованные SQL-запросы так, чтобы данные, полученные этим запросом, можно было бы использовать как новую таблицу. Примеры поименованных запросов в исследуемой базе данных: «qryUklonisty», «qryLicoPostanovleniaCnt» и «qryPosledneePostanovlenie». Используя выбранные таблицы БД и созданные поименованные SQL-запросы, создаем форму отчета с требуемыми параметрами. За основу формы возьмем поименованный SQL-запрос «qryLicaNaUcheteAll». Данный запрос до реорганизации формирует отчет по таблице «Лица на учете». Кроме того, что набор этих полей не соответствует требуемому, поля набора расположены в неправильном порядке.

Для построения формы отчета используем режим конструктора Microsoft Access, который предоставляет графический интерфейс для

создания SQL-запросов и дает возможность конфигурации отображаемых полей в отчете. Реорганизовать данный запрос необходимо таким образом, чтобы набор полей запроса включал поля не только таблицы «Лица на учете», но и поля таблиц «Медучреждение», «Адрес» и «Места отбывания наказания».

Для соответствия требованиям организации, необходимо подключить дополнительно три поименованных SQL-запроса: «qryPosledneePostanovlenie», «qryLicoPostanovleniaCnt», «qryUklonisty». Информация, возвращаемая этими запросами, является достаточной для формирования отчета с соблюдением всех требований к набору полей, которые предоставлены организацией (рис. 2).

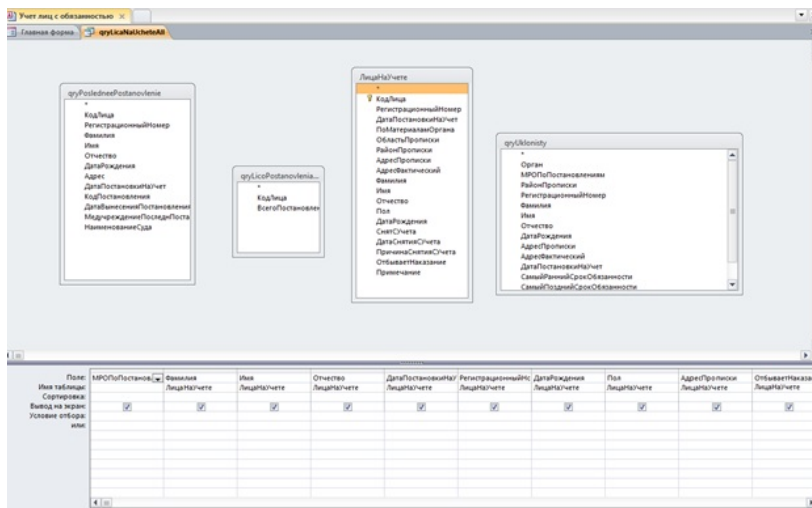


Рис. 2. Реорганизованный запрос «qryLicaNaUcheteAll»

Связь с запросом «qryPosledneePostanovlenie» организуем по полю «КодЛица». Данное поле является первичным ключом для главной таблицы «Лица на учете» и для самого поименованного SQL-запроса. В данном запросе «qryPosledneePostanovlenie» хранится информация последнего зарегистрированного постановления, а значит самая актуальная информация касательно определенного человека.

Благодаря информации, собранной этим запросом, можно узнать наименование медучреждения и дату постановления. Данные поля необходимо использовать в конечной форме отчета в соответствии с требованиями к набору полей, предоставленными организацией.

Связь с запросом «qryLicoPostanovleniaCnt» также организуем по полю «КодЛица». Данное поле, как и в предыдущем случае, является первичным ключом для главной таблицы «Лица на учете» и для самого поименованного SQL-запроса. В данном запросе «qryLicoPostanovleniaCnt» возвращается количество всех записей, удовлетворяющих SQL-запросу.

Благодаря информации, собранной этим запросом, можно узнать количество записей в отчете, который будет сформирован в конечном итоге. Таким образом, выполняется еще одно требование организации в построении отчета.

С SQL-запросом «qryUklonisty» связь организуется по внешнему ключу, который представлен полем таблицы «РегистрационныйНомер». Для того, чтобы была возможность выбрать данные об «уклонистах», необходимо организовать связь рассматриваемого запроса не с главной таблицей «Лица на учете», а с запросом «qryPosledneePostanovlenie». Именно этот запрос дает необходимую информацию для того, чтобы точно определить «уклониста». Так как в поименованном SQL-запросе «qryPosledneePostanovlenie» хранится информация последнего зарегистрированного постановления, а значит самая актуальная информация касательно определенного человека, то связь запроса «qryUklonisty» необходимо организовывать именно через запрос «qryPosledneePostanovlenie».

Благодаря информации, собранной этим запросом, можно узнать место отбывания наказания и примечание, оставленное в постановлении «уклониста». Эти поля завершают требуемый набор полей к отчету.

Для формирования отчета выберем требуемые поля и расположим их в строгом порядке. Таким образом, SQL-запрос «qryLicaNaUcheteAll» будет соответствовать всем предоставленным требованиям.

3. Реализация интерфейса

Выбрав на главной форме пункт формирования отчета, будет вызван SQL-запрос «qryLicaNaUcheteAll», который сформирует отчет, соответствующий всем предоставленным требованиям.

Отчет хранит данные из нескольких таблиц, поля отчета расположены в строгом порядке, имеется возможность быстрого редактирования полей, предоставляет самые полные данные об объекте рассмотрения, удобен в использовании (рис. 3).

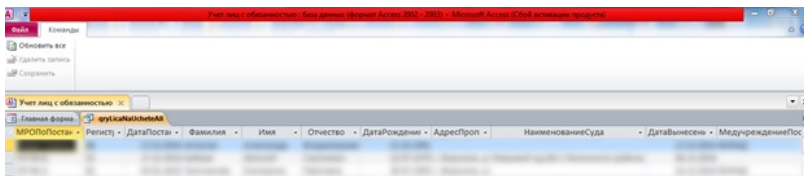


Рис. 3. Итоговый вид сформированного отчета

Заключение

В данной статье было рассмотрено формирование отчета, который предоставляет информацию столь необходимую для наркоконтроля по Воронежской области в строго упорядоченном виде. Что позволяет обрабатывать, хранить и использовать эту информацию эффективнее. Кроме того, изучая отчетность, можно обнаружить проблемы и ошибки в работе организации. Отчет, сформированный по всем требованиям, – мощный инструмент для помощи в обеспечении порядка и спокойствия граждан правоохранительными органами.

Список литературы

1. Подольский, К. Д. Формирование предварительного коммерческого предложения ООО «ЧерноземАгроماش» средствами MS Excel / К. Д. Подольский, Я. Э. Головин, Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ. — Воронеж, 2020.— Вып. 14, ч. 1: Научные работы студентов бакалавриата. - С. 143-147.
2. Копытина, Е.А. Оптимизация стоимости доставки ресурсов при строительстве инженерных технологий / Е. А. Копытина, Н. А. Петрикеева // ВІМ. Проектирование. Строительство. Эксплуатация: материалы Всероссийского форума, Воронеж, 15 ноября 2018 г. — Воронеж, 2018. — С. 51-55.
3. Копытина, Е.А. Определение тенденции развития строительной организации на основе прогнозирования временных рядов / Е. А. Копытина, Н. А. Петрикеева, Д. М. Чудинов // Информационные технологии в строительных, социальных и экономических системах. — Воронеж, 2020. — № 1(19). - С. 87-91.

О нильпотентных алгебрах Ли с «избыточными» симметриями

М. А. Куприн

Студент бакалавр

А. В. Лобода

Профессор

Введение

Как известно (см. [1]), всякая нильпотентная 7-мерная вещественная алгебра Ли содержит (хотя бы один) 4-мерный абелев идеал. В полном списке из 149 вещественных, неразложимых 7-мерных нильпотентных алгебр Ли (см. [1]) содержится 33 алгебры, у которых такой идеал единственный. Из [2] следует, что орбитами этих алгебр в комплексном пространстве \mathbb{C}^4 могут быть только вырожденные по Леви гиперповерхности.

В данной работе мы будем рассматривать блок алгебр из [1], имеющих ровно два абелевых 4-мерных идеала (случай «избыточных» симметрий). Таких алгебр в полном списке содержится 18, и у некоторых из них имеются невырожденные по Леви орбиты в четырехмерном комплексном пространстве.

Целью настоящей работы является описание всех невырожденных орбит в \mathbb{C}^4 для 7-мерных нильпотентных алгебр Ли, имеющих «избыточные» симметрии.

1. Поиск реализаций алгебр Ли в виде алгебр векторных полей

Запишем каждый элемент базиса e_1, \dots, e_7 обсуждаемой алгебры Ли \mathfrak{g} в виде голоморфного векторного поля в пространстве \mathbb{C}^4 :

$$e_k = a_k(z) \frac{\partial}{\partial z_1} + b_k(z) \frac{\partial}{\partial z_2} + c_k(z) \frac{\partial}{\partial z_3} + d_k(z) \frac{\partial}{\partial z_4} \quad (1)$$

В этой записи $a_k(z), b_k(z), c_k(z), d_k(z)$ – голоморфные (вблизи обсуждаемой точки поверхности) функциональные коэффициенты, $z = (z_1, z_2, z_3, z_4)$ – вектор комплексных координат. Мы также

используем записи вида $e_k = (a_k, b_k, c_k, d_k)$ для сокращения формулы (1).

Для более удобных рассмотрений исследуемый блок разбивается на подблоки, в каждом из которых используется сходство коммутационных соотношений.

Ниже приведена табл. 1, содержащая один из подблоков, в котором каждая алгебра имеет два одинаковых абелева идеала вида: $I_4 = \langle e_4, e_5, e_6, e_7 \rangle$ и $I_4' = \langle e_3, e_5, e_6, e_7 \rangle$; s_{ij} обозначает коммутационное соотношение вида $[e_i, e_j]$.

Таблица 1

Список алгебр одного из подблоков

Алгебры	s_{12}	s_{13}	s_{14}	s_{23}	s_{24}	s_{26}	s_{34}
2357 C	$-e_4$	e_6	e_5		e_6	e_7	$-e_7$
2357 D	$-e_4$	e_6	e_5	e_5	e_6	e_7	e_7
2357 D ₁	$-e_4$	e_6	$-e_5$	e_5	e_6	e_7	e_7

Благодаря наличию схожих коммутационных соотношений и совпадающим абелевым идеалам, становится возможным существенное сокращение количества рассмотрений.

Для поиска реализаций алгебр Ли в виде алгебры векторных полей в \mathbb{C}^4 необходимо использовать Лемму из [3], согласно которой базис абелева идеала I_4 7-мерной алгебры Ли можно упростить до одного из трех видов:

$$\begin{array}{lll}
 1) & 2) & 3) \\
 e_4 : (1, 0, 0, 0), & (0, b_4(z_1), c_4(z_1), d_4(z_1)), & (0, 1, 0, 0), \\
 e_5 : (0, 1, 0, 0), & (0, 1, 0, 0), & (0, 0, c_5(z_1), d_5(z_1)), \\
 e_6 : (0, 0, 1, 0), & (0, 0, 1, 0), & (0, 0, 1, 0), \\
 e_7 : (0, 0, 0, 1), & (0, 0, 0, 1), & (0, 0, 0, 1).
 \end{array}$$

Применим первый случай для алгебр из табл. 1. Рассмотрим коммутаторы полей e_1 и e_3 с выпрямленными полями из $I_4 = \langle e_4, e_5, e_6, e_7 \rangle$. Эти поля примут вид:

$$e_1 = (A_1, -z_1 + B_2, C_1, D_1),$$

$$e_3 = (A_3, B_3, C_3, -z_1 + B_3).$$

Тогда $[e_1, e_3] = (0, A_3, 0, -A_1)$, что противоречит условию:

$$[e_1, e_3] = e_6 = (0, 0, 1, 0).$$

Похожие противоречия возникают во втором и третьем случае Леммы из [3].

Сожим образом рассматриваются остальные алгебры из исследуемого блока.

Предложение 1. Только 4 из всех 18 алгебр Ли с «избыточными» симметриями имеют реализации в виде алгебр векторных полей в \mathbb{C}^4 .

Эти алгебры представлены в табл. 2.

Таблица 2

Список алгебр одного из подблоков

Алгебры	s_{12}	s_{13}	s_{14}	s_{15}	s_{23}	s_{24}	s_{25}	s_{26}	s_{34}
137 В		e_5		e_7		e_6		e_7	$-e_7$
1357 Н	e_4	e_5		e_7		e_6	e_7	e_7	e_7
147 A_1		$-e_5$		e_7	$-e_6$			e_7	e_7
137 B_1		e_5	e_6	e_7	$-e_6$	e_5		e_7	e_7

Пример реализации алгебры 1357Н для 1-го случая Леммы из [3]:

$$e_1 = (1, 0, 0, 0), e_2 = (A_2, z_1 + B_2, -z_2 + C_2, -z_3 + D_2),$$

$$e_3 = (0, 1, 0, 0), e_4 = (0, B_4, C_6 z_1 + C_4, -z_2 - \frac{1}{2} z_1^2 - z_1 D_6 + D_4),$$

$$e_5 = (0, 0, 1, 0), e_6 = (0, 0, C_6, -z_1 + D_6), e_7 = (0, 0, 0, 1).$$

Здесь A_k, B_k, C_k, D_k – комплексные коэффициенты. При этом имеется два ограничения на коэффициенты:

$$C_6 - A_2 = 1; -A_2 D_6 - B_2 + C_4 = 0. \quad (2)$$

Несложно увидеть, что (при любых значениях свободных коэффициентов) для такой алгебры векторных полей выполняются исходные коммутационные соотношения для алгебры 1357 Н.

2. Интегрирование алгебр Ли

Вещественная гиперповерхность $M = \{\Phi = 0\}$ является орбитой (или интегральной поверхностью) голоморфной реализации алгебры g , если для каждого базисного поля e_k этой алгебры выполняется условие касания поверхности M в виде:

$$\text{Re}(e_k(\Phi)|M) = 0. \quad (3)$$

Расписывая условие (3) для семи базисных векторных полей любой алгебры, мы получаем систему уравнений в частных производных, которую последовательно решаем.

Для алгебры 1357 Н такая система примет вид:

$$\begin{aligned}
 1) \frac{\partial F}{\partial x_1} = 0; \quad 3) \frac{\partial F}{\partial x_2} = 0; \quad 5) \frac{\partial F}{\partial x_3} = 0; \quad 7) \frac{\partial F}{\partial x_4} = 0; \\
 2) (y_3 - D_{22} + A_{22}) \frac{\partial F}{\partial y_1} + (y_1 + B_{22}) \frac{\partial F}{\partial y_2} + (-y_2 + C_{22}) \frac{\partial F}{\partial y_3} = 0; \\
 4) x_1 y_1 + y_2 - D_{42} + B_{42} \frac{\partial F}{\partial y_2} + (C_{61} y_1 + C_{62} x_1 + C_{42}) \frac{\partial F}{\partial y_3} = 0; \\
 6) y_1 - D_{62} + C_{62} \frac{\partial F}{\partial y_3} = 0.
 \end{aligned}$$

Из уравнений 1), 3), 5), 7) можно сделать вывод, что F не зависит от переменных x_1, x_2, x_3, x_4 . Таким образом $F = -y_4 + F(y_1, y_2, y_3)$.

Из уравнения 6) можно найти $F(y_1, y_2, y_3)$:

$$F(y_1, y_2, y_3) = \frac{-(y_1 + D_{62})y_3}{C_{62}} + F_1(y_1, y_2) \quad (4)$$

Подставив значение (4) в остальные уравнения, найдем из уравнения 4) функцию $F_1(y_1, y_2)$:

$$F_1(y_1, y_2) = Ay_1 y_2 + By_2 + Cy_2^2 + Dy_1^2 y_2 + F_2(y_1), \quad (5)$$

где A, B, C, D – функции, зависящие от коэффициентов A_k, B_k, C_k, D_k .

После подстановки значения (5) в уравнение 2) мы сможем выразить $F_2(y_1)$:

$$F_2(y_1) = Ey_1 + Gy_1^2 + Hy_1^3 - Ky_1^4 + C, \quad (6)$$

где E, G, H, K – функции, зависящие от коэффициентов A_k, B_k, C_k, D_k .

Таким образом, подставив (5) и (6) в (4), мы получим искомую функцию $F(y_1, y_2, y_3)$. Применив ограничения (2) и используя различные замены переменных, получим орбиту вида: $\text{Im } z_4 = z_1 \bar{z}_2 + z_2 \bar{z}_1 + |z_3|^2 - |z_1|^4$.

Аналогичным методом рассматриваются и интегрируются остальные возможные реализации алгебр Ли из изучаемого блока.

Предложение 2. Невырожденными орбитами четырех алгебр из предложения 1 являются с точностью до голоморфных преобразований лишь следующие повержности:

$$137 A_1: \operatorname{Im} z_4 = |z_1|^2 + |z_2|^2 - |z_3|^2,$$

$$137 B: \operatorname{Im} z_4 = z_1 \bar{z}_2 + z_2 \bar{z}_1 + |z_3|^2 + |z_1|^4,$$

$$137 B_1: \operatorname{Im} z_4 = |z_1|^2 + |z_2|^2 + |z_3|^2,$$

$$1357 H: \operatorname{Im} z_4 = z_1 \bar{z}_2 + z_2 \bar{z}_1 + |z_3|^2 - |z_1|^4.$$

3. Заключение

В отличие от 33 нильпотентных 7-мерных алгебр Ли, имеющих единственный 4-мерный абелев идеал и не допускающих невырожденных орбит в комплексном пространстве \mathbb{C}^4 , показано, что в блоке из 18 алгебр Ли с «избыточными» симметриями имеются невырожденные орбиты. Имеется всего 4 алгебры с такими орбитами. Все эти невырожденные орбиты описаны явными уравнениями.

Список литературы

1. Gong, M.-P. Classification of Nilpotent Lie Algebras of Dimension 7 (Over Algebraically Closed Fields and \mathbb{R}) / M.-P. Gong. – University of Waterloo, 1998. – 165 p.
2. Крутских, В. В. О 7-мерных алгебрах Ли с единственным 4-мерным абелевым идеалом / В. В. Крутских, А. В. Лобода // Материалы Международной конференции «Воронежская весенняя математическая школа-2021». – Воронеж, 2021. – С. 150–151.
3. Loboda A. V. On the Orbits of Nilpotent 7- dimensional Lie Algebras in 4-dimensional Complex Space / A. V. Loboda, R. S. Akopyan, V. V. Krutskikh // Журнал. Сиб. Фед. Университета. Математика. Физика. – 2020. – 13(3). – С. 360–372.

Разработка приложения для получения данных с 4D-рига

Я. В. Лазарев

Студент бакалавр

Д. И. Соломатин

Старший преподаватель

Введение

Современные стандарты индустрии кино и видеоигр требуют тщательности и точности в воспроизведении внешности настоящих людей. Важное место в сфере компьютерной графики занимает создание правдоподобных 3D-моделей человеческого лица, особое внимание уделяется мелким деталям. правдоподобного отображения лиц используют цифровых дублеров реальных актеров. В настоящее время существует несколько способов создания цифрового дублера.

Первый способ – создание 3D художником лицевого скелета, который может относительно точно передать мимику лица реального актера. Для создания анимационной сессии с использованием этого скелета, необходимо установить на голове актера шлем с камерой для захвата лица, а также провести аннотирование лица маркерами. Впоследствии лицевой скелет старается повторить движения маркеров с видеопотока. На выходе получается анимационная сессия для данного актера, которую можно использовать в производстве.

Другой способ получения данных для создания цифровых дублеров заключается в следующем. Актера помещают в фотограмметрическую установку (риг), состоящую из большого количества видеокамер и элементов освещения, и снимают необходимую сцену. На выходе получают данные 4D-секвенции в виде набора кадров или видео, которые после преобразования отдельными программами становятся серией высокополигональных 3D-моделей (сканов).

При использовании последнего способа появляется необходимость в ПО, позволяющем получать данные с 4D-ригов. Данная работа посвящена разработке такого решения.

1. Устройство 4D-рига и постановка задачи

Фотограмметрическая установка, или 4D-риг, представляет собой конструкцию, на которой закреплены камеры и элементы освещения. Камеры расположены таким образом, чтобы захватить человеческое лицо со всех ракурсов. Это необходимо для точного построения 3D-модели лица. Пример 4D-рига показан на рис. 1 ниже.

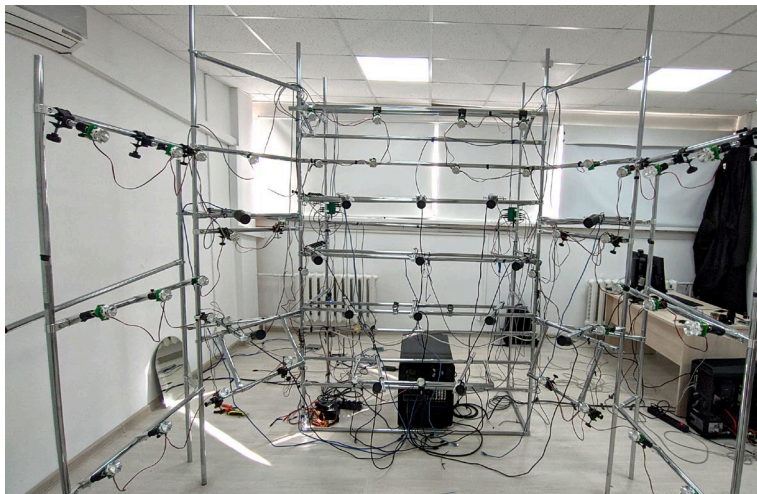


Рис. 1. Пример фотограмметрической установки

В данной установке используются камеры машинного зрения Basler A2040-120um, способные выдавать кадры с разрешением в 2048×1536 пикселей в монохромном формате, то есть полученные кадры имеют цветовую гамму в градациях серого.

Такие камеры снимают изображения в высоком разрешении и отправляют их несжатыми на компьютер, в хорошем качестве и с высокой детализацией. Камеры машинного зрения обычно используются в автоматизации производства или в медицинской сфере, где необходимо видеть мелкие детали.

Помимо камер есть и светодиодные вспышки, которые могут сфокусировать большое количество света в небольшом диаметре. Это позволяет равномерно осветить лицо актера, чтобы все мелкие детали были хорошо видны. Стоит отметить, что вспышки не светят все время, а лишь в те моменты, когда камеры производят захват кадра.

Для того, чтобы работа этих двух компонентов была слаженной, используют специальные устройства – синхронизаторы. Они посылают

периодические электроимпульсные сигналы, которые могут принимать как вспышки, так и камеры по специальным портам. Кроме того, решается проблема синхронной съемки со всех камер. Устройства синхронизации могут изменяться, так как установка все время модифицируется и дорабатывается.

Съемка осуществляется с частотой 60 кадров в секунду. Этого достаточно для того, чтобы переходы между кадрами были плавными. Размер кадров, получаемых с камер, равен 3 мегабайта (2048×1536 пикселей и 1 бит на пиксель). Всего в установке используется 15 камер, а значит поток данных, получаемый с установки каждую секунду, равен $3 \text{ Мб} \times 60 \text{ fps} \times 15 = 2700 \text{ Мб}$.

Для обработки потока данных с рига используется ПК со следующими характеристиками:

- Процессор Intel Core i9 9920X (3.5 ГГц, 12 ядер, 24 потока).
- Размер ОЗУ: 64 Гб.
- 3 SSD накопителя Intel Optane 500 Гб и 2 SSD Samsung 980 Pro 512 Гб.
- Материнская плата ASUS (7 разъемов PCI-e x16).
- Видеокарта Nvidia GeForce GTX 1080 Ti.

Для подключения камер к ПК используются кабели USB, 5 встроенных и 10 дополнительных портов USB 3.1, обеспечиваемые USB-контроллерами, работающими через интерфейс PCI-e.

Итак, задача состоит в том, чтобы разработать приложение, отвечающее следующим требованиям:

- Возможность работать с большими потоками данных.
- Поддержка кастомных средств синхронизации.
- Возможность масштабирования системы, конвейеров обработки.

2. Обработка данных и исследование мощности ПК

Работа с камерами Basler осуществляется с помощью Pylon API. С помощью него можно производить настройку камер, а также контролировать выделение памяти для буферов кадров. После получения данных с рига, они проходят через конвейер обработки, который позволяет сжать кадр, отправить его на отображение в графический интерфейс, записать на устройство долговременного хранения.

Как уже говорилось ранее, камеры машинного зрения выдают несжатые кадры. Имея 5 накопителей с емкостью 500 Гб каждый и распределение 3 камеры на один SSD, возможно записать сцены с суммарной продолжительностью около 15 минут, после чего свободного пространства на дисках не останется. Чтобы увеличить время съемки,

существует два способа: либо увеличивать объем и/или количество дисков, либо уменьшать размер кадров путем их сжатия. В целом, можно применять оба подхода одновременно, что даст еще больший выигрыш.

Было принято решение исследовать способы сжатия кадров и оценить, насколько выгодно это использовать. Также стоит отметить, что сжатие кадров должно происходить с минимальными потерями информации, иначе построенные сканы будут иметь плохое качество. Минимальное сжатие, при котором качество получаемых сканом было хорошим, сохраняло 96% информации. В среднем такое сжатие позволяло уменьшить размер кадра в 2-3 раза.

Для начала использовалось сжатие с использованием ресурсов CPU при помощи библиотеки `turbojpeg`. Однако тесты показали, что ресурсов имеющегося процессора недостаточно, чтобы успевать обрабатывать поток данных с 15 камер при частоте съемки 60 кадров в секунду.

Далее было проверено сжатие на видеокарте, поддерживающей технологию CUDA. Для этого было использовано специальное API, предоставляющее набор функций для работы с сжатием изображений. Тесты показали, что GPU способен справляться с вбдным потоком в 900 кадров, и в зависимости от коэффициента сжатия, использование вычислительных ресурсов видеокарты варьировалось от 40% до 80%.

Использование сжатия на GPU позволило не только успешно обрабатывать большой поток данных, но и сэкономить немалое количество ресурсов процессора.

Запись кадров на накопители производилась в режиме прямого доступа к памяти (Direct Memory Access) [1]. В этом режиме скорость записи достигает максимально возможных значений, и CPU практически не участвует в этом процессе. Для того, чтобы использовать этот режим, под кадры необходимо выделять выровненную по адресам память, размер которой кратен 512 байтам. Так, тесты SSD Intel Optane показали максимальную скорость записи около 1710 Мб/с.

3. Реализация архитектуры и построение конвейеров

Для данной задачи была выбрана архитектура «Pipes and Filters» [2] (каналы и фильтры), реализуемая при помощи языка программирования C++ и Qt Framework.

Это архитектурный шаблон, который имеет независимые объекты, называемые фильтрами (компонентами), выполняющие преобразование данных, которые они получают, и каналы, которые соединяют фильтры и по которым передаются потоки данных.

Взаимодействие в паттерне «канал-фильтр» характеризуется последовательными преобразованиями потоков данных. Данные текут в одном направлении. Поток начинается с источника данных, затем данные поступают с помощью каналов во входной порт фильтра, где выполняется их преобразование, а затем передаются через его выходной порт по каналу к следующему фильтру. Процесс обработки заканчивается в целевом объекте.

Каналы и фильтры образуют конвейеры обработки данных (рис. 2).

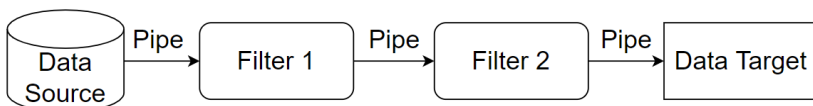


Рис. 2. Схема конвейера в архитектуре «Pipes and Filters»

Данная архитектура обладает следующими преимуществами:

- Свободное и гибкое соединение компонентов (фильтров).
- Позволяет менять фильтры без изменения других фильтров.
- Хорошо подходит для параллельной обработки данных.

Любой независимый модуль (синхронизатор, фильтр, конвейер) может быть реализован путем наследования от абстрактного класса Worker (рис. 3).

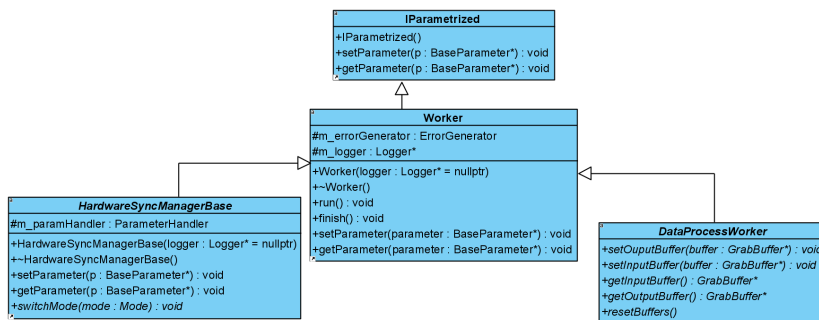


Рис. 3. Абстрактный класс Worker и его применение

Рассмотрим реализацию конвейера для режима «Continuous Shot», который позволяет производить как запись кадров на накопители, так и отображать в графическом интерфейсе приложения.

На схеме, представленной на рис. 4, отображены основные логические модули конвейера. Получаемые данные с блока чтения распараллеливаются на два потока, которые идут в блоки для

отображения и записи данных. Подробнее об этом будет рассказано далее.

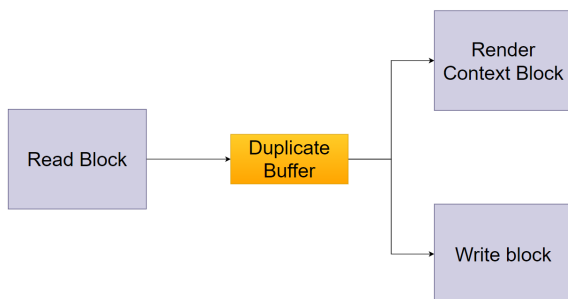


Рис. 4. Схема основных модулей конвейера «Continuous Shot»

Прежде чем рассказать об устройстве блоков, необходимо показать, как реализована работа с буферами данных и как реализованы каналы. Данные кадра, представляющие собой набор байтов и основные метаданные, такие как ширина, длина, формат, идентификатор камеры, хранятся в структуре `GrabBufferCell`.

Экземпляры данного класса создаются в фильтрах, порождающих данные, а по конвейеру передаются лишь указатели на такие структуры. Это позволяет избежать лишнего копирования данных, которое лишь отнимает ресурсы процессора. Для каждого нового кадра не выделяется отдельная ячейка памяти в виде объекта `GrabBufferCell`, а происходит запись в заранее выделенные. Обычно на один поток данных с камеры резервируется от 30 до 60 ячеек. За счет параллельной обработки ячейка буфера с кадром успевает пройти весь конвейер, прежде чем она будет перезаписана.

Каналы, соединяющие фильтры, реализуются наследниками абстрактного класса `GrabBuffer`. Они позволяют не только передавать данные от фильтра к фильтру, но и распараллеливать потоки данных (`Duplicate Buffer`), работать с несколькими потоками данных без слияния, используя одну структуру (`MultiBuffer`), класс `FreshBuffer` хранит только последний записанный в него кадр (рис. 5).

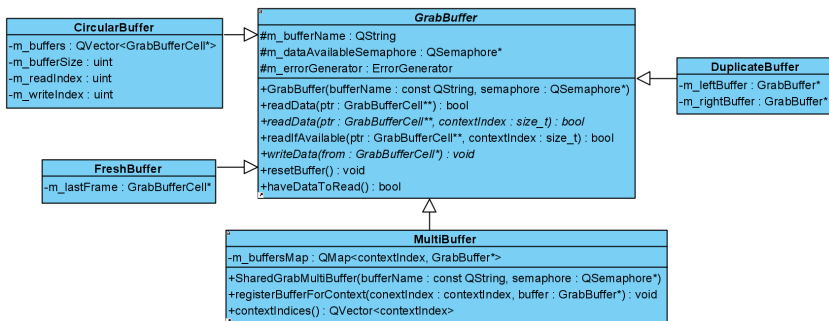


Рис. 5. Реализованные классы буферов

Абстрактный класс `GrabBuffer` имеет поддержку логирования ошибок, работающей на системе сигналов и слотов Qt. Другой важный объект в данном классе – семафор. Он позволяет потокам, которые читают данные из буфера, не проверять постоянно наличие данных в цикле, а ожидать их появления. Без этого ядра процессора будут работать на полную мощность, но не будут делать полезную работу.

Рассмотрим блок чтения данных, представленный на рис. 6.

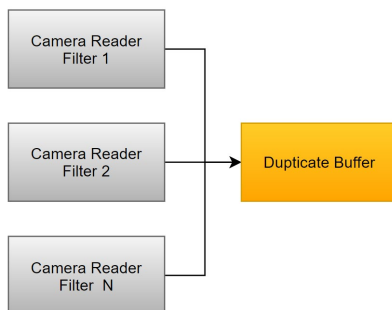


Рис. 6. Read Block

Для каждой камеры создается отдельный фильтр, который генерирует данные. Данная совокупность фильтров является глобальной, то есть все конвейеры используют одни и те же экземпляры этих объектов.

Далее перейдем к рассмотрению блока записи данных (рис. 7).

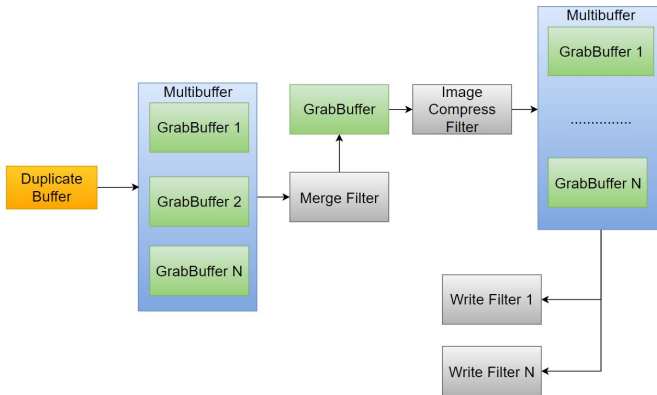


Рис. 7. Write Block

Фильтры реализованы в виде классов, которые имеют один входной и один выходной порт. Отдельный метод содержит в себе логику по обработке данных. Также каждый класс может быть подключен к обработчикам ошибок.

Отметим, что для каждого фильтра выделен отдельный поток исполнения (thread), реализованный при помощи класса-контейнера. Таким образом, работа фильтров осуществляется в параллельном режиме, что позволяет максимально быстро обрабатывать потоки данных.

На схемах, приведенных выше, представлено несколько видов буферов. GrabBuffer обозначает произвольный буфер, в который происходит непосредственно запись данных. Такие буферы создаются для под каждую линию конвейера, связанную с камерой.

DuplicateBuffer позволяет писать данные в два буфера, Multibuffer – в произвольное количество. Сами по себе они не хранят данные, а лишь передают на запись во внутренние буферы. Multibuffer отличается тем, что внутренние буферы в нем связаны с определённым контекстом, который определяется уникальным идентификатором камеры. Контекст есть у каждого кадра, который генерирует источник данных. Это позволяет объединить несколько потоков данных в одну структуру, однако реального их слияния не происходит. В результате можно, например, подключить к одному порту фильтра несколько буферов.

4. Результаты тестов

Реализованное приложение позволило успешно производить съемку данных с 4D-рига, оснащенного пятнадцатью камерами Basler.

Сжатие данных на видеокарте сохранило много ресурсов процессора. На рис. 8 показана загруженность процессора во время съемки.

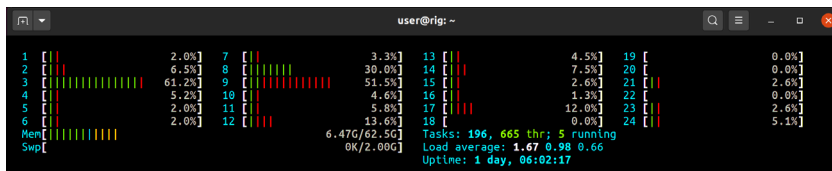


Рис. 8. Данные о загруженности процессора

При помощи разработанного приложения было произведено две успешные продакшн-съемки, то есть съемки актеров для реальных бизнес-проектов.

Заключение

Данная статья посвящена разработке и реализации приложения для получения данных с фотограмметрической установки. Были изучены API для работы с камерами машинного зрения, API для работы с сжатием изображений. Реализована и описана архитектура, позволяющая создавать и модифицировать конвейеры для обработки видеопотока.

Разработанное решение позволило успешно получать большой поток данных с фотограмметрической установки и производить его дальнейшую обработку.

В будущем предполагаются различные модификации приложения для новых устройств синхронизации, камер и прочих физических компонентов рига.

Список литературы

1. Direct Memory Access (DMA) [Электронный ресурс]. – Режим доступа: <https://www.techopedia.com/definition/2767/direct-memory-access-dma>
2. Об архитектуре «Pipes and Filters» [Электронный ресурс]. – Режим доступа: <https://homepages.fhv.at/thjo/lecturenotes/sysarch/pipes-and-filters.html>

Приложение для совместного прослушивания музыки с персонализированным плейлистом

Д. В. Макушин

Студент бакалавр

В. С. Тарасов

Старший преподаватель

Введение

На сегодняшний день существует множество различных музыкальных сервисов со своими мобильными приложениями для прослушивания треков. Для одиночного использования их более чем достаточно.

Но на мероприятиях, площадках да и просто посиделках в кругу друзей очередность и выбор треков должен зависеть от пожеланий всех участников.

Кто-то один проигрывает треки через звуковую колонку, а остальные могут заказать песни на свой вкус. Но тут возникают несколько проблем.

Все пользуются разными сервисами и заказ песни у человека, проигрывающего треки, заставляет его каждый раз искать необходимый трек через то приложение, которым он пользуется, которое у него установлено. Может даже возникнуть ситуация, когда желаемая песня есть в одном музыкальном сервисе, а в другом ее нет.

Еще одна проблема заключается в том, что трек надо будет выбирать каждые 2-5 минут, в зависимости от длительности. Можно конечно создать плейлист в едином музыкальном сервисе заранее. Но это спорное решение, ведь иногда надо заказать песню уже в процессе прослушивания. И, к тому же, никуда не исчезает проблема отсутствия необходимых треков в некоторых музыкальных сервисах. С другой стороны, если несколько заказов сделаны почти одновременно, какой трек включать первым? Хорошо было бы голосовать за очередность и выбор треков.

Одним из приложений, решающим некоторые вышеописанные проблемы, является RequestBox. Но в нём своя база данных треков, а так же нет возможности прослушивания. Человеку, подключенному к

звуковой колонке, всё так же придётся искать треки вручную, как и человеку, заказывающему песню.

1. Обработка и заказ песен

В любом мобильном приложении от большинства музыкальных сервисов есть возможность поделиться песней. Как получателя можно выбрать любое приложение, зарегистрированное в системе android с настроенным на текст intent-filter и action SEND [1].

Таким образом, приложение получит shared ссылку, то есть ту, которой можно поделиться из музыкального сервиса. Дальше необходимо обработать её и получить прямую ссылку на mp3 и остальную информацию о песне.

Чтобы осуществить вышеописанные действия без каких-либо ограничений в большинстве сервисов необходимо быть авторизованным под чьим-нибудь аккаунтом. Также, в большинстве сервисов для прослушивания песен, с web версии подписка не обязательна.

Следовательно, можно создать аккаунт, с которого будем получать ссылку на необходимую песню через web версию приложения.

2. Требования к приложению

При разработке рассматриваемой системы учитывались следующие требования:

- проигрывание песен, имея ссылку на mp3, обязательное условие
- работа в фоне, если приложение свёрнуто или телефон заблокирован;
- обработка shared ссылок от других музыкальных сервисов, получить которые можно просто поделившись песней в данное приложение, причём обязательное условие – пользователю не нужно вводить данные своего аккаунта в музыкальном сервисе
 - формирование плейлиста;
 - просмотр текущей проигрываемой песни;
 - голосование за очередность песен;
 - просмотр имени пользователя, заказавшего песню.

3. Реализация

Реализация серверной части выполнена на языке java и местами kotlin [2], который совместим с java. Ядро сервера – Spring boot [3], среда разработки приложений для JavaEE. Для работы с СУБД используется две библиотеки: Jooq [4] – DSL-генератор, для отображения базы данных в java код. И flyway [5] – инструмент для запуска набора скриптов миграции СУБД. Авторизация с помощью JSON Web Token (JWT) [6] и модуля Spring security [7].

Реализация клиентской части выполнена для операционной системы android, на языке kotlin [2]. Архитектура приложения – MVVM [8] (Model – View – ViewModel). Концепция заключается в отделении логики представления данных от бизнес-логики путем вынесения её в отдельный класс для более четкого разграничения. Для проигрывания музыки использован MediaPlayer [9]. Для сетевых запросов используется библиотека retrofit2 [10]. Это типобезопасный HTTP-клиент, позволяющий представить REST API в виде интерфейса. Можно управлять телами, заголовками, параметрами API-запросов и многим другим с помощью аннотаций, что делает этот процесс простым и понятным. Для асинхронных действий используется rxjava3 [11]. Она позволяет связывать асинхронные задачи и события в наблюдаемые (англ. observable) последовательности.

Взаимодействие сервера и клиента реализовано с помощью дополнительной библиотеки моделей, которые передаются в формате JSON, подключаемой на обеих сторонах. Помимо этого, в библиотеке реализована проверка некоторых полей у сущностей, используемая например при регистрации. Чтобы лишний раз не посылать запросы с клиента можно сделать валидацию на нём. Но и сервер без проверки данных, сохраняемых в БД, оставлять нельзя.

Если на сервере происходит какое-либо исключение, помимо передачи кода HTTP, будет передана модель ошибки из вышеописанной библиотеки, содержащая одно из значений внутренних кодов, реализованных в виде enum. Это нужно для того, чтобы на устройстве, зная внутренний код ошибки получить текстовое значение на нужном языке для показа пользователю.

4. Регистрация

Чтобы пользоваться приложением, пользователю необходимо зарегистрировать аккаунт. Здесь возможны два способа. Первый – зарегистрировать постоянный аккаунт введя логин, пароль, имя пользователя, почту (необязательно). Или второй – создать временный аккаунт, в подавляющем числе случаев его достаточно. Для этого необходимо лишь ввести имя пользователя. При этом создаётся пользователь, у которого одинаковый логин и имя, которое можно сменить в дальнейшем. При отсутствии действий более 24 часов временный аккаунт удаляется.

5. Комната

Для того, чтобы объединить несколько пользователей одним плейлистом, им необходимо находится в одной комнате. Создать

которую может любой юзер. Для чего потребуется выбрать следующие настройки будущей комнаты:

- Видно пользователя заказавшего песню. Если включено, то, вместе с остальной информацией песни, будет возвращаться id заказавшего песню юзера.

- Разрешить голосование за песни. Если включена, то можно проголосовать за или против песни, что даст ей +1 или -1 рейтинга соответственно. По убыванию которого пользователи будут получать плейлист.

- Любой может прослушивать песни. Если выставлена, то любому пользователю в ответ на запрос песен с сервера, в плейлисте, принадлежащему комнате, придут не только лишь информация о песне (исполнитель, название, длительность, id), а ещё и ссылка на mp3 и время когда эта ссылка устареет. Иначе эта информация придёт только одному пользователю, «прослушивающему» плейлист. Эту роль можно снять с себя, тогда другой юзер может её включить.

- Приоритет редко заказывающим. Суть в том, чтобы прибавлять баллов для песни, если её заказал тот юзер, который давно не заказывал.

6. Интерфейс пользователя

Приложение имеет интерфейс в стиле material design. Основные экраны представлены на рисунке.

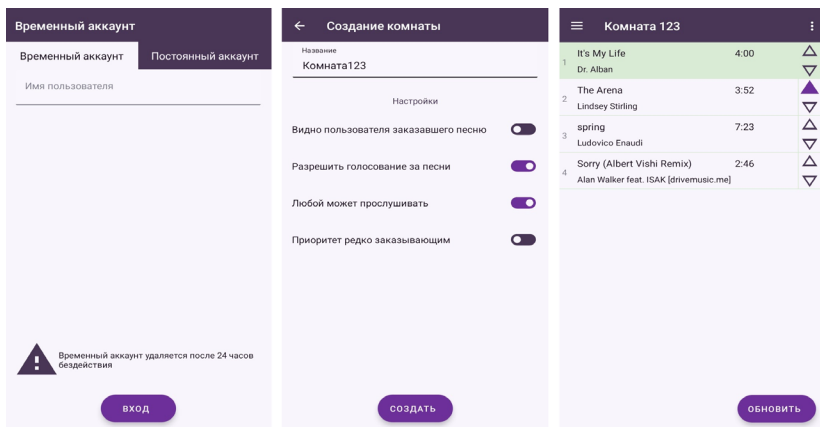


Рисунок. Интерфейс пользователя

На экране комнаты, в меню опций есть пункты «выйти из комнаты», «информация о комнате» или «Прослушивать» / «Закончить прослушивание». Последний пункт позволяет пользователю получать

ссылки на mp3 чтобы прослушивать плейлист. Та песня, которая у него проигрывается в данный момент, будет у других участников отображаться с зелёным фоном.

Заключение

Таким образом, реализовано клиент-серверное приложение с использованием современных подходов к разработке, решающее проблему формирования и прослушивания плейлиста разными людьми через разные музыкальные сервисы в реальном времени.

В приложении предусмотрено создание и вход в комнату, формирование плейлиста посредством заказов песен, используя возможность «поделиться» в приложениях музыкальных сервисов, и его прослушивание, голосование за очередность треков.

Список литературы

1. Introduction to Android [Электронный ресурс] : Android Developers. – Режим доступа : <https://developer.android.com/guide/index.html>
2. Kotlin docs [Электронный ресурс] : JetBrains. – Режим доступа: <https://kotlinlang.org/docs/home.html>
3. Spring Boot Reference Documentation [Электронный ресурс]. – Режим доступа : <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>
4. The jOOQ User Manual [Электронный ресурс] : Data Geekery GmbH. – Режим доступа : <https://www.jooq.org/doc/3.14/manual/>
5. Flyway documentation [Электронный ресурс] : Red Gate Software Ltd. – Режим доступа : <https://flywaydb.org/documentation/>
6. Mauricio Urraco. Implementing JWT with Spring Boot and Spring Security [Электронный ресурс]. – Режим доступа : <https://medium.com/@xoor/jwt-authentication-service-44658409e12c>
7. Spring Security Reference [Электронный ресурс]. – Режим доступа : <https://docs.spring.io/spring-security/site/docs/current/reference/html5/>
8. Guide to app architecture [Электронный ресурс] : Android Developers. – Режим доступа : <https://developer.android.com/jetpack/guide>
9. MediaPlayer overview [Электронный ресурс] : Android Developers. – Режим доступа : <https://developer.android.com/guide/topics/media/mediaplayer>
10. Retrofit [Электронный ресурс] : Square. – Режим доступа : <https://square.github.io/retrofit/>
11. RxJava [Электронный ресурс]. – Режим доступа : <https://github.com/ReactiveX/RxJava>

Универсальный испытательный стенд для тестирования и демонстрации работы интерфейсов человек-компьютер

А. П. Миронкин

Студент бакалавр

И. С. Шаталов

Студент бакалавр

Туровский Я. А

Доцент,
заведующий лабораторией
медицинской кибернетики

Введение

В настоящее время происходит стремительное развитие информационных технологий. Они активно применяются в науке, технике и медицине. В связи с этим возникает потребность в разработке новых способов взаимодействия человека с компьютером. Использование человек-компьютерных интерфейсов даст возможность человеку быстрее воспринимать информацию, обрабатывать большие объемы данных и работать со сложными устройствами, а при необходимости обеспечит людей с инвалидностью возможностью восполнить полученные ограничения опорно-двигательного аппарата или органов восприятия.

Бурное развитие кибернетики привело к появлению большого количества интерфейсов человек-компьютер. Разные группы ученых используют разные виды интерфейсов, начиная с голосовых и заканчивая нейрокомпьютерными, программное обеспечение которых зачастую создается под определенные платформы. Но не все они могут позволить качественно работать с вычислительными устройствами. Поэтому возникла необходимость иметь возможность тестировать совершенно различные интерфейсы на их скорость, точность и эргономичность, используя общий тестовый протокол.

Практическая и теоретическая значимость заключается в том, что разработанная информационная система, интегрированная в самоходное шасси, может быть использован для сравнения и

проверки работы интерфейсов на едином устройстве, что позволит точно выявлять ошибки и особенности работы, минимизировать сторонние факторы влияющих на результаты тестов. Также, имея единый программно-механический комплекс тестирования, мы сможем сравнивать результаты испытаний новых разработок со старыми без повторного проведения экспериментов.

Цель работы – создать универсальное единое устройство способное обеспечить возможность тестировать интерфейсы человек-компьютер созданные различными группами учеными и для различных систем.

Реализация системы испытательного стенда

Для создания испытательного стенда было принято решение использовать в качестве основы инвалидную коляску. Это обусловлено тем, что человек-компьютерные интерфейсы часто используется для помощи людям с ограниченными возможностями. Также коляска может обеспечить необходимую наглядность, и ее испытания в лабораторных условиях можно максимально точно приблизить к реальным-бытовым.

Для разработки было использовано кресло-коляска Ortonica Base 130 DY с шириной сидения 48см. Данная модель имеет компактную крестовину в основании сидения, благодаря которой удалось оптимально надежно и с необходимой возможностью подачи охлаждения разместить бортовое-управляющее устройство, организованное на базе стандартного компьютера.

Модификация стандартной версии с шириной сидения 48см позволила закрепить на металлическом каркасе в фронтальной части коляски опорную плоскость с характеристиками, позволяющими разместить два мотора, два редуктора, блок управления, систему фрикционной передачи крутящего момента, систему охлаждения, защитные кожухи. Благодаря увеличенной базе, удалось избежать потери проходимости, безусловно имеющие ключевое значение в любых устройствах передвижения. Также по периметру устройства были установлены датчики дальномеры для предотвращения столкновений.

В качестве основных двигателей были использованы щеточные электромоторы, в быту используемые в устройствах передачи вращательного движения режущему или другому инструменту. Необходимая мощность электродвигателя была определена опытным путем и составила 600w. Доступный двигатель с данной мощностью имеет скорость вращения 3000 оборотов в минуту. Для получения оптимально необходимой скорости движения и нужного направления вращения были использованы поворотные понижающие редукторы.

Системы фрикционной передачи крутящего момента с регулируемой прижимной силой позволили избежать пробуксовки между колесами и резиновыми шкивами. Данные механические узлы дали возможность точно настраивать и удобно обслуживать систему, обеспечили минимальные потери мощности установки. Блоки принудительного охлаждения, размещенные в местах крепления моторов, искусственно нагнетают холодный воздух в области якорей двигателей, что дает возможность без повреждений проводов питания работать продолжительное время.

Для управления двигателями был использован микроконтроллер Arduino Nano. Устройство управления было отделено от основного компьютера системы для устранения просадок напряжения или помех вызываемых работой электромоторов. Прошивка Arduino позволяет через COM порт получить команду в виде массива байт и интерпретировать его в команду и, используя блоки реле, подать напряжение на моторы. Использование блоков реле, оснащенных оптопарами, необходимо для разделения высоковольтных и низковольтных систем. Такой подход позволит максимально обезопасить вычислительные устройства и пользователя.

Далее, для безопасности пользователя, было принято решение сделать защиту от неправильно отправленных или же неверно интерпретируемых команд. То есть стояла задача сделать систему, которая не позволит пользователю столкнуться с каким-либо препятствием.

Для решения этой задачи было использовано 4 ультразвуковых дальномера, расположенных по сторонам коляски. Выбор пал именно на эти датчики, так как они являются самыми оптимальными по соотношению цена/точность. Датчики работают очень просто и поэтому надежно, а именно на них подается питание +5В и в момент посылы на них команды, один из динамиков отправляет ультразвуковую волну, в это же время второй динамик настраивается на прием этой волны. Время, за которое звуковая волна вернется, фиксируется и далее путем деления на определенную константу превращается в расстояние. Управляются они микроконтроллером Arduino Nano, питается отдельным блоком питания, коммутация выполнена экранированными проводами. На компьютер данные передаются через COM порт в виде массива байт, который десериализуется в расстояние, которое принимает клиент Filter, написанный на языке C#.

Для корректной работы были установлены задержки в 1000 миллисекунд, так как COM порт оптимально принимает и передаёт данные при достаточно медленном их поступлении. В клиенте

происходит проверка на возможность выполнения команды, в зависимости от расстояния, пришедшего с конкретного датчика, соответствующего этой команде. Если расстояние больше определенного (в нашей тестовой версии определенным расстоянием было принято считать 50см), то команда выполняется, то есть передается дальше через NMServer на клиент, отвечающий за управление двигателем, DrelControl. Иначе, выводится сообщение о невозможности выполнения, и команда не передается.

Программа DrelControl является примером стандартного клиента, написанного на языке C# для работы с проектом NMServer, который реализует серверную часть программного обеспечения устройства. Задача DrelControl принять от Filtr сообщение с верным кодом доступа и командой определенного формата. Если все условия соблюдены, то клиент отправляет через COM порт на Arduino команду. Основные этапы и особенности работы микроконтроллера, управляющего моторами, описаны выше. Связка программ управления самоходного шасси Filtr и DrelControl позволяет безопасно и надежно управлять коляской. Общая схема реализации работы комплекса программного обеспечения указана на схеме изображенной на рис. 1.

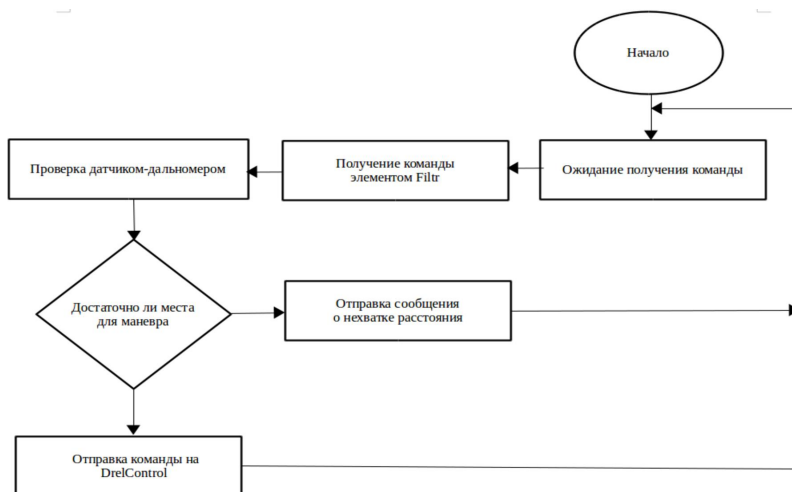


Рис. 1. Схема реализации управления испытательной платформой

При завершающей сборке универсального испытательного стенда электродвигатели, системы фрикционной передачи, а также агрегатные

узлы и системы управления подачи питания были закрыты металлическими кожухами. Данные необходимые меры защиты позволили обезопасить испытуемых и защитить самоходное шасси от преждевременного выхода из строя. Легкосъемный кожухи и модульная конструкция позволят легко обслуживать испытательный стенд. Последняя версия самоходного шасси представлена на рис. 2.



Рис. 2. Вид итоговой сборки испытательного стенда

Заключение

Использование кресла-коляски позволяет на действительных примерах тестировать человек-компьютерные интерфейсы любой сложности. Имея константные условия, обеспеченные единым проверочным устройством, появляется возможность сохранять данные испытаний и сравнивать результаты без повторных тестов. Понятная и надежная конструкция позволяет проводить исследования на людях любых возрастных групп. Использование стандартного донора дает возможность заменять изношенные детали. Имея интуитивно понятную систему питания установки, двигательного аппарата, и управляющей системы испытательный стенд можно подстраивать под специфические задачи а после возвращать к исходному состоянию без кардинальных изменений.

Бортовое-навигационное устройство, организованное на базе стандартного компьютера, имеет запас вычислительных мощностей позволяющих реализовывать математически сложные задачи. При необходимости компьютер можно легко модернизировать или полностью заменить.

Рама данной модели имеет необходимый запас прочности и позволяет при необходимости усовершенствовать конструкцию.

Проведенные первые тесты на небольших группах добровольцев были проведены с использованием методов, описанных в статьях [1-3] и доказали эргономичность стенда. В ходе испытаний были выявлены особенности реализации и возможные пути развития интерфейсов, что доказывает компетентность использование данного программно-аппаратного комплекса

Список литературы

1. Туровский, Я. А. Вариабельность сердечного ритма пользователей видеоокулографическим интерфейсом в процессе обучения управлением самоходным шасси / Я. А. Туровский, А. В. Алексеев // Вестник ВГУ – 2017а. – № 1. – С. 118-124.

2. Туровский, Я. А. Вариабельность сердечного ритма в ходе обучения пользователей применению интерфейсов человек-компьютер / Я. А. Туровский, С. В. Борзунов, А. А. Вахтин, А. В. Алексеев, А. В. Мамаев // Вестник ВГУ – 2018. – № 2. – С. 255-263.

3. Туровский, Я. А. Анализ движения глаз человека при управлении самоходным шасси с использованием системы видеоокулографического интерфейса / Я. А. Туровский, С. Д. Кургалин, А. В. Алексеев // Сенсорные системы – 2017б. – Т. 31. – № 1. – С. 51-58.

О разложении в ряд по неортогональным цилиндрическим функциям Бесселя первого рода целого порядка

Э. С. Нурулаев

Студент бакалавр

Е. А. Киселев

Доцент

Введение

В настоящее время цилиндрические функции Бесселя применяются достаточно широко. В большинстве случаев используются ряды, состоящие из попарно ортогональных функций. Ярким примером являются краевые задачи математической физики [1]. Тем не менее, существуют и модели, в которых используются именно неортогональные ряды. В качестве примера можно привести процедуру приближенного построения диаграмм направленности кольцевых фазированных антенных решеток [2], а также некоторые задачи квантовой теории [3]. Разложение по неортогональной системе функций, как правило, является достаточно трудной задачей, поскольку технология рядов Фурье здесь уже не применима. Приходится использовать иные математические подходы, более частного характера. Важно при этом учитывать особенности исследуемой системы функций. Данная работа посвящена созданию алгоритмов разложения по цилиндрическим функциям целого порядка. Во-первых, именно они чаще всего возникают в различных приложениях. Во-вторых, для функций Бесселя целого порядка известно достаточно много интегральных соотношений, которые могут быть положены в основу алгоритмов разложения.

1. Теоретическая часть

Постановка задачи: дана функция

$$f(x) = \sum_{m=0}^{M-1} A_m J_n(\lambda_m x), \quad (1)$$

где n – заданное целое число, M – количество компонент (считается конечным, но заранее неизвестным), λ_m , A_m – неизвестные действительные параметры, которые требуется определить. В

дальнейшем будем предполагать, что $n \geq 0$, поскольку $J_{-n}(x) = (-1)^n J_n(x)$, $n \in Z$. Кроме того, так как $J_n(-x) = (-1)^n J_n(x)$, $n \in Z$, можно считать, что $\lambda_m > 0$. Для удобства также будем предполагать, что значения λ_m упорядочены по убыванию.

Рассмотрим следующее интегральное соотношение [4]:

$$\int_0^{\infty} \exp(-ax) J_n(\lambda x) dx = \frac{\lambda^n}{\sqrt{a^2 + \lambda^2} \left(\sqrt{a^2 + \lambda^2} + a \right)^n}, \quad (2)$$

где $n \in Z$, $a \in C$, $\lambda > 0$. Положим в формуле (2) $a = -i\omega$, $\omega > 0$, $\omega \neq \lambda$:

$$\int_0^{\infty} \exp(i\omega x) J_n(\lambda x) dx = \frac{\lambda^n}{\sqrt{\lambda^2 - \omega^2} \left(\sqrt{\lambda^2 - \omega^2} - i\omega \right)^n}. \quad (3)$$

При $\omega > \lambda$ получим выражение

$$\int_0^{\infty} \exp(i\omega x) J_n(\lambda x) dx = \frac{(-i)^{n+1} \lambda^n}{\sqrt{\omega^2 - \lambda^2} \left(\sqrt{\omega^2 - \lambda^2} - \omega \right)^n}. \quad (4)$$

Выделяя действительную и мнимую часть интеграла, получим, что при четном $n = 2p$, $p = 0, 1, 2, \dots$ имеет место равенство

$$\dots \quad (5)$$

Соответственно, при нечетном $n = 2p + 1$, $p = 0, 1, 2, \dots$ получим

$$\int_0^{\infty} \sin(\omega x) J_{2p+1}(\lambda x) dx = 0, \omega > \lambda. \quad (6)$$

Обсудим подробнее смысл полученных соотношений (5) и (6). Если в пространстве $L_2(R_+)$ задать скалярное произведение формулой

$$(f, g) = \int_0^{\infty} f(x) \overline{g(x)} dx, \quad (7)$$

то равенство (5) означает, что косинус ортогонален всем функциям Бесселя четного порядка, у которых значение параметра λ меньше, чем частота косинуса ω . Смысл соотношения (6) аналогичен, с той лишь разницей, что для функций Бесселя нечетного порядка вместо косинуса используется синус.

В дальнейшем нам потребуются также следующие два соотношения, справедливые при $\omega < \lambda$:

$$\int_0^{\infty} \cos(\omega x) J_n(\lambda x) dx = \frac{\left(\sqrt{\lambda^2 - \omega^2} + i\omega\right)^n + \left(\sqrt{\lambda^2 - \omega^2} - i\omega\right)^n}{2\lambda^n \sqrt{\lambda^2 - \omega^2}}, \quad (8)$$

$$\int_0^{\infty} \sin(\omega x) J_n(\lambda x) dx = \frac{\left(\sqrt{\lambda^2 - \omega^2} + i\omega\right)^n - \left(\sqrt{\lambda^2 - \omega^2} - i\omega\right)^n}{2i\lambda^n \sqrt{\lambda^2 - \omega^2}}. \quad (9)$$

Они легко следуют из формулы (3). Выясним, что из себя представляют величины в числителях равенств (8) и (9). Для этого комплексное число $\sqrt{\lambda^2 - \omega^2} + i\omega$ удобно представить в экспоненциальной форме:

$$\sqrt{\lambda^2 - \omega^2} + i\omega = \lambda e^{i\phi}, \quad \phi = \arctg\left(\frac{\omega}{\sqrt{\lambda^2 - \omega^2}}\right). \quad (10)$$

Подставляя равенство (10) в формулы (8) и (9), получим:

$$\int_0^{\infty} \cos(\omega x) J_{2p}(\lambda x) dx = \frac{\cos(2p\phi)}{\sqrt{\lambda^2 - \omega^2}}, \quad (11)$$

$$\int_0^{\infty} \sin(\omega x) J_{2p+1}(\lambda x) dx = \frac{\sin((2p+1)\phi)}{\sqrt{\lambda^2 - \omega^2}}. \quad (12)$$

Приступим к описанию алгоритма разложения. Прежде всего отметим, что все интегралы вычисляются приближенно по формуле трапеций, поскольку на практике при работе с цифровыми сигналами все равно неизбежно придется использовать квадратурные формулы.

1. Выбираем достаточно большое значение $\omega_0 > \lambda_0$ и шаг $\Delta\omega$. Величина ω_0 определяет диапазон поиска значений λ_m , а $\Delta\omega$ – требуемую разрешающую способность. Задавать ω_0 следует, опираясь на то, какие характерные значения могут принимать параметры λ_m в конкретной прикладной задаче.

2. Полагаем $m = 0$, $\omega = \omega_0$, $f_m(x) = f(x)$.

3. Берем $\lambda_m = \omega$, а коэффициент A_m вычисляем по одной из следующих формул:

$$A_m = \frac{\sqrt{\lambda_m^2 - \omega_m^2}}{\cos(2p\phi_m)} \int_0^\infty \cos(\omega_m x) f_m(x) dx, n = 2p, \quad (13)$$

$$A_m = \frac{\sqrt{\lambda_m^2 - \omega_m^2}}{\sin((2p+1)\phi_m)} \int_0^\infty \sin(\omega_m x) f_m(x) dx, n = 2p+1, \quad (14)$$

где $\omega_m = \omega - \Delta\omega / 2$, $\phi = \arctg\left(\omega_m / \sqrt{\lambda_m^2 - \omega_m^2}\right)$.

4. Строим $f_{m+1}(x)$ по формуле

$$f_{m+1}(x) = f_m(x) - A_m J_n(\lambda_m x). \quad (15)$$

Если рассчитанный коэффициент A_m оказался близок к нулю, то можно положить $f_{m+1}(x) = f_m(x)$.

5. Повторяем пункты 3 и 4, заменяя m на $m+1$, ω на $\omega - \Delta\omega$. Вычисления продолжаем, пока $\omega > 0$.

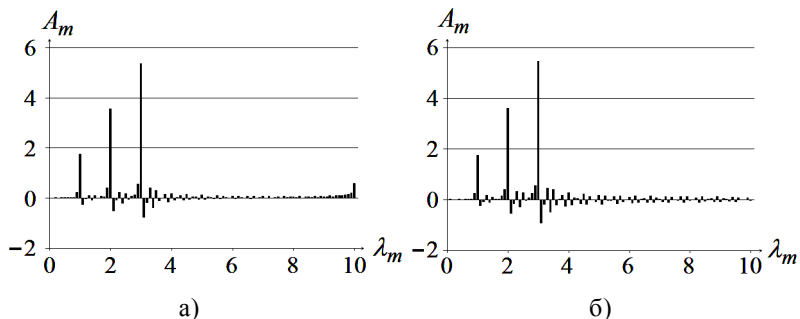
2. Практическая часть

Рассмотрим несколько примеров работы алгоритма. На рисунках ниже приведены гистограммы вычисленных с помощью предлагаемого алгоритма коэффициентов разложения A_m тестового сигнала (1) для функций Бесселя нулевого (рисунок а) и первого порядка (рисунок б). Точные значения параметров в обоих случаях таковы: $M = 3$, $\lambda_0 = 3$, $\lambda_1 = 2$, $\lambda_2 = 1$, $A_0 = 6$, $A_1 = 4$, $A_2 = 2$. На гистограммах можно видеть достаточно четко выраженные максимумы, что позволяет определить по их местоположению значения параметров λ_m . Величины амплитуд A_m найдены с погрешностью $\sim 10\%$. Точность можно повысить, уменьшив шаг, используемый при численном интегрировании, однако это не всегда возможно при работе с цифровыми сигналами.

Заключение

В данной работе предложен новый алгоритм разложения в ряд по неортогональным цилиндрическим функциям Бесселя целого порядка. В его основе лежит несколько общеизвестных интегральных соотношений. Сам алгоритм строится по рекурсивной схеме, что позволяет избежать слишком громоздких вычислений. По этой причине его достаточно легко реализовать в виде программы. К недостаткам алгоритма следует отнести необходимость выполнения численного интегрирования.

В дальнейшем планируется провести обобщение предложенного алгоритма на сферические функции Бесселя, выполнить анализ его вычислительной устойчивости, а также применить данный алгоритм для решения прикладных задач, связанных с теорией фазированных антенных решеток.



$a - n = 0, \bar{b} - n = 1$

Рисунок. Гистограмма коэффициентов A_m

Список литературы

1. Владимиров, В. С. Уравнения математической физики / В. С. Владимиров, В. В. Жаринов. – Москва : ФИЗМАТЛИТ, 2004. – 400 с.
2. Виноградов, А. Д. Математическое описание диаграмм направленности эквидистантных кольцевых фазированных антенных решеток с использованием функций Бесселя / А. Д. Виноградов, Л. А. Минин, В. А. Работкин // Антенны. – 2017. – № 10(237). – С. 5–12.
3. Берсон, И. Я. Отражение электрона от стенки в присутствии электромагнитной волны / И. Я. Берсон, Х. Я. Бондарс // Квантовая электроника. – 1974. – Т. 1, № 7. – С. 1612–1616.
4. Лаврентьев, М. А. Методы теории функций комплексного переменного / М. А. Лаврентьев, Б. В. Шабат. – Москва : Наука, 1965. – 716 с.

Разработка приложения для голосового ввода с восстановлением пунктуации в тексте

Д. Е. Первицкий

Студент бакалавр

Д. И. Соломатин

Старший преподаватель

Введение

В современных компьютерных системах все больше внимания уделяется на построение интерфейса для ввода информации с помощью голоса. Но большинство таких систем выдают результат в виде последовательности слов без каких-либо знаков препинания, делая такие системы крайне неудобными.

Восстановление пунктуации в тексте – это одна из задач в области NLP (Natural Language Processing). Данная задача подразумевает, что исходный текст не имеет знаков препинания, а современные системы распознавания речи как раз и возвращают текст без каких-либо знаков препинания.

Данная работа посвящена созданию приложения с графическим интерфейсом, который облегчит ввод информации, а именно сможет распознавать человеческую речь с дальнейшей расстановкой знаков препинания.

1. Распознавание голоса в текст

Современные системы распознавания речи могут распознавать речь от нескольких носителей и имеют огромный словарный запас на различных языках. Сначала речь должна быть преобразована из физического звука в электрических сигнал с помощью микрофона. А затем цифровые данные можно использовать для транскрибирования аудио в текст.

Для языка программирования Python существует каталог программного обеспечения PyPI в котором есть несколько пакетов распознавания речи, среди которых есть SpeechRecognition. [1] Этот пакет позволяет получать доступ к микрофону и обрабатывает аудиофайлы с нуля. Также он выступает посредником для нескольких популярных речевых API.

Были рассмотрены наиболее популярные системы распознавания речи, такие как Google Speech Recognition, Kaldi, Yandex Speech Recognition. Все системы справлялись с задачей примерно на одном уровне. [2] Но поскольку SpeechRecognition поставляется из коробки с ключом API для Google, то и мой выбор остановился на этой ауди-системе.

2. Анализ методов машинного обучения, направленные на решение задач обработки текста.

Расстановка знаков препинания бывает затруднительна даже для грамотного человека, а на сколько это сложная задача для машины, даже сложно оценить. Это связано с тем, что расстановка знаков препинания не однозначна и очень сильно привязана к контексту. К примеру, сложное предложение может быть разбито на более простые и вместо запятой можно поставить точку.

Данную задачу можно решить с помощью синтаксического дерева. Но из-за того, что грамматика русского языка очень многогранна, то такой подход кажется слишком трудным, если вообще реальным.

Другой подход для решения этой задачи будет использование Марковских сетей, рекуррентных нейронных сетей или же архитектуры Transformer. В данной работе будет рассмотрена последняя.

BERT (Bidirectional Encoder Representations from Transformers) – двунаправленная модель с transformer – архитектурой, заменившая собой последовательные по природе рекуррентные нейронные сети, с более быстрым подходом на основе механизма внимания [3].

Пример данной модели на рис. 1.

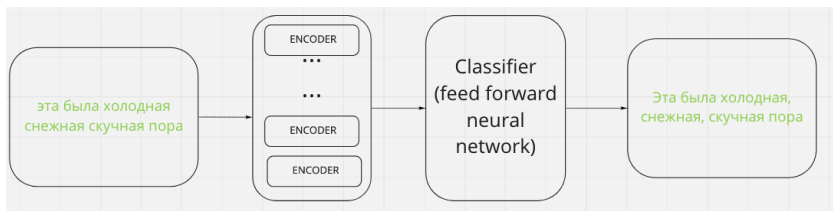


Рис. 1. Абстрактная модель

Bert принимает на вход последовательность слов, которая затем движется по стеку encoder, где каждый слой применяет механизм внимания и передает результаты в сеть прямого распространения, после чего переходит по стеку выше.

Для обработки слов с помощью моделей машинного обучения, слова необходимо перевести в некоторое числовое представление. Выделяют 2 подхода. Первый, character-level подход, когда на вход нейронной сети передается буква, кодируемая числом от 0 до 32 (плюс еще какой-то запас на знаки препинания). И world-level подход, когда на вход подается целое слово (или хотя бы по слогам). На практике как раз используется последний подход.

Всем словам на входе в нейронную сеть назначается не число, а некий вектор, чтобы представлять слова в виде, который может передавать их семантику или смысловые отношения. Это позволяет сопоставлять одному слову сразу несколько близких по смыслу слов (смотря по какой оси считать).

Также с векторами можно происходить арифметические операции.

Популярный пример: если из вектора "король", вычесть вектор "мужчина" и прибавить вектор для слова "женщина", то получится некий вектор-результат. И он будет равен "королева". И это не абстрактный пример [4].

Такой подход получил название Embeddings [5]. Все пакеты машинного обучения (TensorFlow, PyTorch) позволяют первым слоем нейросети поставить специальный слой Embedding Layer, который делает это автоматически. То есть на вход нейросети подаем обычный номер слова в словаре, а Embedding Layer, самообучаясь, переводит каждое слово в вектор указанной длины, скажем, в 32 числа.

Идея в основе BERT лежит очень простая: на вход нейросети будут подаваться фразы, в которых 15% слов заменено на [MASK], и обучим нейронную сеть предсказывать эти закрытые маской слова. По сути, такая нейронная сеть на своем выходе создает векторные представления для слов, и даже целых фраз. А навесив сверху над такой языковой моделью небольшой блок из пары дополнительных слоев нейронов, можно дообучить эту нейронную сеть на любые задачи [4].

Например, если подавать на вход нейросети фразу "Я пришел в [MASK] и купил [MASK]", она должна на выходе показать слова "магазин" и "молоко". Это упрощенный пример с официальной страницы BERT, на более длинных предложениях разброс возможных вариантов становится меньше, а ответ нейросети однозначнее [4].

3. Применение модели и ее оценка.

Веса инициализируются весами модели с архитектурой BERT, предобученной на большом корпусе русского языка ("DeepPavlov/rubert-base-cased-sentence"). [6]

Для обучения модели, используется готовая модель [7] предсказывающей наличие знаков препинания, используется новости с сайта Lenta из корпуса “Тауга”, раздел “All News”.

Теперь, когда у нас есть модель Bert и модель для предсказания знаков препинания можно изобразить схему рис.2, с помощью которой мы будем преобразовывать распознанный текст без каких-либо знаков препинания в текст, с расставленными знаками препинания.

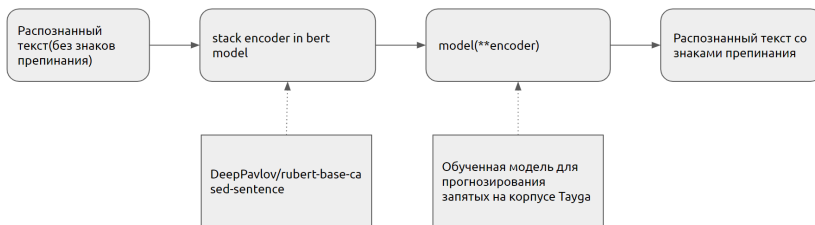


Рис. 2. Схема работы расстановки пунктуации

Оценка результатов работы расстановки знаков препинания проводилась на 20 страницах романа Толстого “Война и мир”. Весь этот текст был разделен на блоки предложений до 600 символов (т.к. максимальная длина выборки при обучении в районе 600 символов), после чего во всем блоке удалялись все знаки препинания. Дальше все блоки с исходным текстом прогонялись через модель. Такой же алгоритм оценки был проделан и для группы текстов из новостей. Результаты работы в таблице.

Таблица

Символ/выборка для теста	Точка (.)	Запятая (,)
Война и мир	53.26	89.4
Новостные тексты	74.3	93.75

Большой разброс в результатах можно объяснить тем, что в обучающей выборке для нашей модели были использованы новости, формат которых отличен от художественного формата текста, которым и является Война и Мир. Модель не справляется с текстом, в котором есть прямая речь, обращения. Также в тексте Толстого очень много предложений, состоящих меньше чем из 3 слов, что также создает трудности для правильного предсказания пунктуации.

В тоже время модель себя неплохо показала на выборке из текста, стиль которых похож на стиль из обучающей выборки.

4. Реализация графического интерфейса.

Для реализации графического интерфейса был выбран фреймворк PyQt5 [8] из-за его простоты и наличия PyQt5 Designer, с помощью которого можно без проблем нарисовать красивый GUI. Итоговый вариант графического интерфейса представлен на рис. 3.

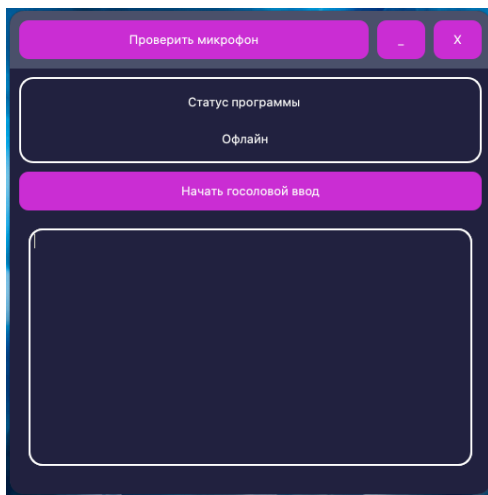


Рис. 3. Графический интерфейс

Также была реализована поддержка hot key, с помощью которых можно работать с приложением в фоновом режиме. При нажатии определенной клавиши (пользователь может задать любую удобную ему комбинацию), происходит старт/стоп распознавания голоса. Также есть комбинация клавиш, которая позволяет скопировать надиктованный текст в буфер обмена.

5. Проверка результатов

Результаты работы приложения вы можете оценить самостоятельно. Весь текст в этой статье (кроме списка литературы из-за специфики оформления) был набран с помощью данного приложения (за исключением исправлений пунктуации, где присутствуют аббревиатуры и английские слова, также запись происходила в

помещение без сторонних шумов с максимально четкой дикцией). Всего было исправлено 7 предложений вручную.

Заключение

В данной статье были освещены только самые ключевые моменты приложения. В итоге данной разработки получилось рабочее приложение, которое можно использовать в повседневной жизни. В дальнейшем планируется улучшение данного функционала, а именно добавление новых слов (профессиональных терминов) в словарь, а также использование системы распознавания голосов с публичным открытым кодом, чтобы видеть сразу результаты распознавания голоса без какой-либо задержки.

Список литературы

1. SpeechRecognition documentation [Электронный ресурс]. – режим доступа: <https://pypi.org/project/SpeechRecognition/>
2. Хабр/Поиск оптимальной системы аудио распознавания речи с закрытым исходным кодом, но имеющими открытые API, для возможности интеграции [Электронный ресурс]. – Режим доступа : <https://habr.com/ru/post/470696/>
3. BERT model documentation documentation [Электронный ресурс]. – режим доступа : https://huggingface.co/transformers/model_doc/bert.html
4. Хабр/BERT – state – of – the – art языковая модель для 104 языков [Электронный ресурс]. – режим доступа: <https://habr.com/ru/post/436878/>
5. Neural Network Embeddings Explained | by Will Koehrsen [Электронный ресурс]. – режим доступа : <https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526>
6. DeepPavlov model [Электронный ресурс]. – Режим доступа: <https://huggingface.co/DeepPavlov/rubert-base-cased-sentence>
7. GitHub Repository Leshal7 Punctuation [Электронный ресурс]. – Режим доступа : <https://github.com/Leshal7/Punctuation>
8. PYQT5 / documentation [Электронный ресурс]. – Режим доступа : <https://pyqt5.files.wordpress.com/2017/06/pyqt5tutorial.pdf>

Адаптация тиражируемого решения от фирмы 1С (1С: Управление нашей фирмой) и разработка приложения на мобильной платформе 1С

Р. И. Погорелов

Студент бакалавр

А. В. Копытин

Доцент

Е. А. Копытина

Старший преподаватель

Введение

В настоящее время с целью облегчения труда работников склада, многие руководители коммерческих организаций приобретают специальное программное обеспечение. Используя компьютерные программы, можно организовать и автоматизировать правильный синтетический и аналитический учет склада. Одним из готовых прикладных решений для автоматизации управленческого учета в сервисных, производственных и торговых компаниях является «1С:УНФ 8», которое поддерживает ведение в единой информационной базе управленческого учета по всем участкам, необходимым малому предприятию. По причине того, что все товаручетные системы создаются разработчиками как массовые продукты, рассчитанные на широкий круг потребителей, они не всегда могут учитывать специфику того или иного предприятия, поэтому появилась необходимость усовершенствования конфигурации «1С:УНФ 8» и создание новой конфигурации для мобильной платформы. Что позволит сотрудникам компании эффективно взаимодействовать, а также ускорит обработку заявок на материалы.

1. Постановка задачи

Опишем текущий процесс взаимодействия сотрудников строительной фирмы-заказчика ООО «Регион Проект» (см. [1]): производитель работ на листке пишет необходимые материалы и по электронной почте или мессенджеру отправляет фотографию листа в планово-технический отдел (ПТО) (если необходимо узнать статус

заявки, то производитель работ звонит в ПТО); затем сотрудник ПТО составляет заявку в MS Excel, после чего согласует с руководителем и составляет заказ поставщику; далее заказ поставщику согласуется с руководителем, если сумма заказа менее пятидесяти тысяч рублей или с директором, если сумма заявки более пятидесяти тысяч рублей. В итоге сотрудник ПТО оформляет заказ поставщику; сотрудники склада приносят материал, после чего оформляют перемещение на объект.

Следовательно, в текущем взаимодействии можно выделить следующие недостатки:

- Отсутствие механизма отслеживания документов.
- Отсутствие возможности быстрого формирования отчетов.
- Отсутствие механизма оповещений о создании, изменении документов.
- Нет единой системы для хранения всех данных.

Целью описываемой в данной статье работы является автоматизация взаимодействия сотрудников строительной фирмы-заказчика ООО «Регион Проект», для чего необходима адаптация конфигурации «1С:УНФ 8» и разработка мобильного приложения для Android и IOS.

К доработанной конфигурации «1С: УНФ 8» были выдвинуты следующие требования:

- Наличие нового документа «заявка» с табличной частью номенклатуры и полями: дата, номер заявки, статус заявки, приоритет, объект, ФИО прораба, автор.
- Наличие доработанного документа «заказ поставщику», позволяющего отображать номер заявки для каждой позиции из списка товаров.
- Наличие механизма группового создания «заказов поставщикам» на основании заявок.

К мобильному приложению были выдвинуты следующие требования:

- Наличие установочного файла для Android и IOS.
- Оповещение о создании, изменении документов.
- Работа с документами «заявка» и «заказ поставщику».
- Возможность согласования документов «заявка» и «заказ поставщику».

2. Реализация логики

Основное приложение представляет собой совокупность толстого клиента и файловой БД, которые используют собственный (встроенный) протокол обмена.

Взаимодействие мобильного приложения с основным выполняется при помощи расширения веб-сервера. Расширение веб-сервера выполняется в процессе сервисного хоста, который выполняет функцию приема/передачи сообщений из/в менеджер сервисов. В качестве сервисного хоста может использоваться веб-сервер IIS или Apache. Расширение веб-сервера содержит в себе пул соединений, а также другие механизмы, через которые идет взаимодействие с БД системы «1С:Предприятие». В данном случае Механизм Web-сервисов, реализованный в системе «1С:Предприятие», использует стандарт HTTP 1.1 (см. [2]).

Команды для выполнения обмена данными будут инициированы из мобильного приложения. При этом будет произведено обращение к Web-сервису, предоставляемому основным приложением, и вызваны его соответствующие операции, которые возвращают данные обмена. Для описания возвращаемых значений используется пакет XDTO [3].

Мобильное приложение, установленное на устройстве, содержит мобильную платформу и информационную базу. В свою очередь, информационная база включает аналог файловой информационной базы, предназначенной для хранения данных пользователя, и мобильное приложение, то есть программный код, выполняющийся на мобильном устройстве (см. [4]).

3. Реализация интерфейса

После входа в информационную базу, слева пользователь видит новую подсистему «Заявки и согласования», которая была добавлена в процессе адаптации конфигурации (рис. 1).

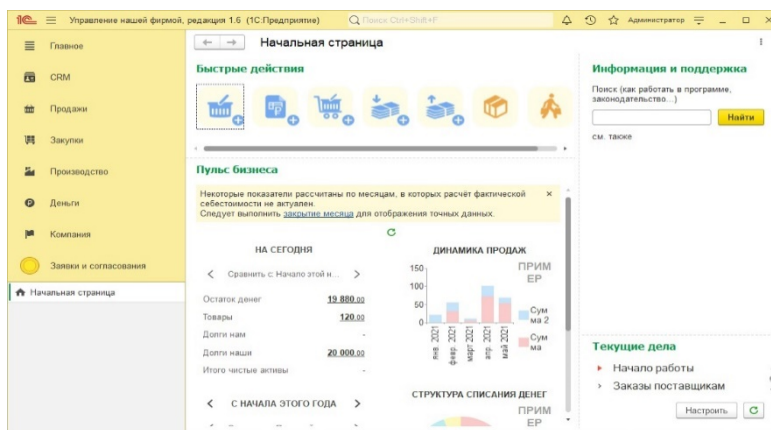


Рис. 1. Начальная страница

Подсистема включает в себя следующие команды: документ «Заявка»; справочник «Приоритеты заявок»; обработка «Групповое создание заказов поставщикам по заявкам». После перехода к разделу «заявка», пользователь видит все документы заявок в таблице со следующими данными: дата заявки. номер заявки, статус Заявки, объект, приоритет, ФИО Прораба, комментарий, автор.

В документе «Заказ поставщику» добавлено новое поле «Заявка» в табличной части. Данное поле отображает заявку, на основании которой была добавлена строка номенклатуры в заказ (рис.2).

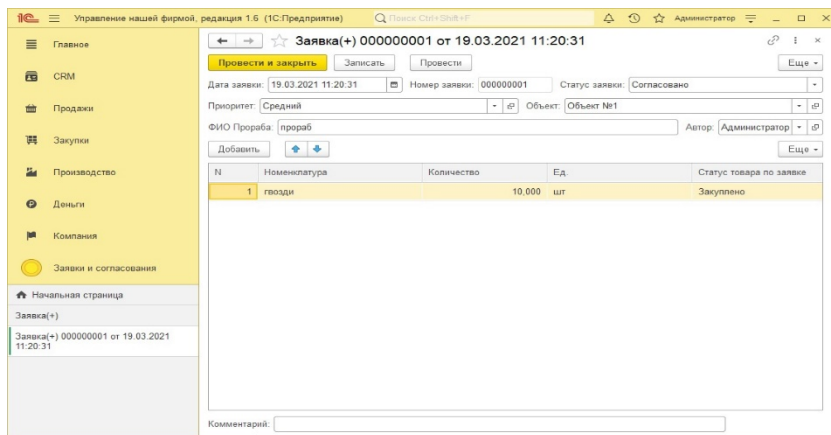


Рис. 2. Документ «Заказ поставщику»

Интерфейс разработанного документа «Заказ поставщику» в мобильном приложении показан на рис.3, рис.4.

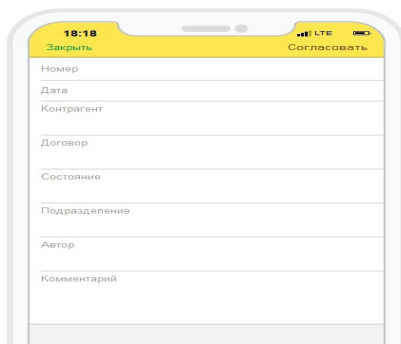


Рис. 3. Документ «Заказ поставщику» (1 страница)

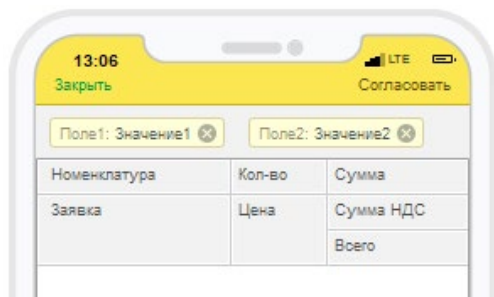


Рис. 4. Документ «Заказ поставщику» (2 страница)

Заключение

В данной статье была рассмотрена адаптация конфигурации «1С: УНФ» и реализация мобильного приложения для работы с 1С: УНФ. Описанная работа позволит фирме-заказчику организовать автоматизированный учет документации и слаженную работу всех сотрудников. Сотрудники ПТО смогут в удобном формате работать с самыми необходимыми документами: заявка, заказ поставщику, приходная накладная, перемещение товаров.

Список литературы

1. Петрикеева, Н. А. Экономически целесообразный уровень теплозащиты зданий при работе систем теплогазоснабжения и вентиляции/ Н. А. Петрикеева, О. В. Тюленева, Н.Н. Кучеров// Научный журнал. Инженерные системы и сооружения. - 2012.- № 1 (6). - С. 9-12.
2. Другалев, А. И. Разработка модуля постановки и контроля задач на платформе 1С : Предприятие 8.3 с применением технологии SMART / А. И. Другалев, Е. А. Копытина // Информатика : проблемы, методология, технологии : материалы 16 Международной научно-методической конференции, Воронеж 11-12 февраля 2016 г. — Воронеж, 2016 . – Секция 5: Прикладное моделирование и E-business. - С. 74-79.
3. Новохатский, Н. М. Разработка мобильного приложения для медицинских клиник «Медангел» / Н. М. Новохатский, А. В. Копытин, Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ. — Воронеж, 2019. – Вып. 13, ч. 1 : Научные работы студентов бакалавриата. – С. 150-155.
4. Карпов, В. Д. Разработка мобильного приложения »Ruparking" / В. Д. Карпов, А. В. Копытин, Е. А. Копытина // Сборник студенческих научных работ факультета компьютерных наук ВГУ. – Воронеж, 2019. — Вып. 13, ч. 1 : Научные работы студентов бакалавриата. - С. 92-97.

Реализация и исследование методов прогнозирования отклика системы на входные воздействия

В. С. Попков

Студент бакалавр

В. А. Степанцов

Доцент

Введение

Каждый день в нашей жизни происходят запланированная и непредсказуемая череда событий, в результате которых получается определенный итог, но что влияет на его формирования? На этот и многие схожие вопросы может ответить понятие воздействие на систему, эффективность воздействия и отклик системы на воздействия. Например, ставится задача оценки влияния рассылки электронной почты на повышение спроса у потребителей интернет-магазина. Построив модель и произведя ряд экспериментов, мы сможем отделить реальный эффект воздействия от внешних причин (шум, влияние неучтенных факторов и т. д.). В результате мы сможем дать более точный прогноз воздействия и подготовить более развёрнутый анализ системы. Подобного рода задачи можно встретить не только в сфере интернет-рекламы, но также и в медицине – оценка эффективности нового разработанного препарата, а также в социологии и политологии – оценка эффективности воздействий (методы пиара кандидата) на избирателей перед выборами.

1. Постановка задачи

Имеется множество объектов, которые мы обозначим через G . Каждый элемент из множества G опишем через набор признаков из множества D . Соответствие между этими множествами реализуем через функцию $\delta: G \rightarrow D$. Тогда для $g \in G$ описание будет представлять кортеж вида:

$$\langle x_1, x_2, x_3, \dots, x_k \rangle, \quad (1)$$

где $x_i, i \in \overline{1, k}$ – это числовые характеристики объекта.

Окажем какое-то воздействие на объект и обозначим множество воздействий через T . Воздействие на конкретный объект обозначим через функцию $t : G \rightarrow T, t_g = t(g)$.

Для исследования влияния воздействий на систему, введём целевой параметр Y , который будет отражать отклик нашей системы. В общем случае целевой параметр Y будем считать функцией, зависящей от: воздействий, характеристик объекта исследования, а также от внешней среды E , в которой система функционирует. Данная зависимость отражена ниже:

$$Y(g, t, E), \quad (2)$$

где $g \in G, t \in T, E$ – внешняя среда.

Тогда исходные данные для решения задачи будут представлены в виде кортежа с набором следующих значений:

$$\langle y_g, t_g, x_1^g, x_2^g, \dots, x_k^g \rangle \quad (3)$$

где y_g – это результат воздействия, t_g – это значение воздействия, $x_i^g, i \in \overline{1, k}$ – это числовые характеристики объекта.

Теперь мы можем сформулировать цель анализа. Целью анализа является вывод о величине эффекта $\Delta Y_g(t)$ от воздействия $t \in T$ для каждого объекта генеральной совокупности $g \in G_0$. Данная зависимость отражена ниже:

$$\Delta Y_g(t) = \tau(g, t). \quad (4)$$

В данном случае под генеральной совокупностью G_0 мы будем понимать потенциальное множество всех объектов, по отношению к которым мы хотели бы принимать решение о том или ином воздействии [1]. Генеральная совокупность G_0 за редким исключением бывает полностью обозримой. В процессе исследования нам доступна только некоторое подмножество $G \subseteq G_0$. Однако выводы анализа случайного слепого исследования мы хотим получить на всю генеральную совокупность G_0 . Под величиной эффекта $\tau(g, t)$ будем понимать разницу в целевом параметре под воздействием и без него:

$$\tau(g, t) = Y_g(t_1) - Y_g(t_0), \quad (5)$$

где $Y_g(t_1)$ – значение целевого параметра с воздействием, $Y_g(t_0)$ – значение целевого параметра без воздействия.

Обратим внимание на то, что подобного рода задача не является тривиальной и её нельзя решить стандартными методами машинного

обучения: классификацией и регрессией. Так как при проведение отдельного взятого эксперимента мы можем получить только один отклик системы от конкретного объекта на воздействие:

$$Y_g^{(\text{exp})} = Y(g, t_g), \quad (6)$$

где t_g – воздействие, оказанное на объект $g \in G$.

Важно понять, что отклик от одного и того же объекта, но с другим воздействием невозможно получить из-за ограничений и свойств самой системы. Данная проблема известна также, как фундаментальная проблема выявления причинности [2]. Например, мы не можем выяснить каким могла бы быть реакция организма человека на плацебо, когда мы уже дали этому человеку препарат или наоборот. Под плацебо понимается некоторое безвредное вещество, приготовленное в виде лекарства, но не обладающее лечебными свойствами и проявляющее свой эффект в результате самовнушения пациента. Поэтому для решения подобного рода проблем выявляют однородные группы объектов, обладающие схожими характеристиками, и высчитывают эффект от воздействия для данных групп.

Таким образом, результаты распределенных эффектов $\tau(g, t)$ заключаются в следующих двух задачах:

- Подсчёт среднего эффекта от воздействия для нашей системы
- Выявление однородных подгрупп

2. Пример исходных данных

Рассмотрим классических пример исходных данных для оценки отклика системы на входные воздействия, представлен в виде рис. 1. Имеется ряд наблюдений, располагающихся в строках, а также ряд известных параметров для каждого из наблюдений: характеристики объекта (пользователя интернет-магазина) воздействия x_1^g, x_2^g, x_3^g , воздействие t_g , а также значение целевого параметра y_g .

Ставится задача оценить, насколько эффективна рекламная email-рассылка на основе представленных исходных данных. Для решения поставленного вопроса рассмотрим основные методы решения задачи оценки отклика системы на входные воздействия.

Id	Количество денег, потраченных за текущий год, в рублях (X1)	Время с последней покупки, в днях (X2)	Пол, мужской/женский (X3)	Наличие включенной Email рассылки, да/нет (T)	Сумма денег, потраченная за последние 2 недели, в рублях (Y)
1	36632	34	мужской	Да	2400
2	12543	12	мужской	Нет	0
...
65678	40213	78	женский	Нет	6000
65679	16217	2	женский	Да	1500

Рис. 1. Пример исходных данных задачи

3. Методы решения

Для решения поставленной задачи мы будем использовать методы, сводящиеся к задаче регрессии. В данной работе рассмотрим методы, которые условно можно разделить на 2 подгруппы:

- методы выделения переменной
- методы трансформации.

Суть методов выделения переменной сводится к преобразованию матрицы исходных данных к определенному виду. С последующим обучением и прогнозированием на основе случайного леса или на основе других методов, решающих задачу регрессии. К данной подгруппе методов можно отнести следующие методы:

- Кадит моделирования.
- Воздействие как признак.
- Эконометрический метод.
- Метод двух моделей.

Методы трансформации является более сложными по сравнению с методами выделения переменной. Их суть уже сводится не просто к преобразованию кортежей исходных данных, а к предварительной классификации объектов с последующим переходом к задаче регрессии. К данной подгруппе относят [3, 4, 5]:

- Метод Лаи.
- Метод Кейна.

- Метод Джасковски.
- Рефлексивный метод.
- Пессимистичный метод.

Классификацию осуществляют по следующим категориям объектов, к примеру, для задачи, представленной в Главе 2:

- CR – постоянные клиенты.
- CN – случайные разовые клиенты.
- TN – клиенты, воздействуя на которых можно легко потерять.
- TR – клиенты, ищущие наибольшей выгоды.

После произведенной классификации методы используют свою уникальную формулу для расчёта эффекта от воздействия.

4. Методики оценки эффекта от воздействия

В качестве метрики оценки отклика системы на входные воздействия рассмотрим средний эффект от воздействия. Суть её заключается в том, что после разделения наших исходных данных на тестовую и обучающую выборку, и после обучения нашей модели на основе какого-то выбранного метода, сводящегося к задаче регрессии, мы получаем подготовленную модель, на которой осуществляем тестирование данных. Эти данные мы упорядочиваем по величине эффекта в двух подгруппах, где воздействие было и где его не было. И выбираем процент α для отбора наилучших наблюдений в двух подгруппах, которые мы будем использовать для подсчета среднего эффекта от воздействия:

$$ATE = \frac{1}{N_\alpha} (y_1(g) - y_0(g)) \quad (7)$$

где N_α – это количество наилучших наблюдений, $y_1(g)$ – это наблюдения, где воздействие было, $y_0(g)$ – это наблюдения, где воздействия не было.

5. Практические результаты.

Сравним результаты различных методов на примере датасета с результатами маркетинговой кампании банка, целью которой было продажа клиентам банка срочных депозитов (денежная инвестиция, хранящаяся в банке в течение определённого периода времени). В качестве воздействия в данной системе можем выбрать звонки, которые совершали работники банка по телефону, или электронные письма, отправляемые на почту клиентам. Целевым параметрам будет выступать бинарная переменная, отвечающая на вопрос: «Согласен ли на срочный депозит клиент банка?». Результаты отражены на рис. 2.

Метод	Модель	MSE	ATE
Джасковски	Случайный лес	0.21	537
Эконометрический	Случайный лес	0.15	612
Воздействие как признак	Логистическая регрессия	0.27	587
Рефлективный	Случайный лес	0.25	499
Пессимистичный метод	Логистическая регрессия	0.36	721

Рис. 2. Таблица сравнения методов

Согласно результатам данной таблицы Пессимистичный метод справился хуже всех из представленных, а Эконометрический показывает довольно хороший результат. Также обратим внимание на то, что модели, основанные на случайном лесе, к которому были сведены методы, показывают лучше результаты, чем модели с логистической регрессией. Это можно объяснить сложностью зависимости целевого параметра и характеристик объекта исследования, которые случайные лес лучше смог определить.

Заключение

В данной работе были рассмотрены методы оценки отклика системы на входные воздействия, была выбрана метрика для оценки величины эффекта от воздействия, а также проведено сравнение методов на реальных данных. Стоит отметить, что различные методы и модели, на базе которых строится прогноз среднего эффекта от воздействия, далеки от идеалов и показывают не всегда хорошие результаты на конкретном датасете. Поэтому для подобного рода задач необходимо проводить множество комбинаций моделей и методов для получения оптимального и достаточно точного результата.

Список литературы

1. Бузмаков, А.В. Машинное обучение для выявления подгрупп индивидов, сильно реагирующих на воздействие [Электронный ресурс]. – Режим доступа: <https://u.to/QcFUGw>
2. Ascarza E. Retention Futility: Targeting High-Risk Customers Might Be Ineffective воздействие [Электронный ресурс]. – Режим доступа: <https://journals.sagepub.com/doi/abs/10.1509/jmr.16.0163>

3. Devriendt, F. A Literature Survey and Experimental Evaluation of the State-of-the-Art in Uplift Modeling: A Stepping Stone Toward the Development of Prescriptive Analytics / F. Devriendt, D. Moldovan, W.Verbeke [Электронный ресурс]. – Режим доступа: <https://www.liebertpub.com/doi/full/10.1089/big.2017.0104>

4. Jaskowski, M. Uplift modeling for clinical trial data / M.Jaskowski, S. Jaroszewicz [Электронный ресурс]. – Режим доступа: http://people.cs.pitt.edu/~milos/icml_clinicaldata_2012/Papers/Oral_Jaroszewicz_ICML_Clinical_2012.pdf

5. Lai LY. Influential marketing: a new direct marketing strategy addressing the existence of voluntary buyers (Doctoral dissertation, School of Computing Science-Simon Fraser University). [Электронный ресурс]. – Режим доступа: <https://summit.sfu.ca/item/6629>

Разработка системы для мониторинга очагов распространения заболеваний

Е. Д. Проскуряков

Студент бакалавр

А. И. Чекмарев

Старший преподаватель

Введение

Последние события в мире показали необходимость вовремя обнаруживать новые заболевания и очаги их распространения, так как это непосредственно влияет на дальнейшую судьбу человечества.

Каждый хотя бы раз за свою жизнь слышал фразу «заболевание, передающееся воздушно-капельным путем». Оно означает, что это заболевание является легко передающимся, а следовательно, одним из самых массовых и быстро распространяющихся. Человек, зараженный заболеванием, относящимся к этой категории, может легко его передавать людям, проживающим вместе с ним в одной квартире, соседям многоквартирного дома, посетителям ближайших магазинов, где этот человек бывает частым покупателем, и тем самым создавать некий очаг распространения заболевания.

Хотелось бы знать, где находятся такие очаги, чтобы можно было своевременно взять ситуацию под контроль и ввести необходимые меры предотвращения распространения заболевания.

1. Цель работы

Целью работы является разработка веб-приложения, которое предназначено для работников медицинских учреждений. К разрабатываемому приложению выдвинуты следующие требования:

- Добавление заболеваний в базу данных и присвоение им своего индивидуального цвета.
- Добавление в базу данных больных с местом проживания и зафиксированным заболеванием.
- Отображение маркеров на карте в тех местах, где проживают заболевшие.
- Возможность отфильтровать отображаемые маркеры по территории и заболеваниям.

– Кластеризация точек при уменьшении масштаба.

2. Актуальность

В последнее время появились сервисы, отображающие статистику случаев заболеваний COVID-19. Одним из самых популярных является сайт «стопкоронавирус.рф» [1], статистика с которого продемонстрирована на рис. 1. Все подобные сервисы отражают только общую картину по одному конкретному заболеванию и не позволяют обнаружить очаги распространения.

Регион	Выявлено	Новые	Активные (7)	Выздоровело	Умерло
Москва	• 1143100	2430	• 93543	• 1030199	• 19358
Санкт-Петербург	• 428917	772	• 23391	• 391682	• 13844
Московская область	• 261513	709	• 41102	• 214395	• 6016
Нижегородская область	• 116248	135	• 1342	• 111475	• 3431
Ростовская область	• 91735	200	• 5355	• 82127	• 4253
Свердловская область	• 88003	107	• 2884	• 81840	• 3279
Воронежская область	• 83075	142	• 1771	• 78513	• 2791
Красноярский край	• 72256	98	• 2035	• 66567	• 3654
Иркутская область	• 68392	117	• 758	• 65253	• 2381
Самарская область	• 63526	113	• 1132	• 60642	• 1752
Архангельская область	• 62934	59	• 2047	• 59923	• 964
Саратовская область	• 61213	115	• 3506	• 56696	• 1011
Челябинская область	• 60678	93	• 2801	• 56334	• 1543
Волгоградская область	• 58610	91	• 1485	• 55905	• 1220
Ханты-Мансийский АО	• 56854	40	• 443	• 55616	• 795

Рис. 1. Статистика заболеваний COVID-19

Необходимость создания подобного сервиса стала очевидна давно. Работники медицинских учреждений города Воронеж обратились на кафедру цифровых технологий факультета компьютерных наук за разработкой сервиса, с помощью которого можно было бы находить очаги распространения туберкулезной инфекции в Воронежской области.

Этот проект поможет работникам медицинских учреждений определять местоположения очагов распространения заболеваний посредством визуального мониторинга карты с нанесенными на нее случаями заражения одним или несколькими заболеваниями в пределах любой территориальной единицы.

3. Архитектура приложения

Данный проект разделен на две архитектурные составляющие:

– Back-end – программно-аппаратная часть, отвечающая за основную логику приложения и реализующая API, по которому происходит взаимодействие с клиентской частью [3].

– Front-end – клиентская часть, реализующая функциональность, с помощью которой пользователь получает доступ к возможностям проекта [2].

Такой вариант архитектуры был выбран для осуществления принципа разделения ответственности между внешним представлением и внутренней реализацией.

Программно-аппаратная часть разделена на модули, каждый из которых отвечает за отделенную функциональность приложения. Наиболее подробное представление продемонстрировано на рис. 2.

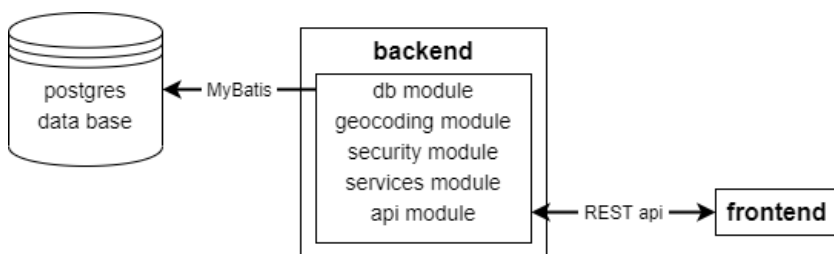


Рис. 2. Архитектура приложения

4. Функциональность приложения и интерфейс пользователя

При первой загрузке главной страницы пользователь автоматически перенаправляется на страницу авторизации (рис. 3), так как полная функциональность приложения доступна только работникам медицинских учреждений. Самостоятельно зарегистрироваться на сайте нельзя, для первой авторизации новый пользователь получает логин и пароль от администраторов.

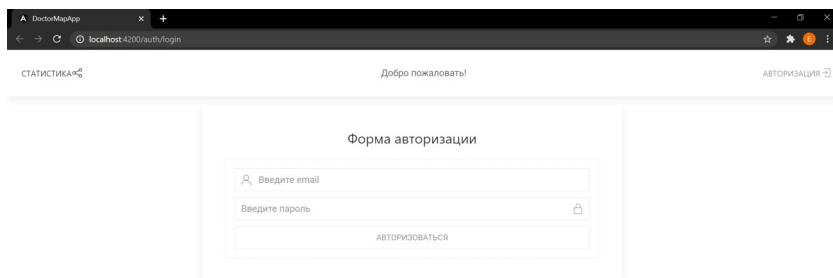


Рис. 3. Страница авторизации

Пройдя авторизацию, пользователь попадает на главную страницу (рис. 4). Отсюда он может перейти в управление заболеваниями и управление больными. На навигационной панели появляются приветствие авторизированного пользователя и кнопки перехода на главную страницу и на страницу с картой.

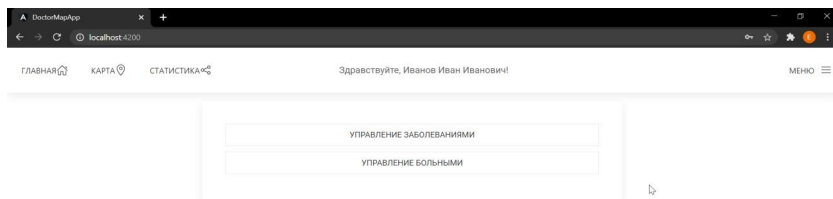


Рис. 4. Главная страница

Перейдя по кнопке «управление заболеваниями», попадаем на страницу (рис. 5), на которой можно увидеть все добавленные в базу данных заболевания, их названия и цвет, которым они будут обозначаться на карте. Здесь реализована возможность добавления нового заболевания, изменения и удаления, внесенного ранее.

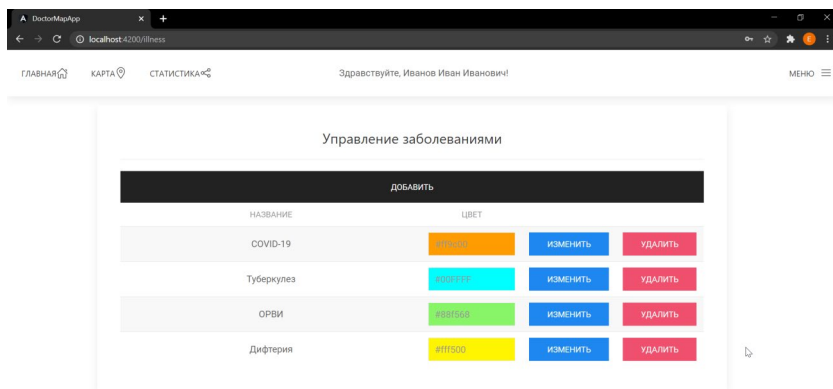


Рис. 5. Страница управления заболеваниями

Перейдя с главной страницы по кнопке «управление больными», попадаем на страницу (рис. 6), на которой можно увидеть фамилию, имя, отчество, адрес проживания и список заболеваний каждого больного, добавленного в базу данных. Здесь также реализована возможность добавления, удаления и изменения любой информации о заболевшем.

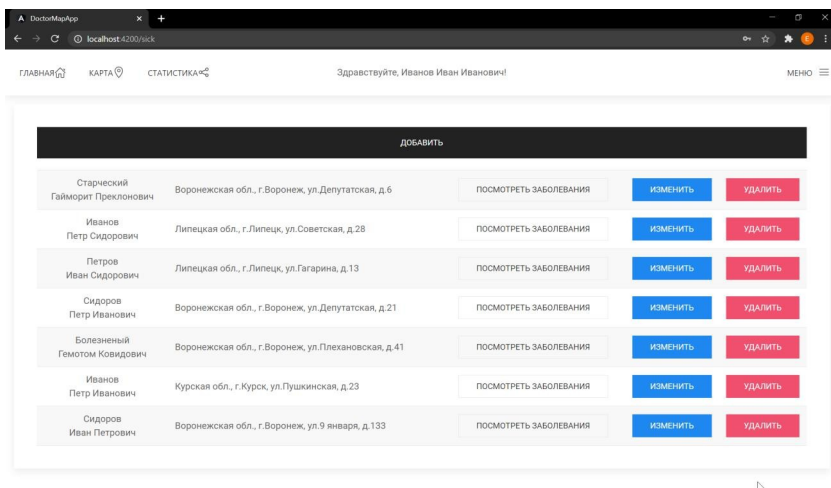


Рис. 6. Страница управления больными

На рис. 7 продемонстрировано диалоговое окно добавления нового больного. В выпадающем списке в качестве подсказки появляются существующие регионы, которые имеют совпадение в названии с тем, что вводит пользователь [4]. Аналогично дополняются остальные поля адреса. Кнопка «добавить» не активна до тех пор, пока не будут заполнены все поля информации.

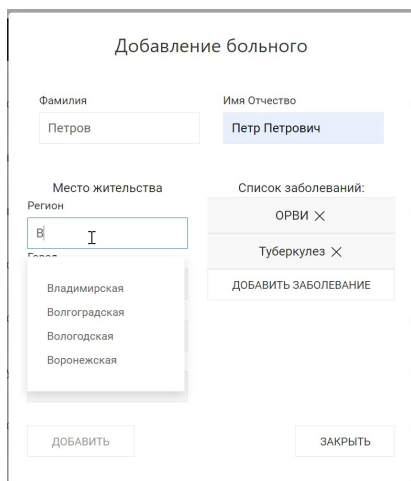


Рис. 7. Диалоговое окно добавления нового больного

На навигационной панели находится кнопка «Карта». Нажав на нее, пользователь попадает на страницу с картой. Эта страница разделена на две карточки, на левой из которых находится форма фильтрации отображаемых больных, а на правой – сама карта [5]. В форме можно установить разные параметры поиска в разных комбинациях. На рис. 8 продемонстрирован вариант с отображением всех больных с одним выбранным заболеванием.

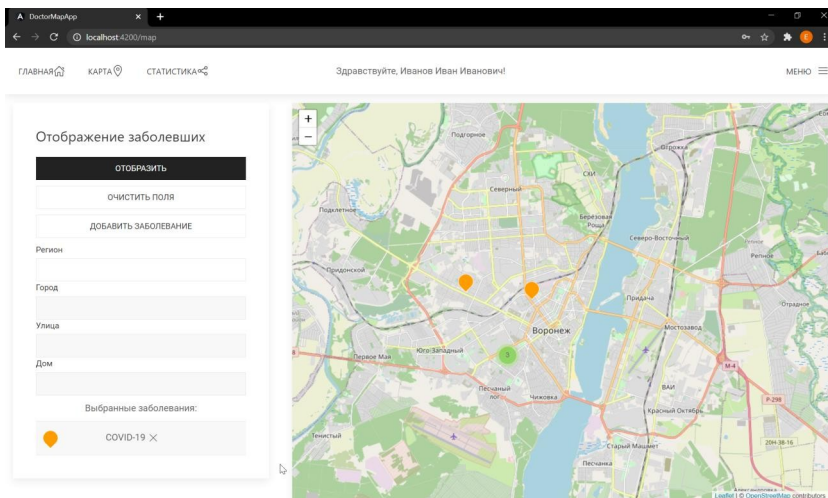


Рис. 8. Отображение больных с одним выбранным заболеванием

Маркеры, которые находятся слишком близко, при уменьшении масштаба кластеризуются и обозначаются как кружок с количеством кластеризованных точек. При выборе какой-либо территориальной единицы карта автоматически переводится в центр выбранной территории и устанавливается оптимальный масштаб. В качестве примера, на рис. 9 отображаются все больные в Воронежской области без уточнения заболевания.

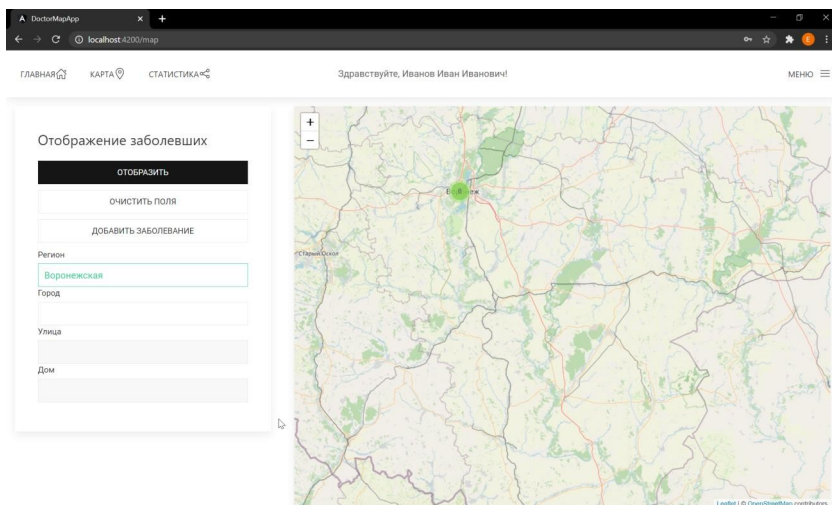


Рис. 9. Отображение всех больных в Воронежской области

Заключение

Было реализовано веб-приложение, которое в полной мере удовлетворяет поставленной цели. Можно смело говорить, что оно востребовано, так как оно запрошено действующими медицинскими работниками, и аналогов до сих пор нет. Приложение удобно в использовании, поскольку обладает интуитивно понятным и ненагруженным интерфейсом, а грамотно выстроенная архитектура и реализация позволяют вносить изменения и дорабатывать приложение без затруднений.

Список литературы

1. Официальный интернет-ресурс для информирования населения по вопросам коронавируса (COVID-19) [Электронный ресурс]. – Режим доступа: <http://стопкоронавирус.рф>
2. Документация по Angular [Электронный ресурс]. – Режим доступа: <https://angular.io/docs>
3. Документация по Spring Boot [Электронный ресурс]. – Режим доступа: <https://spring.io/projects/spring-boot>
4. Документация по работе с KladrAPI [электронный ресурс]. – Режим доступа: <https://kladr-api.ru/docs>
5. Документация по работе с библиотекой Leaflet [Электронный ресурс]. – Режим доступа: <https://leafletjs.com>

Распознавание лиц в условиях ограниченной видимости

М. О. Саввин

Студент бакалавр

Е. Ю. Митрофанова

Доцент

Введение

В настоящее время камеры получили огромное распространение и находят своё применение в решении все большего числа практических задач в различных сферах деятельности человека. Они могут использоваться в системах контроля доступа, в системах слежения и системах детектирования различных объектов, в том числе конкретных людей. Совокупность таких камер генерирует большое количество непрерывных потоков данных, ручная обработка которых требует огромных затрат человеческих ресурсов. Одновременно с этим, современные методы обработки данных и вычислительные мощности позволяют обрабатывать кадры видеопотока практически в реальном времени, поэтому автоматизация таких систем способна повысить их охват и эффективность. Перед разработчиками систем, анализирующих данные с камер, стоит задача получения максимального количества полезной информации из каждого кадра, что на практике может осложняться условиями съемки.

Одной из основных проблем при обработке и анализе видеопотока является проблема ограниченной видимости, которая может возникать при обработке кадров, полученных в плохо освещенных местах или, например, в случае дождя или тумана. Такие ситуации являются обычными для систем слежения, работающих в общественных местах. Для решения этой проблемы возможно применение методов корректировки кадра [1], на основании которых и будет строиться реализация прикладного программного обеспечения, которому посвящена данная работа.

1. Нормализация изображений, полученных при слабой освещенности

Нормализация подразумевает под собой технологию улучшения изображений, которая заключается в усилении контрастности, что

оказывается полезным в случае с темными изображениями. Для проведения нормализации необходимо определить верхний и нижний уровень допустимых значений пикселей в результирующем изображении, либо оставить их по умолчанию, то есть определенными в рамках формата изображения. Обозначим их Max_{out} и Min_{out} . Нормализация описывается следующим образом:

$$P_{out} = (P_{in} - Min_{in}) * \left(\frac{Max_{out} - Min_{out}}{Max_{in} - Min_{in}} \right) + Min_{out} \quad (1)$$

где P_{out} – результирующее значение пикселя, P_{in} – значение пикселя исходного изображения, а Max_{in} и Min_{in} соответствуют максимуму и минимуму исходного изображения. После применения данной формулы мы получаем новое изображение, где интенсивность пикселей больше, чем в темном кадре.

Пример темного изображения и результата нормализации представлен на рис. 1.

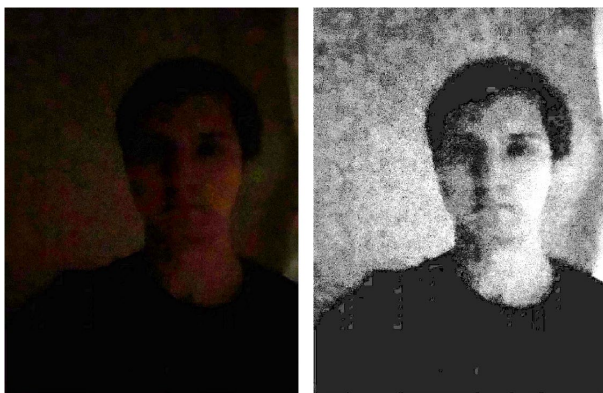


Рис. 1. Пример исходного и нормализованного изображений

2. Гистограммное выравнивание изображений, полученных при слабой освещенности

Гистограммное выравнивание – это метод регулирования интенсивности изображения для увеличения контрастности. Процесс гистограммного выравнивания опирается на применение Кумулятивной функции распределения (CDF). Обозначим через x исходное изображение, а через L общее число уровней интенсивности изображения, тогда значение интенсивности n будет лежать в

промежутке от 0 до $L - 1$. Мы используем такое преобразование $y = T(x)$, чтобы получить новое изображение y , где T описывает кумулятивное распределение x и определяется на $[0, L - 1]$. Формула для гистограммного выравнивания принимает следующий вид:

$$p_y(y) = p_x((T^{-1})y) \frac{d}{dy} ((T^{-1})y), \quad (2)$$

Пример изображения, полученного в результате гистограммного выравнивания, представлен на рис. 2.

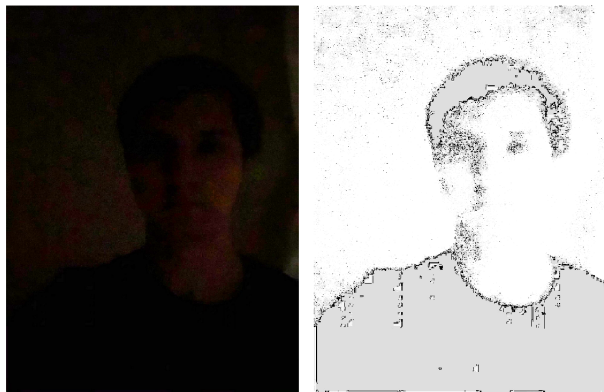


Рис. 2. Пример исходного изображения и результат проведения гистограммного выравнивания

3. Адаптивное гистограммное выравнивание изображений, полученных при слабой освещенности

Адаптивное гистограммное выравнивание – это еще один из подходов к корректировке контраста изображений. Он использует схожий с гистограммным выравниванием механизм, но более эффективен в тех случаях, когда интенсивность на изображении сильно меняется в разных областях. Особенностью данного метода является то, что он преобразует каждый пиксель, основываясь не на интенсивности всего изображения, как в случае с гистограммным выравниванием, а на интенсивности области изображения вокруг этого самого пикселя. Пример изображения после адаптивного гистограммного выравнивания представлен на рис. 3.

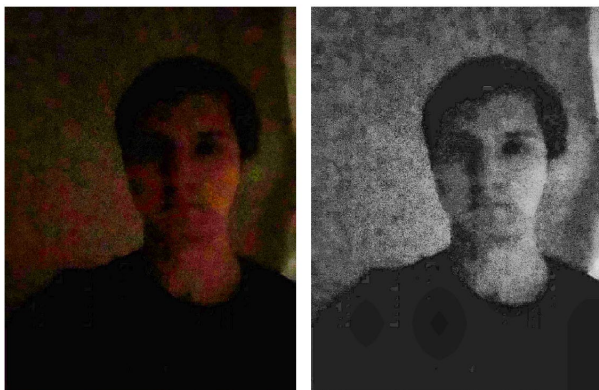


Рис. 3. Пример исходного изображения и результат проведения адаптивного гистограммного выравнивания

4. Детекция лиц в кадре

На полученных промежуточных кадрах лица становятся заметными на общем фоне. Теперь необходимо реализовать функцию детекции лиц. Для этого воспользуемся каскадами Хаара, которые представляют собой обучаемую модель, состоящую из набора масок, являющихся комбинацией черных (цветных) и белых частей [2]. Эти маски поочередно накладываются на разные части изображения для определения факта наличия объекта. Примеры таких масок изображены на рис. 4.

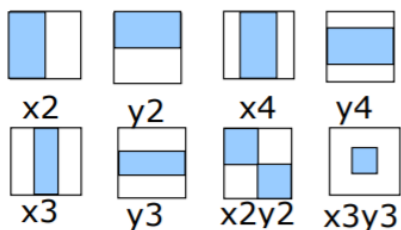


Рис. 4. Маски каскада Хаара

Каскадные модели подходят для детекции различных видов объектов, но нас будет интересовать только каскад для поиска лиц. Для обучения такой модели необходима выборка учебных изображений, состоящая из позитивных примеров лиц и негативных кадров, на которых лица отсутствуют. Полученная модель используется в специальном методе, который реализует поиск объектов на основе

переданного ему каскада. Так как процесс детекции нацелен на выделение областей лиц для их дальнейшей обработки, то, в рамках архитектуры разрабатываемого ПО, было бы логично объединить блок корректировки изображений и блок детекции лиц в один большой модуль, который позволяет получать локализованные лица из кадра. На рис. 5 представлена блок схема такого модуля.

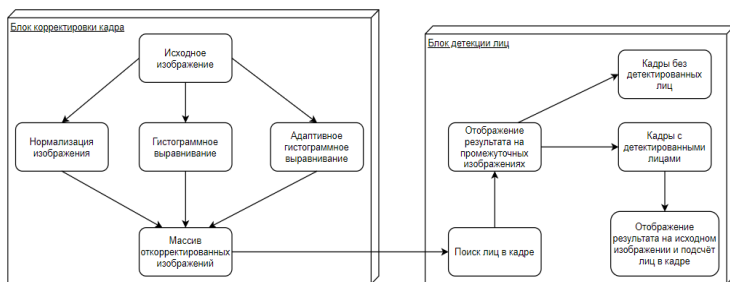


Рис. 5. Блок схема модуля поиска лиц на темных изображениях

Результатом работы модуля будут три промежуточных изображения, полученные в результате разных преобразований, на которых выделены области детектированных лиц, если таковые имеются. Пример результата работы этого модуля представлен на рис. 6.



Рис. 6. Пример откорректированных изображений с выделенными областями лиц

5. Распознавание лиц, детектированных в кадре

На данном этапе работы приложения у нас имеются три изображения, которые содержат найденные области лиц. Проведем отдельно процесс распознавания для каждого из них, чтобы увеличить

шанс получения правильного результата. Для сравнения лиц друг с другом и решения задачи распознавания необходимо представить информацию о свойствах, описывающих лицо конкретного человека, в виде определенного набора значений.

Первым этапом распознавания будет являться поиск ключевых контуров на лице, к которым относятся контур лица, контуры областей глаз, контуры области носа, бровей и губ. Такие контуры можно представить в виде массива точек, которые вместе образуют определенную структуру лица. Для решения задачи поиска таких точек воспользуемся специальным методом, который принимает область изображения с объектом и возвращает набор ключевых точек на основе предобученной модели. На рис. 7 представлен пример отображения точек, описывающих структуру лица.

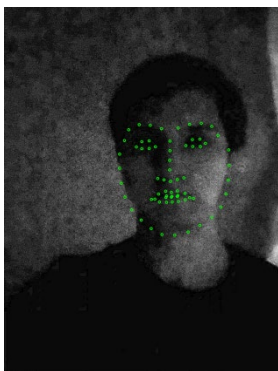


Рис. 7. Пример поиска ключевых точек лица

Для получения единого дескриптора, описывающего конкретное лицо, применяется Residual Network (ResNet) [3] – это остаточная сверточная нейронная сеть для классификации изображений. Она используется для получения из контуров лиц дескрипторы, которые можно использовать для определения степени схожести образов. В базе данных лиц находятся пары, содержащие имя человека и соответствующий ему дескриптор. В процессе распознавания дескрипторы лиц из кадра поочередно сравниваются с дескрипторами из базы данных, лица считаются одинаковыми, если имеет место достаточная степень соответствия дескрипторов. Пример результирующего изображения представлен на рис. 8.

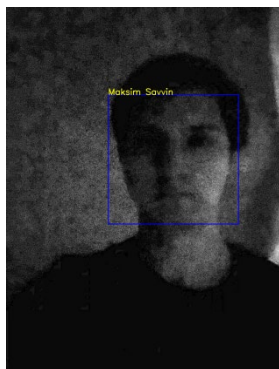


Рис. 8. Результат распознавания лиц на темном изображении

Заключение

Данная статья посвящена разработке и реализации программного модуля для поиска и распознавания лиц на кадрах, полученных в условиях ограниченной видимости. Описаны алгоритмы корректировки и улучшения кадра для корректной локализации лиц на изображениях. В итоге процесса данной разработки получился полностью рабочий и переносимый программный модуль, реализованный в виде тестового приложения.

Список литературы

1. Human face detection in excessive dark image by using contrast stretching, histogram equalization and adaptive equalization / Allayear, M. [и др.] [Электронный ресурс]. – Режим доступа : https://www.researchgate.net/publication/330006071_Human_Face_Detection_in_Excessive_Dark_Image_by_Using_Contrast_Stretching_Histogram_Equalization_and_Adaptive_Equalization
2. Viola, P. Robust Real-Time Face Detection / Jones, M. J. [Электронный ресурс]: – Режим доступа: <https://www.face-rec.org/algorithms/boosting-ensemble/16981346.pdf>
3. Deep Residual Learning for Image Recognition / Kaiming He [и др.] [Электронный ресурс]: – Режим доступа: <https://arxiv.org/pdf/1512.03385.pdf>

Синтез и анализ алгоритмов генерации музыки на основе текстовых данных

Е. А. Стеблева

Студент бакалавр

А. Ю. Иванков

Доцент

Введение

В настоящее время новые цифровые технологии, в том числе и искусственный интеллект, дополняют или даже заменяют работу человека во многих не только технических сферах, но и в сферах искусства (изобразительного, архитектурного и др.). Что же касается музыкальной сферы, то уже существует множество алгоритмов генерации музыки с осмысленным текстом и вокалом, а также без них. Однако, если в изобразительном искусстве на данный момент существуют написанные нейронными сетями картины, которые сложно или практически невозможно отличить от картин современных художников, то в музыкальной сфере таких достижений на сегодняшний день значительно меньше.

Данная статья посвящена синтезу и анализу алгоритмов генерации музыки, обработке текстовых данных на основе нейронных сетей, а также созданию программного обеспечения для звукового сопровождения читаемого текста в режиме реального времени. Основная идея использования именно нейронных сетей заключается в том, чтобы разработанное программное обеспечение могло использовать не уже готовые музыкальные композиции, написанные человеком, а генерировало новые. Вместе с тем могло сразу анализировать и выявлять тональность вводимого пользователем текста.

1. Предобработка данных и формирование выборки музыкальных композиций

В качестве данных для обучения было решено использовать фрагменты из аудиоспектаклей и аудиопостановок, где есть фоновая музыка. Для выявления таких частей было разработано специальное приложение, блок-схема алгоритма которого представлена на рис. 1. Суть алгоритма заключается в том, чтобы выбирать из аудиоспектаклей

такие отрезки, в которых амплитуда сигнала была бы больше заданной определенной продолжительности времени.

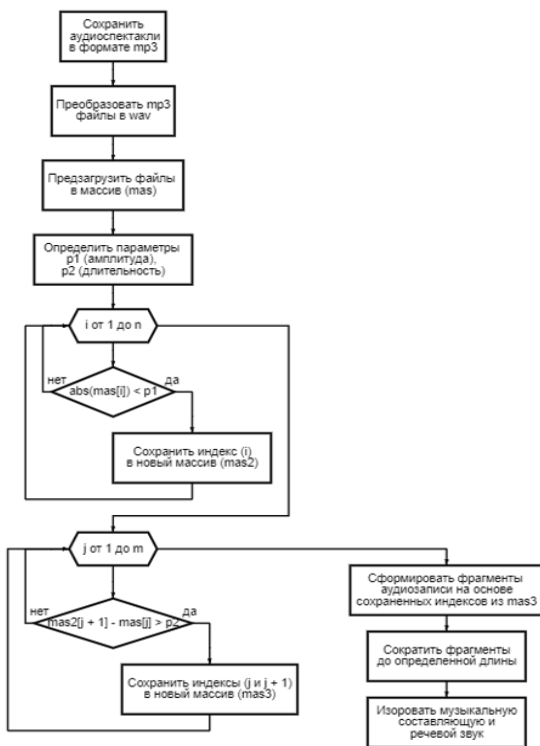


Рис. 1. Блок-схема алгоритма

Для наглядности на рис. 2 представлен график определенного фрагмента аудиофайла. Здесь амплитуда меньше заданной на отрезке $[i_0, i_{100}]$ и $[i_{101}, i_n]$. Из этих индексов формируется первый массив. Затем из него выбираются те индексы, расстояние (или в смысле музыки длительность) между которыми, начиная со 2 через 1, больше заданной минимальной длительности. На рисунке это индексы j_0, j_1 . Низкие амплитуды здесь – это те места, где диктор вздыхает или просто молчит. А если на фоне играет музыка, то таких амплитуд долгое время встретить нельзя.

После выявления нужных аудиофрагментов их них изолировалась речь, которая в дальнейшем распознавалась с помощью библиотеки SpeechRecognition.

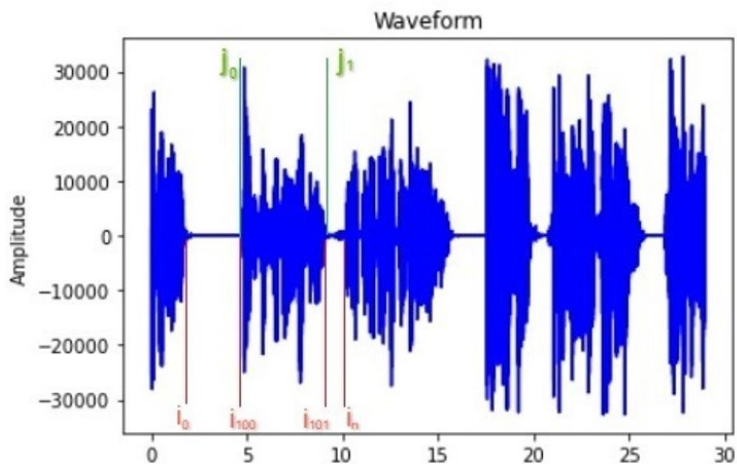


Рис. 2. График аудиофайла

2. Анализ алгоритмов выявления тональности текста

Для выявления тональности распознанного текста было принято решение проанализировать два существующих решения. Первое – это модель нейронной сети Sentiment Analysis of Tweets in Russian, а второе – модель GPT-3.

В первой модели используется CNN [1] архитектура сети, которая представлена на рис. 3. Для начала предложения токенизируются и преобразовывают в матрицу предложений, которая используется в качестве входной матрицы сети. Каждая строка здесь – это векторное представление каждого токена. В данном случае это результаты Word2Vec модель [2]. А в качестве данных для обучения используются заранее размеченные на негативные и позитивные твиты. F-мера такой модели составляет 78.14 %.

Во второй модели используется архитектура сети трансформер, которая состоит из кодировщика и декодировщика. При разработке программного обеспечения использовалась модель `gpt-3 Large` [3], в которой 760 миллионов параметров, длина контекста – 2048 и размер обучающего датасета – 600 ГБ. Идея считывания тональности текста заключается в том, чтобы давать на вход модели заранее распознанный

из аудиофрагментов текст и добавлять в конце предложение «Настроение мое было».

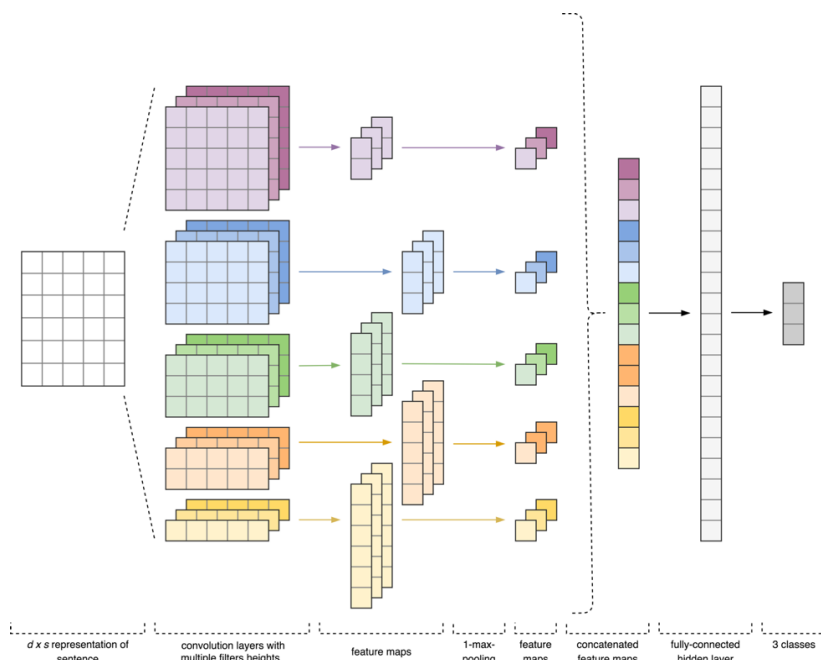


Рис. 3. CNN архитектура сети

Затем считывать то, как модель продолжит это предложение и выбирать оттуда прилагательное. Была использована библиотека синонимов для того, чтобы объединить все прилагательные в несколько больших категорий. Также была разработана сама программа, выбирающая из предложений прилагательные, если они там есть.

3. Анализ алгоритмов генерации музыки

Для того, чтобы сгенерировать пул из музыкальных композиций того или иного настроения, было принято решение проанализировать следующие модели: MusicGenerator и JukeBox от OpenAI.

Первая модель основана на LSTM сетях, архитектура которой представлена на рис. 4. Принцип работы здесь похож на принцип генерации текста, только вместо букв ноты.

Layer (type)	Output Shape	Param #
lstm_1 (LSTM)	(None, 100, 512)	1052672
dropout_1 (Dropout)	(None, 100, 512)	0
lstm_2 (LSTM)	(None, 100, 512)	2099200
dropout_2 (Dropout)	(None, 100, 512)	0
lstm_3 (LSTM)	(None, 512)	2099200
dense_1 (Dense)	(None, 256)	131328
dropout_3 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 456)	117192
activation_1 (Activation)	(None, 456)	0
Total params: 5,499,592		
Trainable params: 5,499,592		
Non-trainable params: 0		

Рис. 4. LSTM архитектура сети

Вторая модель основана на автокодировщиках. Архитектура представлена на рис. 5. Здесь мы пытаемся реконструировать то, что подали на вход нашей сети, в моем случае это аудиофрагмент. На рисунке представлены 3 разных по размеру автокодировщика, которые мы тренируем одновременно. Разница состоит в том, как мы разделяем полученный сигнал. Очевидно, что самый нижний даст нам наиболее полный на входные данные результат и будет «знать» о небольших деталях композиции, тогда как самый верхний будет иметь представление о чем-то, что длится продолжительнее, например, структура куплета или припева.

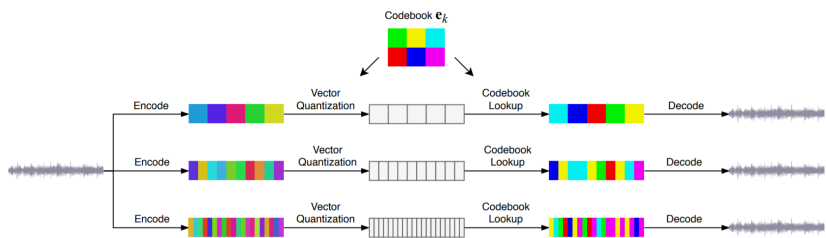


Рис. 5. Архитектура сети, основанная на автокодировщиках

Для улучшения генерации было принято решение совместить их в один автокодировщик так, как представлено на рис 6. Т.е. на вхд последующего автокодировщика с менее частым разделением данных, мы подаем результаты предыдущего с более частым разделением данных. Тем самым на последнем, третьем, уровне модель будет иметь представление как о маленьких, так и о больших деталях.

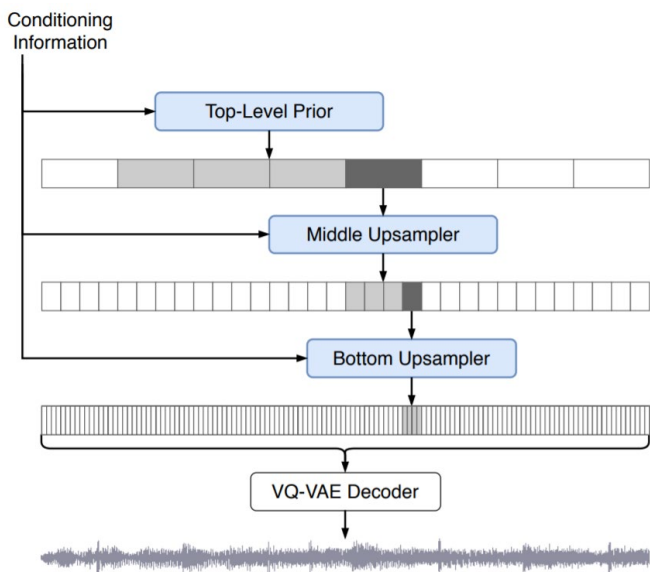


Рис. 6. Алгоритм работы совмещенного в один автокодировщика

Для того, чтобы генерировать музыку большей длительности, чем исходная, на вход каждой итерации алгоритма подается лишь некоторая часть от того, что получилось на предыдущем шаге.

4. Тестирование разработанного программного модуля

После того, как пользователь вводит текст какого-либо произведения в разработанное приложение, полученные данные делятся на блоки по одному предложению. После этого поочередно подаются на вход GPT-3 для выявления их тональностей. Если тональности соседних предложений одинаковы, то они объединяются в один блок. На экране последовательно будут показаны выявленные блоки предложений и на фоне каждого будет играть соответствующая настроению фоновая музыка.

Заключение

В результате данного исследования были проанализированы различные модели нейронных сетей для генерации музыки и выявления тональности текста. Полученные композиции в модели MusicGenerator имеют некую схожесть с тем, что создают люди, однако главным недостатком является излишняя монотонность, а также возможность генерации музыки только в формате MIDI. В модели JukeBox результаты уже более похожи на реально создаваемые человеком, тем не менее в композициях присутствует большое количество шумов.

Список литературы

1. Sentiment Analysis of Tweets in Russian [Электронный ресурс]. – Режим доступа : <https://github.com/sismetanin/sentiment-analysis-of-tweets-in-russian>
2. Zhang, Y. A Sensitivity Analysis of Convolutional Neural Networks for Sentence Classification [Электронный ресурс]. – Режим доступа : <https://arxiv.org/pdf/1510.03820.pdf>
3. ruGPT – 3 [Электронный ресурс]. – Режим доступа : <https://sbercloud.ru/ru/warp/gpt-3>

Разработка алгоритмов парсинга для информационной системы анализа и визуализации данных маркетплейса Wildberries

К. А. Турченко

Студент бакалавр

Д. В. Черницын

Ассистент

Введение

Сегодня все больше людей используют маркетплейсы для покупки товаров, например, Wildberries или Ozon. Из-за растущей популярности у покупателей продавцы все чаще используют данные сервисы для продажи товаров. Для профессиональных участников рынка маркетплейсов (продавцы, маркетинговые агентства, частные производители товаров) очень важно понимать, какие товары, бренды или категории товаров востребованы сейчас, а какие будут популярны в ближайшем будущем. Для того, чтобы заинтересованные лица могли найти такую информацию и воспользоваться ею, было принято решение создать информационную систему обработки и визуализации данных.

Пользователи данной системы смогут наблюдать, какие товары становятся более популярными, какие наоборот, падают в рейтинге. Таким образом, поставщики и другие заинтересованные лица смогут избежать потерь денежных средств при планировании своей деятельности на маркетплейсе Wildberries.

Очень важной является задача автоматического получения данных с интернет ресурсов для дальнейшей обработки. На данный момент на Wildberries продаются около 10 миллионов товаров в различных категориях. Крайне важно своевременно получать информацию обо всех товарах и вести статистику цен, продаж, рейтинга товаров и другой полезной информации для продавцов.

Для решения такой задачи используют алгоритмы парсинга. Они собирают всю необходимую для нас информацию с интернет ресурсов в базу данных для дальнейшей обработки.

1. Выбор инструмента разработки

Для реализации будем использовать язык программирования Python [1] и его библиотеки и фреймворки. Одними из наиболее популярных инструментов для разработки алгоритмов парсинга веб-ресурсов являются:

- re – регулярные выражения
- BeautifulSoup, lxml – две наиболее популярных библиотеки для парсинга html
- Scrapy [2] – open-source framework.

Наиболее мощным инструментом является Scrapy framework. В нем есть возможность асинхронных запросов, что для нашего проекта крайне важно, так как нам необходимо собирать большие объемы данных ежедневно. В Scrapy также можно использовать XPath и CSS селекторы, удобно работать с различными кодировками и еще много полезных и удобных функций. Нужно отметить, что Scrapy поддерживает экспорт данных в различные форматы, например, CSV, XML, JSON, JSON Line [3].

2. Реализация

На Wildberries каждый товар имеет уникальный числовой идентификатор. Его можно найти на странице товара:

Xsport.online / Перчатки спортивные

Артикул 27602528 ☆☆☆☆☆ 0 отзывов

Рис. 1. Пример артикула на карточке товара

либо внутри url-адреса страницы:

<https://www.wildberries.ru/catalog/27602528/detail.aspx>

Первоочередная задача – это сделать скрипт для добавления всех идентификаторов всех товаров в базу данных, при этом нам также необходимо сохранять некоторую дополнительную информацию о товаре (табл. 1).

Список полей для первого парсера

Название	Описание
ID	Уникальный идентификатор с помощью которого строятся ссылки (артикул)
Название	Наименование товара
Бренд	Бренд товара
Картинки	Ссылки на картинки товара
Цвет	(если есть)
Размеры	Размеры (если есть)
Продавец	Юридическое/физическое лицо
Дата обнаружения	Первая дата когда обнаружили товар
Дата первого отзыва	Дата первого отзыва

Всю представленную в табл. 1 информацию мы можем получить на странице с товаром.

В Scrapy скрипты для парсинга называют пауками(spiders). Для начала создадим в пакете spiders нового паука categories_spider.py. Это скрипт будет проходить по всему дереву категорий и подкатегорий товаров на Wildberries и сохранять адрес крайних подкатегорий. Сохранять данные будем в таблицу categories_urls. После работы скрипта таблица будет выглядеть так:

	id [PK] bigint	url character varying (1000)	items integer
1	7828	https://www.wildberries.ru/catalog/yuvelirnye-ukrasheniya/broschi	3211
2	7829	https://www.wildberries.ru/catalog/yuvelirnye-izdeliya/tsepi-i-shnurki	12304
3	7830	https://www.wildberries.ru/catalog/yuvelirnye-ukrasheniya/serebro	117870
4	7831	https://www.wildberries.ru/catalog/yuvelirnye-ukrasheniya/ukrasheniya-iz-keramiki	242
5	7832	https://www.wildberries.ru/catalog/yuvelirnye-izdeliya/aksessuary-dlya-ukrasheniya	111
6	7833	https://www.wildberries.ru/catalog/yuvelirnye-izdeliya/suvenirny-i-stolovoe-serebro	3185
7	7834	https://www.wildberries.ru/catalog/yuvelirnye-ukrasheniya/chetki	52
8	7835	https://www.wildberries.ru/catalog/yuvelirnye-ukrasheniya/plirsing	392

Рис. 2. Фрагмент таблицы categories_urls

Теперь у нас есть список url-адресов всех категорий товаров и мы можем для каждой категории получить все товары. Для этого создадим нового паука `main_goods_spider.py` в пакете `spiders`. Всю полученную информацию будем собирать в таблицу `goods_tmp`. После выполнения скрипта в таблице было около 10 000 000 товаров. Часть таблицы изображена на следующем скриншоте:

```

1 SELECT * FROM public.goods_tmp
2 ORDER BY autoId ASC LIMIT 100
3

```

Data Output

id	image	name	color	page	size	appear_date	first_review_date	brand_name	brand_url	seller_name
integer	image character varying (400)	name character varying (400)	color character varying (255)	page character varying (50)	size character varying (50)	timestamp (with time zone)	timestamp without time zone	character varying (255)	character varying (1000)	character varying (500)
19366114	/image...	Шушкова GrandCHEF	серебристый	[null]	0	2021-05-02	[null]	Tescoma	/brands/tescoma	ПРОФТОРП ООО
18557023	/image...	Худи Relaxed Graphic Hoodie	синий	[null]	6S,M,L,XL,XXL	2021-05-02	2021-04-08 09:54:24	Levi's®	/brands/levis	ВАЙЛДБЕРРИЗ ООО
13512441	/image...	Шушкова 40 см	серебристый	[null]	0	2021-05-02	2020-08-11 14:25:49	Папета	/brands/papeta	Мелочное Дмитрий Влад...
8456376	/image...	Шушкова для ванны, 46см	серебристый	[null]	0	2021-05-02	2019-07-08 16:20:39	Самарская пельца	/brands/samarskayapelca	Петренко Валентина Вячес...
14028241	/image...	Шушкова и лопка для каза...	серебристый	[null]	0	2021-05-02	2020-09-09 19:17:28	PROMONOVA	/brands/promonova	СЕРВИС ТЕХНОЛОГИЯС ООО
8733423	/image...	Толстовка	черный	[null]	6S,M,L,XL,XXL	2021-05-02	2019-09-20 15:09:47	HUGO	/brands/hugo	ХЮГО КАСС Рус ООО
12453409	/image...	Штателъ кондитерский стат...	серебристый	[null]	0	2021-05-02	[null]	FARTUK	/brands/fartuk	Миргалеев Замона Рабба...
25132505	/image...	Штателъ кондитерский с ра...	стальной голубой	[null]	0	2021-05-02	[null]	Sugar Day	/brands/sugar-day	ИНТЕРНЕТ РЕВОЛЮЦИЯ О...
19556735	/image...	Штателъ кондитерский	серебристый	[null]	0	2021-05-02	[null]	inStyle Home	/brands/instyle-home	Индивидуальной предпри...
19150749	/image...	Штателъ кондитерский / На...	металл	[null]	0	2021-05-02	2020-12-25 18:47:11	Osagart	/brands/osagart	Буденская Елена Виталье...
17946186	/image...	Толстовка	темный хаки	[null]	6S,M,L,XL,XXL	2021-05-02	2019-02-20 16:41:14	TBOE	/brands/tboe	ТБое ООО
10771137	/image...	Нож кондитерский 11х3 см	серебристый	[null]	0	2021-05-02	[null]	Ra-shef	/brands/ra-shef	Сидурев Виталий Израил...
18330075	/image...	Штателъ кондитерский 3 шт...	розовый	[null]	0	2021-05-02	2020-12-25 19:03:07	Kitchen Queen	/brands/kitchen-queen	05 000
17106618	/image...	Штателъ лопатка для взго...	[null]	[null]	0	2021-05-02	[null]	FENIX	/brands/fenix	Исков Рамен Хамовцев ИП
26246816	/image...	Штателъ кондитерский	белый	[null]	0	2021-05-02	2021-04-22 19:03:11	Good stuff	/brands/good-stuff	ИП "Корнеев К.А."
51104253	/image...	Толстовка на меху / Плядо...	черный, серый метал...	[null]	0XS,S,M,L,XL,XXL,X...	2021-05-02	2017-12-01 18:44:58	Envy Lab	/brands/envy-lab	Царегородцев Антон Олего...
25754879	/image...	Набор штателъ кондитерс...	[null]	[null]	0	2021-05-02	2021-04-20 05:17:42	ASTYLE HOME	/brands/astyle-home	Пергаев Андрей Алексан...

Рис. 3. Фрагмент таблицы `goods_tmp`

Теперь когда в базе данных есть `id` всех товаров, мы можем внутри скрипта формировать url-адрес для перехода непосредственно на страницу товара. Это ускорит работу нашего следующего паука, который будет ежедневно парсить динамические данные товаров.

Добавим новый парсер `stock_spider.py` в пакет `spiders`. Как было сказано выше, этот скрипт будет собирать динамическую информацию о товарах. Эти данные приведены в табл. 2.

В этом списке стоит отдельно сказать о расчете остатков товара на складе. Серверы Wildberries отправляют клиенту информацию об остатках только тогда, когда товар добавлен в корзину. Это сделано для того, чтобы исключить ситуацию, когда покупатель добавил слишком много товаров в корзину (больше, чем есть на складе в данный момент). Скрипт `stock_spider.py` во время работы использует разные пользовательские сессии. В Scrapy для этого используется мета-параметр `cookie_jar` у объекта `Request`. Это было сделано для того, чтобы обойти ограничение Wildberries: не более 256 товаров в корзине.

Список полей для второго парсера

Название	Описание
Цена конечная	Цена с общей скидкой
Цена без скидки	Цена без скидки
Количество отзывов	Количество отзывов
Рейтинг	Среднее число рейтинга
Количество вопросов	Количество вопросов
Количество покупок	Рассчитывается количество покупок на основе поля “куплено более раз” на странице товара
Остатки	Остатки товара на складах
Дата	Дата загрузки информации

В ходе разработки было выявлено еще одно ограничение Wildberries. При отправке параллельных запросов на добавление товаров в корзину, часть товаров не была в корзине. Поэтому был сделан вывод, что серверы Wildberries не могут корректно обработать параллельные запросы в рамках одной корзины пользователя.

Для решения этой проблемы запретим Scrapy использовать параллельные запросы. Установим `CONCURRENT_REQUESTS = 1` в файле `settings.py`. Также будем использовать `multiprocessing` в Python, который позволит нам использовать несколько процессоров на компьютере.

Результаты парсинга будут добавлены в таблицы `stock` и `goods_stat` (рис. 4, рис. 5).

```

1 SELECT * FROM public.stock
2 ORDER BY id ASC LIMIT 100
3

```

Data Output

	id [PK] bigint	goods_id integer	date timestamp without time zone	size character varying (300)	quantity integer
1	111	16487818	2021-02-01 00:00:00	37	2
2	112	15480654	2021-02-01 00:00:00	43	2
3	113	15480654	2021-02-01 00:00:00	42	1
4	114	16487818	2021-02-01 00:00:00	39	2
5	115	16509193	2021-02-01 00:00:00	39	1
6	116	16487818	2021-02-01 00:00:00	40	1
7	117	16487818	2021-02-01 00:00:00	38	2
8	118	11036617	2021-02-01 00:00:00	37	1
9	119	11036617	2021-02-01 00:00:00	39	1
10	120	15480654	2021-02-01 00:00:00	44	3
11	121	11119645	2021-02-01 00:00:00	40	13
12	122	15480654	2021-02-01 00:00:00	45	2
13	123	15480654	2021-02-01 00:00:00	46	3
14	124	16209470	2021-02-01 00:00:00	37	9

Рис. 4. Фрагмента таблицы stock

```

1 SELECT * FROM public.goods_stat
2 ORDER BY id ASC LIMIT 100
3

```

Data Output

	id [PK] bigint	goods_id integer	date timestamp v	price integer	discounted_price integer	sales bigint	revenue bigint	reviews integer	questions integer	answers integer	rating smallint
1	11	17936080	2021-02-0...	850	363	80	29040	7	0	[null]	5
2	12	10602752	2021-02-0...	35580	15655	1	15655	[null]	2	[null]	[null]
3	13	14168052	2021-02-0...	3599	1774	600	1064400	32	11	[null]	4
4	14	10464931	2021-02-0...	9220	3743	1500	5614500	154	79	[null]	4
5	15	15291706	2021-02-0...	3181	1351	100	135100	3	1	[null]	3
6	16	11226958	2021-02-0...	2445	2371	500	1185500	28	14	[null]	5

Рис. 5. Фрагмента таблицы goods_stat

3. Результаты разработки

Было разработано расширение для браузера Google Chrome, которое встраивает график с данным в карточку товара на сайте wildberries.

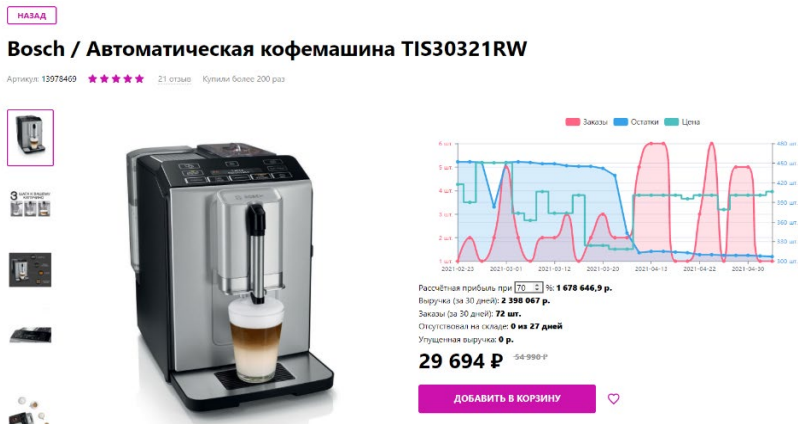


Рис. 6. Готовое расширение для браузера Google Chrome

Пользователю доступна информация о заказах, остатках и цене. Расширение позволяет выбирать любое количество графиков для отображения. Также расширение выводит некоторую сводную информацию, например, выручку, общее количество заказов за период. У пользователя есть возможность посчитать прибыль и изменить процент.

Заключение

В результате проделанной работы были разработаны алгоритмы для сбора информации о товарах на маркетплейсе Wildberries. Скрипты успешно работают на удаленном сервере и ежедневно считывают и добавляют в базу данных большие объемы данных. Дополнительно планируется добавить возможность парсинга более популярных товаров чаще чем раз в сутки, а менее приоритетных – реже.

Также в планах реализовать сбор информации с других маркетплейсов таких как Ozon и ЯндексМаркет.

Список литературы

1. Python3.8 documentation [Электронный ресурс] : база данных – Режим доступа : <https://docs.python.org/3.8/>
2. Scrapy documentation [Электронный ресурс] : база данных –
3. Режим доступа : <https://docs.scrapy.org/en/latest/intro/tutorial.html>
4. Code Complete: A practical handbook of software construction /
5. Steve McConnell. – Second Edition. – Microsoft Press 2004 – 896 с.

Какие проекты интересны школьникам на примере учеников Яндекс.Лицея

Ю. В. Шишко

Студент бакалавр

Д. И. Соломатин

Старший преподаватель

Введение

Ни для кого не секрет, что мастерство в любом искусстве, в том числе в программировании, приходит с практикой и, соответственно, со временем. Чем дольше мы занимаемся чем-то, тем больше проникаемся философией этого дела, тем больше аспектов и тонкостей познаем. Именно поэтому так важно начать программировать со школьного возраста.

К сожалению, сейчас во многих школах программирование преподают так, что дети просто не могут им заинтересоваться. Часто изучаются либо устаревшие языки, не имеющие применения в реальной практике, либо на программирование уделяется так мало времени, что дети не успевают им заинтересоваться. Причем это не всегда вина преподавателей – увы, они обязаны следовать учебному плану и подстраиваться под «среднего ученика», и даже при большом желании от этого никуда не деться.

Именно поэтому сейчас так остро стоит вопрос дополнительного образования школьников. В отличие от многих других предметов, изучить программирование только по книжкам нельзя – всегда нужна практика, нередко пояснения со стороны более опытных людей. Детям же еще во многом важна социальная составляющая – коллектив, общество других таких же заинтересованных ребят, с которыми можно посоревноваться и обсудить задачи.

Проблема также состоит в том, что объем материала в сети на тему программирования измеряется терабайтами. Именно поэтому так важна методика, которая позволит дополнительному обучению всего за два года обучить детей с нуля создавать проекты, которые востребованы на рынке труда.

1. Методика Яндекс.Лицей

Именно такую методику реализует Яндекс.Лицей. Обучение длится два года, учеба представляет собой два занятия в неделю по полтора часа и возможность консультаций с преподавателями. Ребята проходят отбор, позволяющий выявить достаточно мотивированных детей, помимо этого тестируется логическое мышление, а не умение программировать. Детям предоставлена площадка, содержащая методические материалы и задачи на каждую тему. Ссылка на веб-сайт площадки: (<https://yandexlyceum.ru/>). Далее разберем подробнее каждый год обучения.

2. Первый год обучения

Первый год обучения направлен на изучение основ Python и парадигмы объектно-ориентированного программирования. Первые полгода ребята обучаются с нуля строить примитивные алгоритмы, реализовывать это с помощью встроенных структур данных. Изучаются базовые типы данных, методы работы с ними, коллекции, циклы, условия и другие необходимые конструкции.

Второй семестр представляет собой изучение основ объектно-ориентированного программирования и библиотек Python. Задача состоит в доступном объяснении, что такое и зачем нужна модульность; как работают функции и как их писать; почему спагетти-код – это плохо, и что иногда стоит пожертвовать краткостью кода в угоду читаемости. Уже здесь они впервые пробуют участвовать в проектной деятельности и поймут, какой код читается легче, а какой становится непонятен собственному же создателю через неделю-другую.

Важную часть здесь занимает изучение принципов объектно-ориентированного программирования и обучение учеников основам проектирования и разработки классов – следующая ступень в важнейшем базисе для любого программиста [1]. Ребята учатся не просто создавать классы и связывать их, но и рассматривать несколько альтернатив. Так, например, при разработке системы для игры в шахматы рассматривались пять различных вариантов, из которых был выбран оптимальный.

3. Второй год обучения

Второй год полностью посвящен проектной деятельности. За год ребята познакомятся с PyQt, PyGame, Flask, Map-API, научатся делать навыки для голосового помощника Алисы, создавать своих ботов и даже изучат основы асинхронного программирования. Программа очень насыщенная, и в учебнике, в отличие от первого курса, содержится далеко не вся необходимая информация. Дети учатся работать с

документацией, искать необходимое и, конечно же, могут консультироваться с преподавателем в случае, если что-то непонятно.

4. PyQt

В первом проекте ребятам предоставляется возможность создать приложение на PyQt с использованием баз данных. Что же такое PyQt? Для начала разберемся, что такое Qt. Это написанная на C++ библиотека с классами для создания графического интерфейса. Библиотека получилась настолько удачной, что начала собирать вокруг себя большое сообщество программистов, которые разрабатывали приложения не только на C++, но и на других языках программирования. Это привело к тому, что и для других языков программирования стали появляться свои библиотеки-«обертки» для QT. Для Python это PyQt [2].

Это все – необходимая база, чтобы в дальнейшем понимать принцип работы десктопных приложений, уметь проектировать и создавать базы данных, а также умело пользоваться github для командной разработки. В течение двух месяцев они изучают PyQt, QtDesigner – программу для создания графических интерфейсов, учатся работать с исключениями и создавать свои собственные, разбираются с файлами в Python и учатся работать с ними. Дальше – изучают диалоговые окна, работу с изображениями и простые таблицы – текстовые файлы в формате csv. После этого они знакомятся с реляционными базами данных, в нашем случае SQLite, учатся работать с SQL-таблицами и отображать данные в них. Наконец, учатся обрабатывать события нажатий клавиатуры и мыши, и, в конце концов, учатся собирать свои первые независимые приложения.

Одним из таких проектов, реализованных ребятами, стали классические шахматы. Ребята не только полностью спроектировали и реализовали непростую архитектуру для игры, но также подключили искусственный интеллект для возможности играть с противниками разной сложности. Более того, они добавили различные шахматные задачи, в которых можно попрактиковаться. Все задачи хранились в базе данных в определенной нотации и при необходимости могли добавляться и редактироваться. Интерфейс получившегося приложения можно увидеть на рис. 1. Ознакомиться с приложением можно по ссылке: <https://yadi.sk/d/SYkse-DofHbWyw>

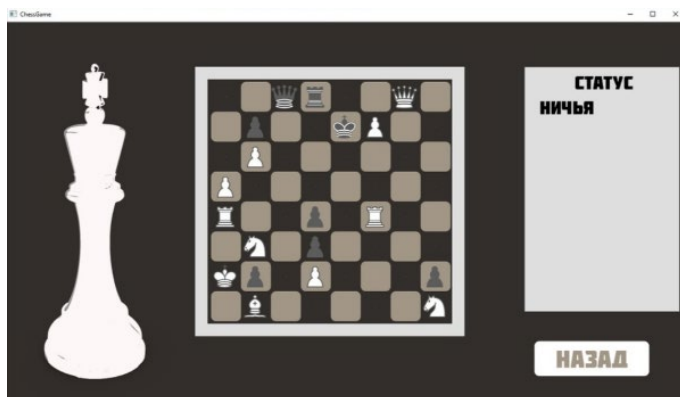


Рис. 1. Проект «Шахматы»

5. PyGame

Вторым проектом, над которым ребятам предстояло потрудиться, стала игра на PyGame – библиотеке для разработки игр. Здесь ученикам пришлось познакомиться с игровым циклом, событиями, вспомнить разработку игр на клетчатом поле, поработать с основами компьютерной графики и физики в играх [3]. Столкновения, взаимодействия предметов, игра в целом, спрайты и их анимация – встроенными способами и «вручную». Помимо этого, они основательно обучились взаимодействию с github и вновь попробовали проектную деятельность.

Здесь у ребят было, где разгуляться, поэтому игры были самые разные: от сапера до 2-D шутеров с видом сверху. Одной из самых интересных таких игр как раз оказался «Radiant Defense». Это игра жанра «Tower Defense», суть которой заключается в защите базы от идущих по тропе врагов, которых атакуют построенные вами башни. Ссылка на приложение: https://github.com/GGewe4/y1_radiant_defense.git

В игре реализован уровень «Лес Эльфов» с 4 типами башен и с 4 типами противников. Каждый из типов башен и врагов обладает разными характеристиками и свойствами, что вносит в игру разнообразие. Постройка башен отнимает деньги, убийство монстров – добавляет. Помимо этого, реализована механика потери жизней и проигрыша после того, как их количество опустится до нуля. При создании игры ученики вдохновлялись игрой «Kingdom Rush». Они взяли у нее стиль графики и спрайтов, черты баланса и геймплея, что хорошо видно на рис. 2.

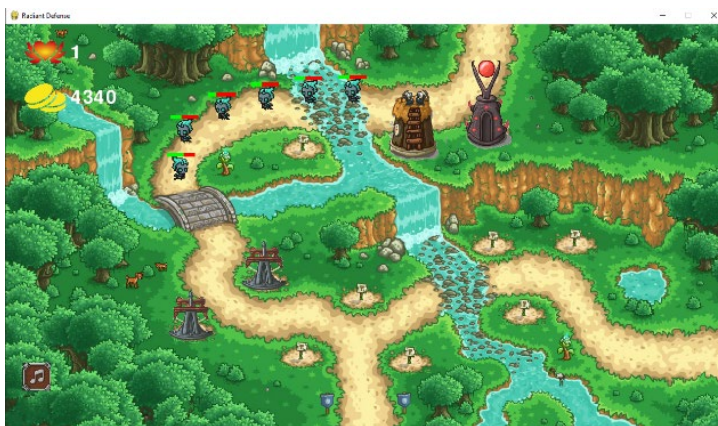


Рис. 2. Скриншот игры «Radiant Defence»

6. Клиент-серверное приложение на Flask

Второй семестр второго года начинается с изучения работы с файловыми системами. Проводится небольшой экскурс в операционные системы, изучаются основы работы с командной строкой и создают свои собственные скрипты. Дети учатся использовать базовые консольные утилиты и «ходить» по системе не только через графический интерфейс. Здесь же начинают изучаться API. Начинают с малого – что это, зачем нужно и как использовать. Учатся пользоваться уже созданными – как, например, API Яндекс.Карт. Все это нужно для того, чтобы ребята как можно лучше познакомились с этой структурой перед тем, как пробовать писать собственные [4].

И только после изучения всего этого можно было приступить к веб-разработке. Последним и самым большим по объему и длительности стал проект на Flask. Ребятам предстояло написать клиент-серверное приложение с использованием rest-api, развернуть и запустить его на облачном сервисе и презентовать свой проект. Для этого они изучали flask, учились обработке html-форм. Изучали шаблоны, а также познакомились с orm sqlalchemy – библиотекой, позволяющей работать с базой данных через объекты языка. После этого учатся создавать свои первые rest-api: вручную обрабатывая запросы и с помощью библиотеки Flask-restful, создав объектно-ориентированную оболочку для использования ресурсов. Помимо этого, прекрасно понимая, что в процессе разработки их модели данных для хранения информации будут меняться, в учебную программу были включены инструменты

миграции, позволяющие не пересоздавать базу данных раз за разом, а правильно редактировать ее.

Одним из таких интересных проектов стал «PriceGame» – сервис для поиска лучшей цены на игру, включая скидки. Помимо бэкэнда и верстки, ребята так же проделали серьезный путь для создания базы данных для работы приложения. Для этого им понадобилось создать парсер и обработать большое количество ответов от API Steam, известной платформы для игр. Ссылка на исходный код проекта: <https://github.com/GagikProger/pricegame>. Дизайн приложения продемонстрирован на рис. 3.

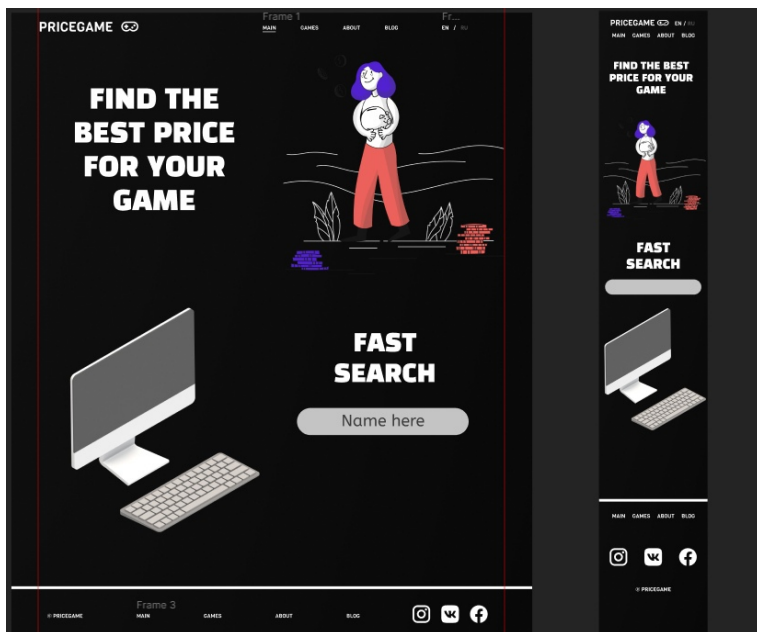


Рис. 3. Дизайн приложения на Flask

Заключение

Подводя итог, хочется заметить, что детям нравится не просто проектная деятельность, а именно возможность выразить себя и продемонстрировать собственную индивидуальность – именно поэтому второй год, несмотря на свою сложность, идет у ребят куда охотнее первого. Программирование дает огромный плацдарм для творчества, именно поэтому так важно учиться ему со школьного возраста.

Список литературы

1. Вирт, Н. Алгоритмы и структуры данных / Н. Вирт. – М.: Мир, 2016. – 360 с.
2. Гергель, В.П. Современные языки и технологии параллельного программирования: Учебник / В.П. Гергель. – М.: МГУ, 2016. – 408 с.
3. Голицына, О. Л. Языки программирования: Учебное пособие / О. Л. Голицына, Т. Л. Партыка, И.И. Попов. – М.: Форум, НИЦ ИНФРА-М, 2017. – 400 с.
4. Плас, Дж. Python для сложных задач. Наука о данных и машинное обучение / Дж. Вандер Плас. – М.: Питер, 2017. – 518 с.

Труды студентов магистратуры

Проверка подлинности цифровых изображений

Т. К. Аюпян
Студент магистрант
А. Ю. Иванков
Доцент

Введение

В настоящее время благодаря различным программам обработки изображений очень просто модифицировать цифровое изображение. В данной статье рассматривается способ проверки подлинности цифрового изображений.

Дипфейк (от англ. «deep learning» – глубокое обучение и «fake» – фальшивый) – реалистичная манипуляция аудио и видеоматериалами с помощью искусственного интеллекта. Эта технология заставляет говорить человека то, что он не произносил, и делать то, что он никогда не делал. В основе дипфейков лежат генеративно-сопоставительные нейросети (GAN) [1]. Это алгоритмы на базе машинного обучения, способные генерировать новый контент из заданного набора. Часто обычному пользователю сложно отличить смонтированное видео от настоящего и определить замены фрагментов изображений. Первый дипфейк создан при помощи открытых алгоритмов машинного обучения и библиотек TensorFlow и Keras. TensorFlow была разработана компанией Google и создана для обучения нейронной сети, чтобы та достигла человеческого качества восприятия. Основное API [2] для работы с библиотекой создано для языка программирования Python. Часто дипфейк может обмануть не только обычного зрителя, но и компьютер. Теперь легко можно дискредитировать любого человека. Распространение ложной информации, вторжение в частную жизнь, разрушение репутации – и это только маленькая часть того, к чему могут привести дипфейки. Целью работы является синтез и анализ алгоритмов глубокого обучения в интересах определения подлинности цифровых изображений и видео.

1. Описание архитектуры генеративно-сопоставительные нейросети

Генеративно-сопоставительные нейронные сети (Generative Adversarial Nets, GAN) алгоритм машинного обучения без учителя, построенный на комбинации из двух нейронных сетей. Наряду с

вариационным автокодировщиком данный тип нейросетевых архитектур используется в качестве одного из современных методов обучения без учителя (Unsupervised Learning). Разработка эффективных методов обучения без учителя является одной из ключевых проблем в области глубинного обучения, так как подготовка размеченных данных является достаточно трудоемкой и дорогостоящей процедурой.

Классической архитектурой GAN (рис. 1) является модель с двумя модулями: генератором и дискриминатором. Задача генератора заключается в преобразовании (генерации) случайного вектора z (зачастую из равномерного распределения) в вектор более высокой размерности, соответствующий размерности реальных данных. При этом в процессе обучения модель генератора стремится подобрать внутренние параметры так, чтобы выход был «похож» на примеры из имеющегося набора данных. Критерий похожести определяется другим модулем – дискриминатором. Дискриминатор в процессе обучения старается подобрать внутренние параметры так, чтобы он мог отличить данные из реального набора от данных, созданных генератором.

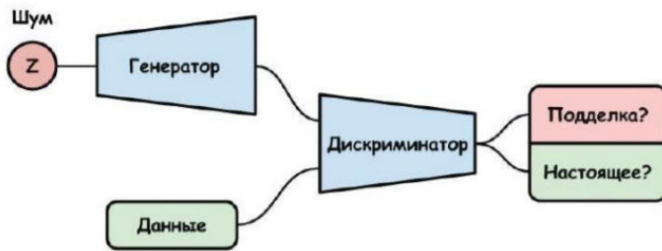


Рис. 1. Общая архитектура GAN

При обучении GAN каждый из модулей конкурирует с другим в процессе минимизации совместной функции ошибки $V(D, G)$, которую можно формализовать как минимум следующим образом [3]:

$$\min_G \max_D V(D, G) = E_{p_{data}(x)} \log(D(x)) + E_{z \sim p_z(z)} \log(1 - D(G(z)))$$

где D – модель дискриминатора; G – модель генератора; x – реальные данные, набираемые из априорного распределения $p_{data}(x)$, z – случайный низкоразмерный вектор, формируемый из случайного равномерного распределения $p_z(z)$.

2. Реализация нейронной сети для синтеза и редактирования лиц

Реализации системы машинного обучения на основе генеративно-состязательной нейронной сети (GAN), позволяет синтезировать и редактировать изображения лиц людей. Сеть была обучена на коллекции лиц реальных людей CelebA. Система позволяет синтезировать новые лица или изменять произвольные фотографии на основе задаваемых характеристик, таких как пол.

Модель генерирующей сети GAN, использованная для решения задачи, приведена ниже (рис. 2).

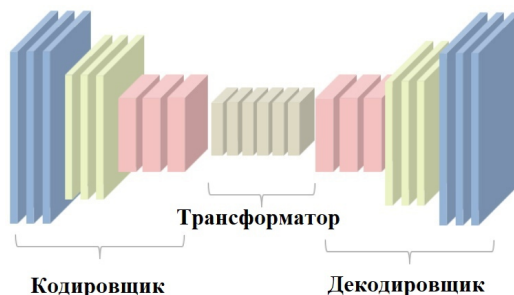


Рис. 2. Генерирующая сеть GAN

Генерирующая сеть состоит из трех частей: Кодировщика, который кодирует характерное представление изображения, трансформатора, состоящего из серии сверточных остаточных блоков и декодера, которая повышает дискретизацию преобразованных функций и выводит окончательное сгенерированное изображение. Начальное количество фильтров определяет количество фильтров для первого сверточного слоя, последующие свертки используют удвоенное количество фильтров предыдущей свертки.

Дискриминатор проверяет участки входного изображения и определяет вероятность того, что эти участки являются настоящими или поддельными (рис. 3).



Рис. 3. Дискриминатор GAN

Пример преобразования на основе характеристики пола приведен ниже (рис. 4).



Рис. 4. Преобразование изображения

Заключение

Генеративно – состязательные модели – быстро развивающаяся область исследований. Они являются одним из наиболее перспективных разработок в области глубокого обучения. GANs—это генеративные сети, которые не так давно приобрели популярность в сфере машинного обучения. Комбинируя характерные особенности входных данных, они создают принципиально новые объекты, неотличимые от реальных. За последние несколько лет генеративные нейросети привлекли внимание и нашли практическое применение. Совершенствуя эти модели, расширяя возможности обучения и увеличивая наборы данных, мы, вероятно, сможем создать образцы, отображающие полностью правдоподобные изображения или видео. Такие модели могут найти самые разные применения. GANы способны работать с многомерными данными. К примеру, они могут создавать реалистичные изображения, используя характерные особенности тренировочных датасетов.

Таким образом, с помощью генеративно – состязательной модели, было выполнено преобразование изображения, на основе характеристики пола.

Список литературы

1. Генеративно-состязательная нейросеть (GAN). – Режим доступа : <https://neurohive.io/ru/osnovy-data-science/gan-rukovodstvo-dlja-novichkov/>
2. Генеративные модели от OpenAI. – Режим доступа : <https://habr.com/ru/company/wunderfund/blog/334568/>
3. PGGAN – прогрессивная генеративная нейросеть от Nvidia. – Режим доступа: <https://neurohive.io/ru/papers/pggan-progressivnaja-generativnajaneyroset-ot-nvidia/>

Организация исследования предметной области и проектирования сервиса учета нетрудоспособности на примере цифровой медицинской платформы Numedy

Е. Г. Аникин

Студент магистр

А. А. Вахгин

Доцент

Введение

В результате развития технологий, главным образом в последние 10-20 лет, на основании цифровых технологических платформ появились новые экономические системы, существование ранее которых было технически невозможно. Трансформировались целые отрасли хозяйствования, в которых принципиально изменилось взаимодействие потребителя и производителя, и произошел значительный экономический рост. Цифровые технологические платформыкратно повышают качество товаров и услуг, значимо экономят время потребителя и производителя и сильно снижают себестоимость и цену реализуемых продуктов [1].

Появление успешных цифровых технологических платформ актуально во многих отраслях, а распространенность мобильных устройств и интернета, а также и запросы пациентов по улучшению качества и доступности и снижению стоимости медицинских услуг обуславливают неизбежность создания цифровых медицинских платформ.

Numedy (numedy.com) – яркий пример цифровой медицинской платформы, т.к. реализованы все три ключевых принципа платформы: множественность участников в одной среде с фокусом на потребителе, единое информационное пространство, контроль и обратная связь в реальном времени.

Numedy – это комплекс программ и сервисов для медицины, предоставляемых клиникам по принципу SaaS. Платформа является одновременно системой управления медицинским предприятием (интегрированной с медоборудованием), атласом (личным кабинетом) пациента и банком (обезличенных) медицинских данных. В настоящий

момент в базе платформы более 500 тыс. пациентов и более 150 млн. медицинских показателей и измерений. Разрабатывается с 2013г. на базе сети медицинских центров МедЭксперт (med-exp.ru) в России. Однако амбиции Numedy не ограничиваются территорией РФ, и функционал проектируется и реализуется с учетом возможности работы на международном уровне, с учетом множественности стран и языков, это реализуемо, т.к. в отличие от других сфер организм человека и данные о его здоровье стандартизирован на уровне Всемирной организации здравоохранения, которая отвечает за внедрение единых справочников и структур медицинских данных в мире.

Логика бизнес-процессов медицинских компаний крайне сложная, прежде всего ввиду сложности человеческого организма, вследствие чего использование стандартных подходов, как, например, «передать разработку подрядчику» и «сделайте, чтобы было хорошо» исчерпало себя очень быстро. Для организации «непрерывности» разработки проекта с учетом множественности специализированных узких предметных областей в медицине необходимо было реализовать «массовые» их исследования, а также изменить этапы постановки технических заданий [2].

В данной статье затронута тема практической организации исследования предметной области и проектирования сервисов платформы Numedy на примере учета нетрудоспособности.

Для реализации процессов исследования предметной области и проектирования в организационной структуре компании выделено специальное подразделение в департаменте разработки – отдел исследований и проектирования, подготовку материалов осуществляет группа аналитиков. Задачей отдела и аналитиков является изучение юридических, экономических, медицинских, математических, логических, обычаев практического оборота и других составляющих бизнес-процессов, итогом которого является предложение реализации продукта в виде интегрированного в цифровую платформу функционала или сервиса. Такое предложение оформляется в виде технического задания для разработчиков компании, которое, как правило, состоит из трех основных блоков – общее описание предмета, структуру базы данных и описание алгоритма, а также графический дизайн.

1. Исследование предметной области учета нетрудоспособности

Отделом исследования и проектирования был проведен анализ данной области и бизнес-процессов, которые осуществляются для практической реализации учета нетрудоспособности разных видов и в различных странах.

В результате были выявлены ключевые характеристики и признаки видов нетрудоспособности и действий акторов процессов. Учет нетрудоспособности осуществляется сжатым образом в разных странах, отличия касаются сроков, оплаты и некоторых других особенностей. Общие характеристики для всех стран, в т.ч. России, было решено реализовать в ядре платформы, а страновые особенности вынести в отдельные микросервисы.

Так общими для всех являются:

1. Виды нетрудоспособности.

Как правило, в странах выделяют:

- нетрудоспособность самого пациента в связи с заболеванием;
- нетрудоспособность здорового человека в связи с уходом за ребенком;
- нетрудоспособность женщины в связи с беременностью и родами.

Для учета видов нетрудоспособности решено предусмотреть такую сущность в проектируемой базе данных сервиса.

1. Периоды нетрудоспособности. Выделяются разные периоды нетрудоспособности, которые устанавливаются (освидетельствуются) в разные дни (при разных медицинских осмотрах).

2. Одну нетрудоспособность (одного пациента) для разных периодов устанавливают разные акторы (медицинские работники), а также одновременно их множество (с участием врачебной комиссии).

3. Нетрудоспособность удостоверяется специальными медицинскими документами, которые подписываются ответственными лицами, в том числе с использованием электронной подписи (в РФ это алгоритм по ГОСТ Р 34.10-2012).

4. Отпуск в связи с нетрудоспособностью может оплачиваться (в США, например, законом нет установленной обязанности работодателя его оплачивать, но может быть частью «сопакета» для сотрудников) работодателем, страховой компанией или специальными государственными фондами для чего требуется предоставить им информацию, как правило, в виде электронного документа.

5. Прочее. Хранимая и/или передаваемая информация является специальными персональными данными, правила обработки которых могут быть очень строгими (в РФ для оператора требуется наличие лицензий ФСБ и ФСТЭК).

Примерно такое, приведенное в этом подразделе статьи, содержание и имеет первый блок технического задания.

2. Проектирование базы данных, алгоритма, графического дизайна и подготовка технического задания

Для учета характеристик, указанных в 1-м подразделе статьи подготовлена структура базы данных, представленная на рис. 1.

Для разработки применяются разные инструменты, например, MS Visio, Draw.io, или, как в данном случае MySQL Workbench. Инструмент выбран просто с точки зрения удобства проектирования, и только для диаграммы, т.к. СУБД применяемая для работы ядра платформы – PostgreSQL.

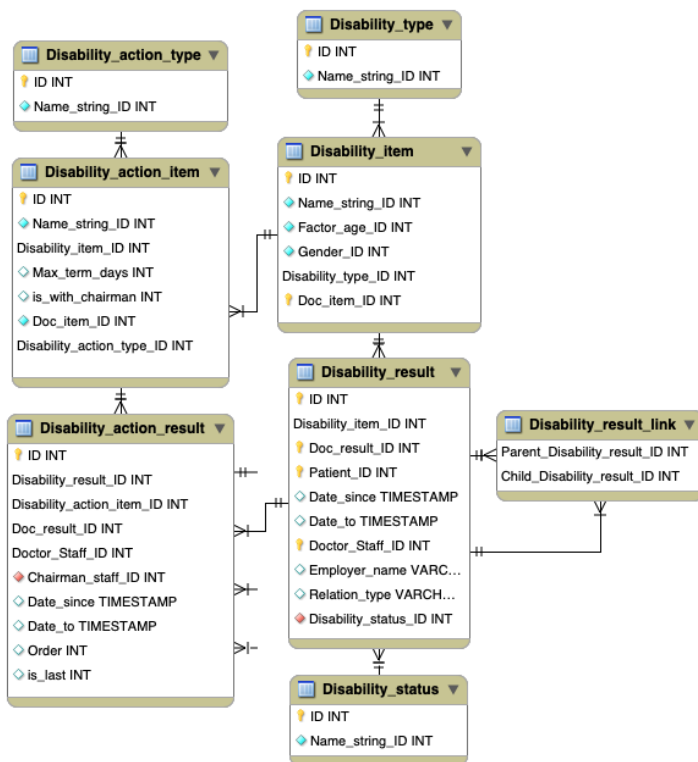


Рис. 1. ER-диаграмма сервиса учета нетрудоспособности

Описание некоторых таблиц.

Disability-item – вид нетрудоспособности. Пример для РФ приведен в таблице.

Виды нетрудоспособности

Название (Name в Disability-item)	Примечание
Беременность и роды при одноплодной беременности	Оформляется листок нетрудоспособности сроком 140 дней, по основному месту работы, без ухаживающего
Заболевание ребенка или учащегося	Заболевшему оформляется Справка по форме 095/у
Заболевание работающего пациента старше 15 лет	Листок нетрудоспособности, по основному месту работы, с причиной нетрудоспособности «Заболевание», сроком до 10 месяцев
Уход работающим за ребенком младше 15 лет при его заболевании	Листок нетрудоспособности оформляется для ухаживающего за заболевшим ребенком лица, с причиной нетрудоспособности «Уход за больным членом семьи», по основному месту работы

Disability-action-type – типы «действий» с нетрудоспособностью (Открыть, Продлить, Закрыть, Аннулировать).

Работа алгоритма описывается в ТЗ любым удобным способом, так, чтобы разработчикам было понятно. В практике Numedy применяются, как правило, описание в виде текста, рисунков, диаграмм, используется некоторые (не все) виды диаграмм UML, исходя из логики «необходимо и достаточно», пример на рис. 2 [3].

Тут одновременно описывается взаимосвязь проектируемого сервиса с другими подсистемами платформы.

Одновременно существенным блоком технического задания является интерфейс работы пользователей. В нашем случае это врач и председатель врачебной комиссии, а также пациент в личном кабинете (приложении), пример на рис 3.

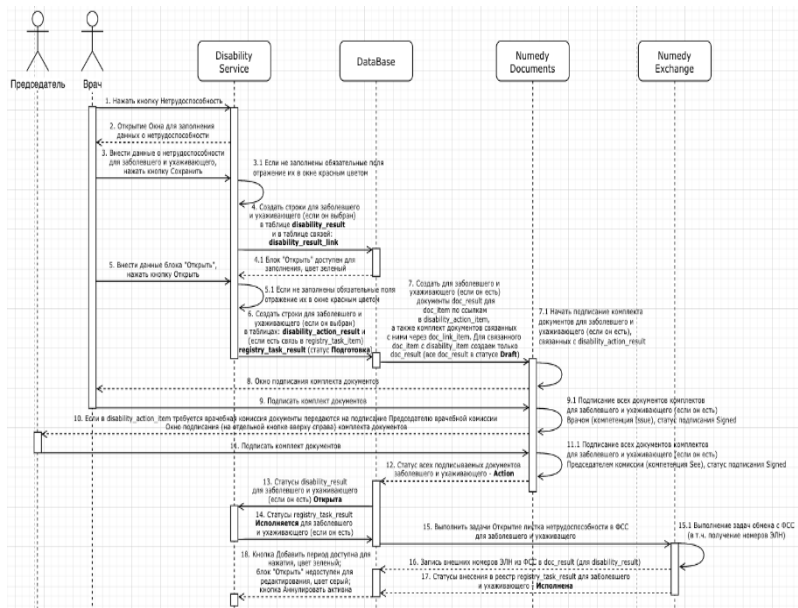


Рис. 2. UML-диаграмма описания работы алгоритма

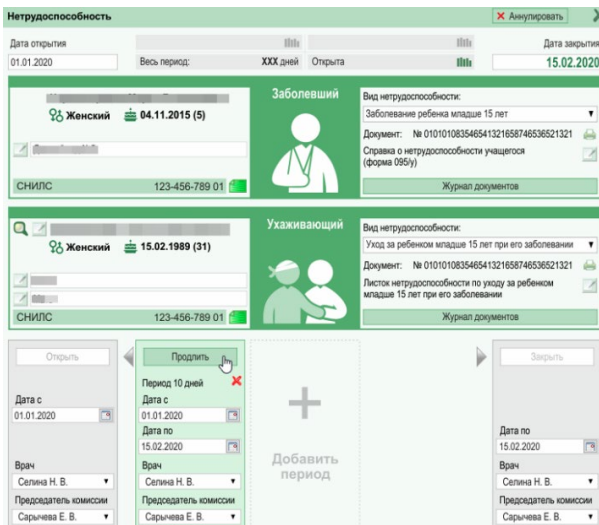


Рис. 3. Графический дизайн работы сервиса нетрудоспособность

Результатом экспериментов по организации исследования и проектирования (изучались разные методики и модели) стала работа аналитиков (не ИТ-специалистов), их задачи:

- исследование предметной области – в ТЗ остаются только краткая нужная информация;
- структура базы данных и алгоритм работы сервисов платформы (без технических деталей);
- разработка дизайна интерфейса (с участием дизайнеров).

Группа программистов на основе такого технического задания дорабатывает приемы или инструменты реализации.

Заключение

Целью данной статьи является описание практической реализации декомпозиции процессов исследования предметной области и проектирования, когда разработчикам предоставляются проработанные технические задания, и им не нужно погружаться в ненужные детали (в нашем случае приказов Минздрава РФ, практик бизнес-процессов или изучения полномочий врача и председателя врачебной комиссии). Им остается проработать технические детали, как то применение каких-то решений, настроек или библиотек. Такая организация работы позволяет структурировать работу ИТ-специалистов (разработчики любят «кодить»), равномерно планировать их загрузку, ну и, учитывая общий дефицит разработчиков на рынке труда, и вследствие чего значительный рост стоимости их труда, получать осязаемый экономический эффект.

Список литературы

1. Гелисханов, И. З. Цифровые платформы в экономике: сущность, модели, тенденции развития / И. З. Гелисханов, Т. Н. Юдина, А. В. Бабкин // Научно-технические ведомости СПбГПУ. Экономические науки. – 2018. – Т. 11. – № 6. – С. 22-36.
2. Купер, А. Алан Купер об интерфейсе. Основы проектирования взаимодействия. / А. Купер, Р. Рейман, Д. Кронин. – Пер. с англ. – СПб.: Символ-Плюс, 2009. – 688 с.
3. Коцюба, И. Ю. Основы проектирования информационных систем. Учебное пособие. / И. Ю. Коцюба, А. В. Чунаев, А. Н. Шиков. – СПб: Университет ИТМО, 2015. – 206 с.

Учет SoD-рисков при разработке ролей в системе SAP R/3

С. Ю. Архипов

Студент магистр

И. В. Илларионов

Доцент

Введение

В настоящее время наблюдается все возрастающий интерес к авторизации как средству обеспечения риск-менеджмента в информационных системах. Как это ни парадоксально, но авторизация сама может быть источником уязвимостей, частным случаем которых является SoD-риск.

Под SoD-риском (от англ. *separation of duties*) понимается присвоение одной роли/одному пользователем определенного набора полномочий, позволяющего выполнить полностью некоторый критичный процесс и за счет этого сознательно или по ошибке сфальсифицировать данные [1].

Превентивный учет SoD-рисков при разработке ролей и администрировании пользователей позволяет минимизировать риск мошенничества или непреднамеренных ошибок в соответствующей системе и, как правило, связан с меньшими затратами, чем ликвидация негативных последствий реализовавшего SoD-риска.

В фокусе рассмотрения в нашем исследовании находятся SoD-риски, характерные для модуля MM, т.е. модуля управления материальными потоками предприятия, в ERP-системе SAP R/3. Цель работы – проектирование и разработка ролей для модуля MM с учетом выявленных SoD-рисков. Модуль MM включает 2 компонента: управления закупками товаров и услуг (MM-PUR) и управление инвентаризацией материальных запасов предприятия (MM-IM). Ввиду ограниченного объема публикации в настоящей статье рассмотрена разработка ролей с учетом SoD-рисков только для первого компонента модуля MM – процесса закупки товаров и услуг на предприятии.

1. Матрица SoD-рисков для MM-PUR

Основная задача первого этапа исследования заключалась в выявлении SoD-рисков, потенциально возможных в модуле MM, и

составлении их технических дефиниций. Для решения данной задачи выбран метод матрицы SoD-рисков, поскольку последний позволяет создать устойчивую модель SoD-рисков, не требующую ревизии после каждого изменения / создания ролей, а также позволяет идентифицировать SoD-риски, локализованные внутри одной роли.

В основе данного метода лежит понимание SoD-риска как комбинации 2 конфликтующих функций. Под функцией понимается отдельная задача, выполняемая пользователем в системе в рамках некоторого бизнес-процесса (например, «создание/изменение основных данных кредитора»). Матрица SoD-рисков представляет собой квадратную матрицу, симметричную относительно главной диагонали, по обеим сторонам которой расположены условные обозначения релевантных функций, а на пересечении строки и столбца, соответствующих 2 функциям, констатирующим SoD-риск, отмечено наличие риска [2-3].

Для построения матрицы SoD-рисков для процесса закупки в модуле ММ был проведен анализ специальной литературы, посвященной администрированию бизнес-процесса в рассматриваемом модуле, в ходе которого выявлено 17 SoD-рисков. Результаты анализа оформлены в виде SoD-матрицы, представленной на рис. 1.

Для разработки ролей, однако, необходимо также понимать, какие именно полномочия позволяют выполнить функции, констатирующие SoD-риски. Для решения данной задачи для каждой функции с помощью трассировки полномочий были выявлены и оформлены в виде таблиц с приведенной в ниже структурой технические дефиниции полномочий, позволяющие выполнить соответствующую функцию.

Таблица

Полномочия для функции «создание/изменение записи кредитора»

группа полномочий	объект полномочий (ОП)	поле ОП	значение поля ОП	оператор
ХК01	F_LFA1_APP	ACTVT	01	
ХК02	F_LFA1_APP	ACTVT	02	
...				

Колонка «группа полномочий» служит для отграничения изофункциональных наборов полномочий. Колонки «объект полномочий (ОП)», «поле ОП», «значение поля ОП» содержат техническую дефиницию полномочия, позволяющего выполнить рассматриваемую функцию. Если одна группа содержит несколько значений для одного и того же поля одного и того же ОП, в колонке «оператор» указывается тип связи для данных значений (OR или AND).

	ф1	ф2	ф3	ф4	ф5	ф6	ф7	ф8	ф9	ф10	ф11	ф12	ф13	ф14	ф15	ф16	ф17	ф18	ф19	ф20
ф1	1																			
ф2		2	3	4	5				7	8	9					13	14	15		17
ф3		2																		
ф4		3																		
ф5		4																		
ф6		5																		
ф7								6												
ф8							6													
ф9		7																		
ф10		8																		
ф11		9									10									
ф12											10									
ф13														12	11					
ф14													12							
ф15													11							
ф16		13																		
ф17		14																		
ф18		15																		16
ф19																		16		
ф20		17																		

Условные обозначения функций: ф1 – создание/изменение основных данных кредитора; ф2 – создание/изменение заказа на закупку; ф3 – создание/изменение основной записи материала; ф4 – создание/изменение инфо-записи закупки; ф5 – создание/изменение условий ценообразования; ф6 – создание/изменение тарифов на услуги; ф7 – создание/изменение заявки на закупку; ф8 – одобрение заявки на закупку; ф9 – создание/изменение списка источников; ф10 – создание/изменение квоты; ф11 – создание/изменение контракта; ф12 – одобрение контракта; ф13 – создание/изменение соглашения о поставках; ф14 – создание/изменение плана поставок; ф15 – одобрение соглашения о поставках; ф16 – одобрение заказа на закупку; ф17 – проводка поставленного материала; ф18 – создание/изменение ведомости учета услуг; ф19 – одобрение ведомости учета услуг; ф20 – верификация счета

Рис. 1. Матрица SoD-рисков для процесса закупки (MM-PUR)

2. Проектирование и разработка ролей для модуля MM

Анализ специальной литературы не обнаружил разработанных методов для решения данной задачи. Поэтому для проектирования ролей мы решили визуализировать составленную матрицу SoD-рисков: принимая ее за матрицу смежности некоторого графа, был построен соответствующий граф. Вершины графа соответствуют функциям, входящим в состав выявленных SoD-рисков. Ребра графа, соединяющие две вершины, соответствуют SoD-рискам.

Визуализация матрицы SoD-рисков с помощью графа существенно упростила задачу проектирования ролей. Границы отдельных ролей демаркированы на графе с помощью пунктирных линий: чтобы роль не

содержала SoD-рисков, в границах одной роли не должно быть вершин, соединенных друг с другом ребром. Это основное условие для проектирования ролей, не содержащих SoD-рисков.

Далее границы отдельных ролей определялись на основании 2 критериев: во-первых, в зависимости от этапа процесса закупки/инвентаризации; во-вторых, в пределах одного этапа роли отграничивались друг от друга сообразно бизнес-логике процесса ММ.

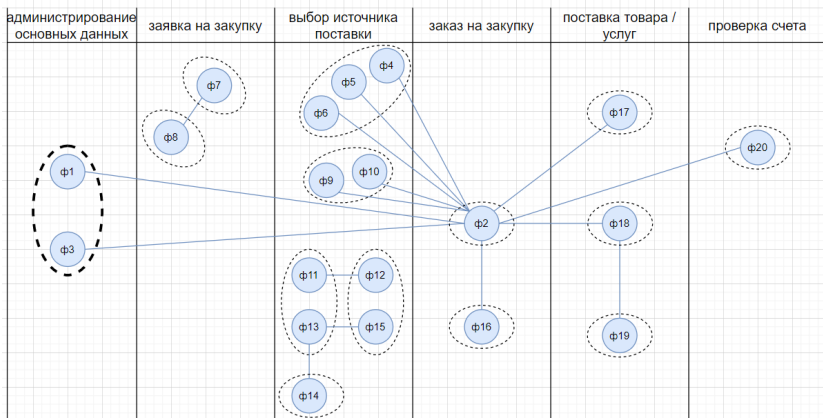


Рис. 2. Проектирование ролей для процесса закупки (MM-PUR)

Рассмотрим, как дальше велось проектирование ролей на примере первой роли – «Администратор основных данных для модуля ММ» (выделена жирным пунктиром на графе). Для каждой роли был определен набор транзакций, права на которые должна содержать проектируемая роль. Частично данный список детерминирован совместимыми с ролью функциями: в рассматриваемом примере это функции создание/изменение основных данных кредитора и основной записи материала (транзакции МК01, МК02, МК05, ХК01, ХК02, ХК05, ХК99, ММ01, ММ02, ММ06, ММ11, ММ12, ММ13, ММ16). Дополнительно в список транзакций для данной роли были добавлены транзакции, нерелевантные для SoD-рисков, но необходимые для администрирования основных данных в ММ (транзакции МК03, МК04, ХК03, ХК04, FK03, FK04, ММ03, ММ04, ММ14, ММ15, AC03).

Далее произведена непосредственная разработка ролей в системе на основании составленного списка транзакций для каждой роли. Стандартный процесс по разработке роли в системе SAP R/3 был

расширен дополнительным пунктом – проверкой профиля роли на отсутствие полномочий для несовместимых с данной ролью функций.

3. Тестирование ролей

Разработанные роли были подвергнуты 2-этапному функциональному тестированию. Вначале был проведен позитивный тест, в ходе которого было подтверждено, что роль позволяет выполнять встроенные в нее транзакции (см. пример на рис. 3а). На втором этапе с помощью негативного теста подтверждено, что разработанная роль не позволяет выполнять несовместимые с данной ролью функции (см. пример на рис. 3б), а, следовательно, не содержит SoD-рисков.

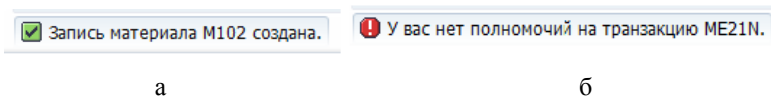


Рис. 3. Результат тестирования роли «Администратор основных данных для модуля ММ» (фрагмент)

Заключение

Модуль ММ в системе SAP R/3, изобилует SoD-рисками, которые относятся прежде всего к монетарным, а также качественным типам IT-рисков. Метод матрицы SoD-рисков, дополненный визуализацией с помощью графа, позволяет разработать роли для модуля ММ в SAP R/3, не содержащие выявленных SoD-рисков. Результаты исследования могут найти применение при разработке ролей для системы SAP R/3 и могут быть адаптированы для использования в других ERP-системах, авторизация в которых основана на модели контроля доступа RBAC.

Список литературы

1. Wilding E. Information Risk and Security: Preventing and Investigating Workplace Computer Crime / E. Wilding. – Burlington : Gower, 2006. – 341 p.
2. Danter E. Auditing Information Systems and Controls / E. Danter. – Bloomington: Xlibris US, 2007. – 238 p.
3. Wildensee C. Zugriffsberechtigungen / C. Wildensee. – Hamburg: Diplomatica Verlag GmbH, 2016. – 208 p.

Система построения таблиц лидеров на основе вероятностной модели для платформы турниров по программированию

Е. С. Бейлина

Студент магистр

В. В. Коротков

Ассистент

Введение

Одной из основных проблем современной высшей школы является снижение у студентов интереса к образованию. Инертность и незаинтересованность студентов в познании наук может повлечь за собой упадок развития общества. Из-за вышеизложенной проблемы усложняется отбор квалифицированных специалистов в IT фирмы. В условиях пандемии кампании были вынуждены отказаться от проведения многих мероприятий, направленных на знакомство студентов с технологиями, необходимыми для найма сотрудников, с помощью дней карьеры и т. п.

Так как обучение студентов с помощью онлайн курсов на онлайн платформах дало положительный результат и оказалось решением проблемы онлайн обучения и получения образования в условиях пандемии, было решено разработать онлайн платформу для проведения IT турниров по разным технологиям. Подразумевается, что данное решение поможет студентам достичь высот и поможет им в поисках работы. Так, например, IT компания может создать турнир, победитель которого сможет пройти в ней стажировку. Плюс данной платформы выражается не только в повышении интереса студентов к изучению IT технологий, но и в предоставлении спонсорам турниров возможности прославиться среди студентов, что приведет к увеличению конкуренции на рабочее место, следовательно кампании смогут найти более квалифицированных специалистов.

1. Описание сервиса

Главная задача проекта состоит в создании функционала для проведения IT турниров среди студентов. Для спонсоров – платформа предоставляет возможность создания, изменения, запуска и оценивания турниров по разным языкам программирования и технологиям. Главная

особенность турниров – функционал настройки процента содержания использованных в нем технологий. Для студентов – выбрать турнир по интересной им технологии, принять в нем участие и загрузить выполненное задание. Если участник турнира внезапно передумает – существует возможность отмены своего участия.

Самым важным моментом проекта является рейтинговая система. После оценивания загруженных участниками заданий турнира, необходим справедливый функционал для расчета навыка участников. Навык сложно измерить, поэтому была создана математическая модель для расчета рейтингов участников турниров. Чтобы результаты по различным технологиям, использованным в турнире, были независимые, модель была построена на базе уже хорошо зарекомендовавшего себя алгоритма TrueSkill.

2. Описание математической модели

TrueSkill – это система ранжирования на основе навыков для Xbox Live, разработанная в Microsoft Research. Целью рейтинговой системы является выявление и отслеживание навыков игроков в игре, чтобы иметь возможность сопоставить их в соревновательных матчах.

Чтобы понять в чем отличие данной системы от других, рассмотрим алгоритм Эло. Система рейтинга Эло – это метод расчета относительных уровней навыков в играх для двух игроков, таких как шахматы и го. Система состоит из двух уравнений:

Учитывая рейтинг игрока А и рейтинг игрока В, вычисляется вероятность выигрыша игрока А и (наоборот)

$$E_a = \frac{1}{1 + 10^{\frac{R_b - R_a}{400}}}. \quad (1)$$

Индексы а и b обозначают игроков, E – ожидаемая вероятность для игрока а, а разница составляет $R_b - R_a$. Использование константы 400 можно отрегулировать в зависимости от соотношения ожидания с различиями в рейтингах.

Учитывая текущий рейтинг игрока R, ожидаемую вероятность выигрыша E и фактический результат S для K-фактора, который можно регулировать для управления чувствительностью обновления:

$$R'_a = R_a + K(S_a - E_a). \quad (2)$$

Эло не справляется с новичками и командами. Алгоритм TrueSkill обобщает Эло, отслеживая две переменные: μ средний навык и σ неопределенность системы в отношении этой оценки (стандартное отклонение). Он делает это вместо того, чтобы полагаться на

фиксированный К фактор и получает динамический К фактор. Это решает проблему новичков, поскольку устраняет необходимость в «предварительных» играх [1].

Основная идея состоит в том, что существует какое-то предварительное убеждение, после наблюдений некоторых свидетельств, которые обновляют убеждение, оставляя с обновленным апостериорным убеждением, укрепляя уверенность в убеждениях.

Алгоритм TrueSkill является байесовским, потому что он состоит из априорного значения, умноженного на вероятность.

Основная модель для нахождения рейтинга технологий, использованных участниками турнира основана на факторном графе. Граф, который связывает факторы и переменные, называется «факторным графом».

Ключевая идея факторного графа состоит в том, что существующие предельные условные вероятности представляются как переменные, а каждая основная функция этих переменных как «фактор». Каждый фактор является узлом в сети, оптимизированной для повышения эффективности. Эффективность данного подхода заключается в том, что узлы-факторы отправляют «сообщения» другим узлам. Эти сообщения помогают упростить дальнейшие предельные вычисления. Передача сообщений выделена стрелками. Серые стрелки представляют сообщения, идущие вниз по модели, а черные показывают сообщения, идущие вверх [2].

Модель предоставлена на рис. 1 и состоит из пяти факторов. Приоритетный фактор запускает весь процесс. Здесь получается предыдущий уровень навыков игрока (из таблицы рейтингов, находящейся в базе данных). На этом этапе добавляется неопределенность к стандартному отклонению навыка, чтобы сохранить интересную динамику игры и не допустить, чтобы стандартное отклонение достигло нуля, поскольку остальная часть алгоритма сделает его меньше. Начальный навык игрока, полученный из гауссова распределения со средним m_0 и дисперсией v_0 :

$$s_i \propto N(m_0, v_0), \quad (3)$$

После каждого турнира навык игрока изменяется на случайную величину, взятую из гауссианы:

$$s_i \propto N(s_i, \gamma^2), \quad (4)$$

Фактор правдоподобия предполагает, что каждый игрок демонстрирует реальную результативность в каждом матче в соответствии с формулой:

$$perf_i \propto N(skill_i, \beta^2), \quad (5)$$

где β – настраиваемый параметр, отражающий степень случайности в игре. Цепочка навыков состоит из худшего игрока в крайнем левом и лучшего игрока в крайнем правом углу. Каждый последующий человек в цепочке получает β лучше и имеет большую вероятность победы над более слабым игроком.

Суммарно-взвешенный фактор правдоподобия по каждой технологии вычисляет производительность участника в целом. Предполагается, что производительность участника складывается из результатов каждой технологии. Особенность этого фактора заключается в том, что можно взвесить вклад каждой технологии по количеству процентов, которое было заявлено в турнире. Этот фактор берет набор гауссиан, а затем суммирует их относительно процента технологии в турнире:

$$perf_{player} = \sum_{i \in player} perf_i \frac{percent}{100\%}. \quad (6)$$

Суммарно-взвешенный фактор правдоподобия каждого участника основан на вычитании результата участников, чтобы получить попарные различия для их сравнения:

$$perf'_{player} = perf_{player_{i+1}} - perf_{player}. \quad (7)$$

Фактор сравнения состоит из коэффициентов сравнения в зависимости от фактического наблюдения исхода игры. Это происходит за счет функций «v» и «w»:

$$v = \frac{N(perf'_{player} - \varepsilon)}{\Phi(perf'_{player} - \varepsilon)}, \quad (8)$$

$$w = v(perf'_{player}, \varepsilon) * (v(perf'_{player}, \varepsilon) + t - \varepsilon). \quad (9)$$

Для повышения точности вычисления различий участников, передаваемые сообщения, постоянно обновляются. В каждой новой итерации цикла изменения будут менее драматичными, и в конечном счете значения для каждого маргинала стабилизируются [3].

Как только внутреннее расписание стабилизирует значения в нижней части графика факторов, сообщения возвращаются обратно вверх по графику. Эти обратные сообщения представлены черными стрелками на рис. 1. Новый рейтинг навыков каждого игрока будет значением маргинальной переменной навыков игрока после того, как сообщения достигнут вершины графа факторов [4].

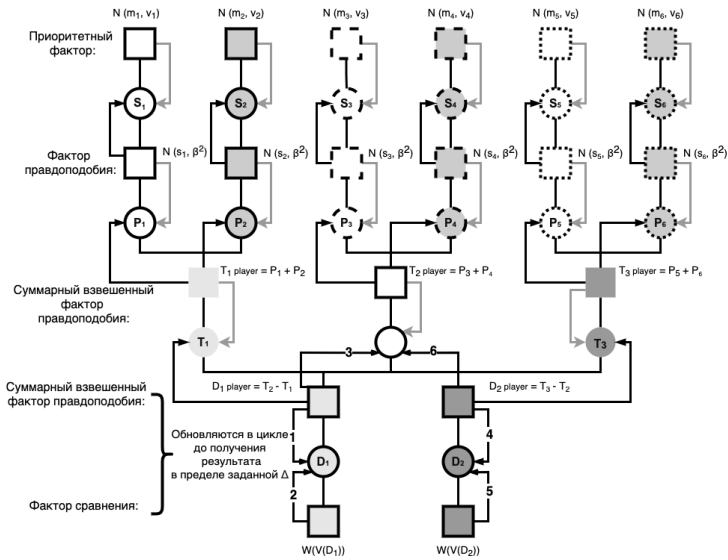


Рис. 1. Модель рейтинговой системы в виде факторного графа

3. Численный пример

В табл. 1 по вертикали представлены 5 турниров с различными технологиями и их процентами содержания в турнирах. По горизонтали расположены участники с их местами относительно каждого из турниров.

Таблица 1

Места, занятые участниками в турнирах

турниры	1			2			3			4			5		
технологии и участники	js	java	js	python	java	js	python	java	python	python	java	js	python	java	python
	100	50	50	50	25	25	25	75	100						
user1	1	2			3		1								-
user2	-	1			2		2								3
user3	2	3			1		-								2

В табл. 2 по вертикали представлены участники с их рейтингами относительно технологий. По горизонтали расположены турниры, полное описание которых находится в табл. 1.

На рис. 2 расположен график, составленный по данным, предоставленным в табл. 2.

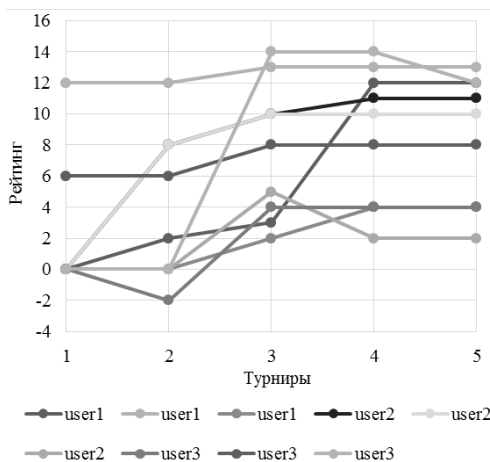


Рис. 2. График изменения рейтингов участников турниров

Таблица 2

Рейтинг участников после каждого турнира

	user1			user2			user3		
	java	js	python	java	js	python	java	js	python
1	-	12	-	-	-	-	-	6	-
2	2	12	-	8	8	-	-2	6	-
3	3	13	2	10	10	5	4	8	14
4	12	13	4	11	10	2	4	8	14
5	12	13	4	11	10	2	4	8	12

Заключение

Реализован проект по проведению турниров для IT специалистов с целью повышения их навыков и компетенции в программировании. Также проект решает проблему найма новых сотрудников в режиме самоизоляции. Для справедливого оценивания навыков участников турниров в каждой отдельной технологии создана математическая модель на базе рейтинговой системы TrueSkill.

Список литературы

1. Рейтинговая система TrueSkill. [Электронный ресурс] – Режим доступа: <https://www.microsoft.com/en-us/research/project/trueskill-ranking-system/>

2. TrueSkill2 – улучшенная рейтинговая система. [Электронный ресурс] – Режим доступа: <https://www.microsoft.com/en-us/research/uploads/prod/2018/03/trueskill2.pdf>

3. Рейтинг и матчмейкинг. [Электронный ресурс] – Режим доступа: <https://www.microsoft.com/en-us/research/wp-content/uploads/2006/10/Game-Developer-Feature-Article-Graepel-Herbrich.pdf>

4. Математика, стоящая за алгоритмом TrueSkill. [Электронный ресурс] – Режим доступа: <http://www.moserware.com/assets/computing-your-skill/The%20Math%20Behind%20TrueSkill.pdf>

Выделение именованных сущностей из текста на основе предобученной модели bert_mult

Т. С. Гаршин

Студент магистр

А. Ю. Иванков

Доцент

Введение

Развитие технологий TextMining связано с решением ряда задач, одной из главных является проблема извлечения именованных сущностей из текста. Решение ее связано с парсингом предложений на естественном языке и выделения сущностей и понятий текста и определения отношений сущностей. Например, в задаче сбора данных о договорной деятельности компаний, актуально выделение: сторон договора – имен компаний, типов договора, условий и т.д.

В технологиях извлечения фактов из текста присутствуют этапы токенизации (выделения элементов) и лемматизации (или стемминга) для построения множества выделенных именованных сущностей. Общая архитектура подхода извлечения именованных сущностей из текста приведена ниже (рис. 1).



Рис. 1. Общая архитектура подхода извлечения именованных сущностей из текста

Статья посвящена применению метода выделения именованных сущностей из текста на основе нейросетевой модели ner_ontonotes_bert_mult, дообученной и настроенной на применение к

задаче сбора данных о договорной деятельности компаний, с учетом специфики собираемых данных.

2. Выделение именованных сущностей из текста с помощью `ner_ontonotes_bert_mult`

`Ner_ontonotes_bert_mult` – нейросетевая модель, базирующаяся на BERT — двунаправленной модели с `transformer`-архитектурой (рис.2) и предобучена на выборке из русскоязычной википедии и корпуса русского языка [1, 2].

Модель BERT нашла широкое применение в задачах извлечения именованных сущностей - NER (Named Entity Recognition). Пример для русского языка – DeepPavlov [3], она также используется для решения задач — моделирование языковых масок и предсказание следующего предложения.

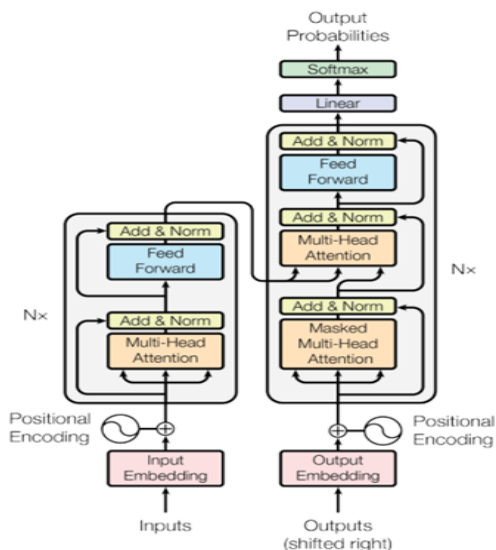


Рис. 2. Схема `transformer`-архитектуры

В текущих подходах в модель вводят энкодер и декодер.

Энкодер — преобразует слова входного предложения в один или больше векторов в пространстве признаков.

Декодер — генерирует из этих векторов последовательность слов.

Стандартные архитектуры для энкодера — RNN (рекуррентная нейросеть) или CNN (сверточная нейросеть), для декодера — чаще всего RNN.

Применение `Ner_ontonotes_bert_mult` позволяет выделить список из 19 токенов (рис.3), обозначающих: организации; люди (персонажи); продукты (товары); предприятия (порты, вокзалы, заводы); даты; деньги; числа и т.д.

Для различения соседних сущностей с одним и тем же тегом используется схема биотегирования, где «В» обозначает начало сущности, «I» означает «внутри» и используется для всех слов, входящих в сущность, кроме первого, а «O» означает отсутствие сущности.

«Росморпорт» разыгрывает контракт на строительство двухтопливных судов.

■ организация
■ контракт\законопроект
■ продукт

Рис. 3. Пример разбора предложения моделью `Ner_ontonotes_bert_mult` с целью выделения токенов именованных сущностей

Проблема качества (достоверности) извлеченных нейросетью фактов состоит в семантической специфике предметной области текстов. Анализ результатов экспериментов показывает часто курьезные ситуации. В связи с этим, практическое применение данного подхода требует настройки НС-модели на специфику примеров анализируемой предметной области.

3. Подход к извлечению именованных сущностей из текста русского языка

Для извлечения тегов из русскоязычных текстов, в работе использовался подход, основанный на дообучении НС-модели `ner_ontonotes_bert_mult` на специфичные примеры предметной области анализируемых текстов. Дообучение модели проводилось с использованием Colabatory (от Google). Выбор был определен в связи с: упрощенной системой подгрузки библиотек и решения их конфликтов, а также возможностью использовать GPU.

Эксперименты проводились с использованием простого обучения с учителем. На вход подавались предложения, извлеченные с новостных сайтов и собранные в файлы по ~200 предложений. После обработки предложений результат просматривался вручную (~6-7 тысяч тегов), ошибки исправлялись (рис. 4). Исправленный результат подавался на обучение в виде обучающей выборки (рис. 5).

```

Договор O
O O
создании O
нового O
предприятия O
Газпром V-ORG
нефть I-ORG
и O
ИКС V-ORG
Холдинг I-ORG
подписали O
в O
июне V-DATE
этого I-DATE
года I-DATE
на O
Петербургском V-EVENT
международном I-EVENT
экономическом I-EVENT
форуме I-EVENT
( O
ПМЭФ V-EVENT
) O
. O
оперативную O
коммуникацию O
с O
государственными O
органами O
+ O
такими O
как O
МВД V-ORG
^

```

Рис. 4. Формат обучающей выборки

```

▶ from google.colab import files
  uploaded = files.upload()
  !ls

```

Файл не выбран Upload widget is only cell to enable.

```

Saving test.txt to test.txt
Saving train.txt to train.txt
Saving valid.txt to valid.txt
drive sample_data test.txt train.txt valid.txt

```

Рис. 5. Загрузка обучающей, тестовой и валидационной выборок

В результате 5 итераций по дообучению качество разбора текста моделью значительно улучшилось (рис.6). Экспертным методом удалось установить, что точность выделения именованных сущностей повысилась на 8% относительно стартовой модели.

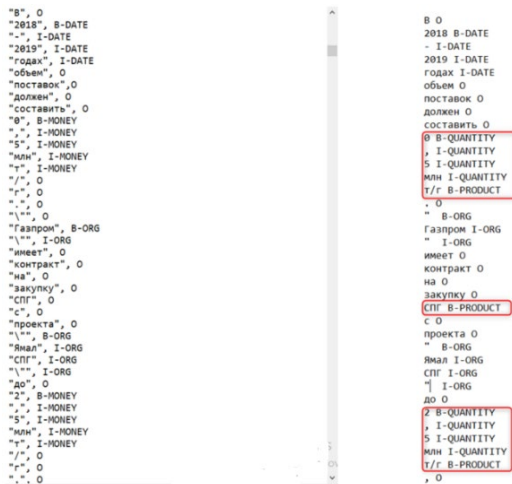


Рис. 6. Сравнение результатов разбора контрольной выборки (слева: до пост обучения, справа: после пост обучения)

Заключение

В результате проведенных экспериментов с дообучением НС-модели `ner_ontonotes_bert_mult` на специфичные примеры предметной области анализируемых текстов точность выделения именованных сущностей повысилась на 8% относительно стартовой модели.

Предлагаемый метод позволяет существенно повысить точность извлечения именованных сущностей – NER, а следовательно качество и надежность данных, используемых в дальнейшем для проведения аналитики.

Список литературы

2. Руководство для начинающих по BERT для мультиклассификации. [Электронный ресурс]. – Режим доступа : <https://www.machinelearningmastery.ru/beginners-guide-to-bert-for-multi-classification-task-92f5445c2d7c/>
3. Классификация текста с помощью BERT Tokenizer и TF 2.0 в Python [Электронный ресурс]. – Режим доступа : <https://pythobyte.com/text-classification-with-bert-tokenizer-and-tf-2-0-in-python-44cafd87/>
4. Burtsev M. DeepPavlov: Open-Source Library for Dialogue Systems / M. Burtsev [и др.] // Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics-System Demonstrations (Melborne, Australia, 15-20 июля 2018 г.) – 2018. – С. 122-127.

Методы выявления семантической близости новостных текстовых сообщений

И. С. Господарикова

Студент магистр

В. В. Гаршина

Доцент

Введение

Сегодня, с появлением большого спроса на аналитику данных, задачи автоматической обработки текстовых сообщений имеют особую актуальность. В частности, существуют и востребованы системы автоматического сбора информации, предназначенные для поддержки принятия бизнес-решений. Такие системы путём мониторинга различных Интернет-ресурсов собирают информацию о тех или иных важных фактах, и хранят её в том или ином виде, предоставляя пользователю интерфейс для взаимодействия.

При сборе информации эти системы ожидаемо сталкиваются с проблемами загрязнения данных, которые обусловлены в том числе частичным или полным семантическим дублированием обрабатываемых текстовых сообщений, размещённых на различных ресурсах. В силу этого, подобные системы нуждаются в алгоритмах очищения добытой информации, а то есть в алгоритмах выявления семантически близких сообщений.

Система, на которую ориентирована данная работа, собирает и анализирует в первую очередь новостные сообщения. Зачастую этот тип сообщения выделяется среди других текстов, в том числе упоминанием заведомо достоверных фактов, что и делает новости столь привлекательными для автоматического анализа. Новостное сообщение имеет ряд существенных особенностей, которые необходимо учитывать при создании алгоритмов выявления семантической близости, а именно:

- сравнительно небольшая длина сообщения;
- наличие заголовка;
- указание источника новости;
- указание даты публикации и дат упоминаемых событий;
- наличие в тексте большого количества имён и названий.

Данная статья посвящена анализу различных существующих методов выявления семантической близости текстов и их применимости для конкретной задачи и специфики данных.

1. Векторное представление слов

Широко распространённый подход к анализу текстов, также известный в литературе под названием *word embedding*. Включает в себя множество методов анализа текстов, которые основываются на представлении слов текста в виде числовых векторов.

$$W : \text{word} \rightarrow X : \mathbb{R}^n, \quad (1)$$

$$X = \{x_i, i = \overline{1, N}\}. \quad (2)$$

Частными реализациями подхода являются такие известные модели как *word2vec*, *bag-of-words*. Представление слов таким образом открывает возможность обрабатывать тексты методами машинного обучения, предназначенными для векторов, например, глубокими нейронными сетями, SVM или методами кластеризации.

Однако, у этого подхода есть существенные недостатки в рамках решаемой задачи.

1. Подход рассматривает смысл текста как сумму смыслов его слов. Не учитывается порядок слов, синтаксис предложений и т.д.

2. Модель обучается на корпусе текстов. Для новостных сообщений это означает увеличение веса семантически бесполезных дежурных фраз и вводных конструкций, которые встречаются в каждой новости, и не слишком высокое качество распознавания редких специфических слов, имён и названий, которые составляют основную часть искомой семантики.

3. Смысл слов в новости часто будет отличаться от смысла, который извлекается моделью. Слова «мужчина» и «женщина» зачастую представляются в моделях очень похожими векторами, так как контекст их употребления схож. Но для семантики новостного сообщения они будут не взаимозаменяемыми. Пример: «Мужчина не пострадал, женщина госпитализирована» – здесь при перемене этих слов местами мы точно можем сказать, что речь идёт о разных инцидентах, тогда как подход векторного представления эту разницу не отследит.

Данные недостатки подхода в целом способны нивелировать достоинства, поскольку никакой алгоритм машинного обучения не даст качественного результата на практике при плохом выборе признаков объекта. В силу этого от подхода *word embedding* в рамках работы было решено отказаться.

2. Текстуальное сходство

Этот подход, очевидно, основывается на определении и оценивании текстуального, т.е. дословного, совпадения текстов. Для семантически идентичных новостей, встреченных на разных ресурсах, бывает весьма характерно полное или частичное текстуальное дублирование. Такие семантические дубликаты часто немного видоизменяются, с текстуальной точки зрения представляя собой нечёткие дубликаты.

Для их отслеживания в работе используется подход анализа шинглов [1]. Текст новостного сообщения разбивается на последовательность слов.

$$W = \{w_i, i = \overline{1, N}\}. \quad (3)$$

Слова определяются пробелами, очищаются от знаков препинания и нормализуются по форме. После этого из последовательности слов W получают последовательность шинглов S следующего вида:

$$S = \{s_j = \{w_j, w_{j+1}, \dots, w_{j+k}\}, j = \overline{1, M}\}, \quad (4)$$

где число k – длина шингла, которую при анализе больших текстов рекомендуется брать около 8 или 10, а для маленьких сообщений, например, 3. Потом последовательности шинглов сравниваются. Это даёт оценить текстуальное сходство сообщений с учётом порядка слов и некоторых видоизменений. Происходит сравнение примерно так, как показано на рисунке 1.

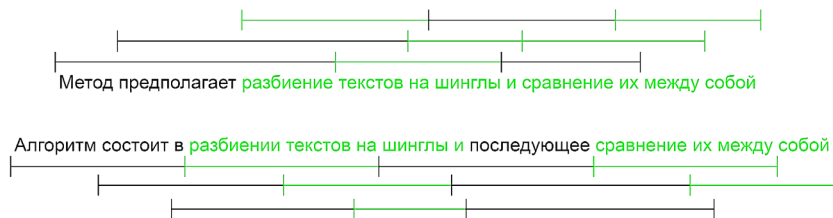


Рис. 1. Иллюстрация анализа шинглов

Достойных реализаций данного подхода не было найдено, поскольку большинство из них ориентировано на большие текстовые документы и на большие их потоки, то есть заточено в том числе для ускорения анализа текстов путём снижения точности алгоритма, поэтому в работе алгоритм реализуется своими силами.

Отдельная загвоздка состоит в том, чтобы как-то рассчитать метрику текстуального сходства. Можно оценивать процент дублирующихся слов в каждом из текстов, максимальную или среднюю

длину совпадающей последовательности шинлов, процент дублирующих шинлов среди всех и т.д.

3. Именованные сущности

Ещё один достаточно популярный и развитый подход к анализу текстов связан с распознаванием именованных сущностей, иначе он называется NER (Named Entity Recognition). Многие тексты новостных сообщений являются набором имён, названий и т.д., о которых что-то рассказывается. Эти слова собирательно называют именованными сущностями. Если выделить их в тексте, то можно добиться до некоторой степени «понимания» текста. На основе технологий NER строятся, например, некоторые вопросно-ответные системы. Данный подход имеет преимущество, ведь он акцентирует внимание именно на названиях и именах, которые несут в себе большую часть смысла любой новости.

В работе распознавание именованных сущностях реализуется при помощи библиотеки DeepPavlov [2], которая предоставляет предобученные нейронные модели в том числе для задач NER. На вход сети подаётся строка, а на выходе получается размеченный текст в виде массива из двух строк, в одной из которых содержатся слова исходного текста, а в другой – их метки. Для наглядности выход сети проиллюстрирован рис. 2.

```
В <---> O
Оренбуржье <---> B-GPE
в <---> O
феврале <---> B-DATE
2019 <---> I-DATE
года <---> I-DATE
запустят <---> O
молочный <---> O
комплекс <---> O

<---> O
16 <---> B-TIME
: <---> I-TIME
21 <---> I-TIME
22 <---> B-DATE
. <---> I-DATE
11 <---> I-DATE
. <---> I-DATE
2018 <---> I-DATE
```

Рис. 2. Иллюстрация распознавания именованных сущностей

4. Дополнительные атрибуты

Новостные тексты предоставляют ряд дополнительных возможностей для анализа по сравнению с обычным текстом. У них зачастую будут присутствовать такие атрибуты как заголовок, дата публикации и источник сообщения, которые можно использовать для

выявления семантической оригинальности. Главную ценность, конечно, будет представлять дата публикации новостей. В случае, если даты разнесены сильно, то даже при сжестости именованных сущностей и самих текстов, можно говорить о том, что вероятно оригинальность обеих новостей. Источник у двух новостей тоже можно сравнить, и при совпадении отдать предпочтение различию этих новостей, если только они не полные текстуальные дубликаты. Достоверность источника сообщения может определяться по хранимому словарю степеней доверия, составленному экспертным путём, и использоваться для отдания предпочтения одной из новостей.

Заключение

Данная статья посвящена главным образом проведённому в ходе научной работы исследованию существующих подходов к анализу текста и их применимости в задаче выявления семантически близких новостей. Описана суть самих подходов, а также принятых в отношении них решений. В настоящий момент избыточность из-за фактов-дубликатов в системе оценивается почти в 30%, так что предлагаемые методы могут существенно увеличить качество данных.

Список литературы

1. Зеленков Ю. Г. Сравнительный анализ методов определения нечётких дубликатов для Web-документов / Ю. Г. Зеленков, И. В. Сегалович // сб. трудов 9-ой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции»: сб. работ участников конкурса.. – Переславль-Залесский: «Университет города Переславля», 2007. – Т. 1., С. 166-174.
2. Burtsev M. DeepPavlov: Open-Source Library for Dialogue Systems / M. Burtsev [и др.] // Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics-System Demonstrations (Melborne, Australia, 15-20 июля 2018 г.) – 2018. – С. 122-127.

Использование цветовых моделей для улучшения работы алгоритмов распознавания

Р. С. Дорофеев

Студент магистрант

А. В. Акимов

Старший преподаватель

Введение

Лицо человека является важным источником информации при общении между людьми. Обнаружение лиц является первым этапом для создания системы идентификации и распознавания лиц человека. Обнаружения лица не требует физического контакта, что значительно расширяет возможности применения.

Одним из самых популярных алгоритмов обнаружения лиц является алгоритм, предложенный Виолой и Джонсом [1, 2]. В этом алгоритме используется технология скользящего окна. Смысл в том, что на исходном изображении объекты (например, лица) выделяются, если признаки в сканирующем окне совпадают с признаками лица на выделенной окном части изображения по итогам применения к нему каскада слабых классификаторов.

Нередко бывает, что этот алгоритм дает ложные срабатывания. Это случается из-за разных условий освещения (затемнении или сильно ярком свете). Но этот недостаток можно исправить, используя информацию о цвете кожи лица. Для этого существуют априорные методы. Основная идея состоит в том, что задается некий жесткий набор правил в определенных цветовых пространствах. Для этого используются следующие цветовые пространства: RGB, HSV и YCbCr.

1. Понятие изображения и его свойства

Цифровое изображение состоит из двумерного массива чисел, в котором каждое число соответствует одному пикселю. Пиксель это неделимый объект, формирующий изображения, характеризуемый определенным цветом или яркостью (для полутоновых изображений).

Черно-белое изображение имеет простой способ для его описания. Яркость каждого пикселя представляется всего двумя значениями:

нулем и единицей, черным и белым цветом. Такое изображение так же называют монохромным.

Полутоновое изображение кодируется с помощью битовой карты (матрица, хранящая значения пикселей). Каждый пиксель полутонового изображения кодируется различным количеством бит. К примеру, 2 бита это 4 полутона, 3 это 8 полутонов, 4 это 16 полутонов и 8 это 256 полутонов. При этом однобитовое изображение способно передать лишь 2 полутона, это и есть черно-белое изображение.

Цветное цифровое изображение состоит из нескольких цветовых каналов. Каждый канал представляет собой полутоновое изображение, содержащее 256 оттенков.

Для цифровых изображений используются различные способы цветопередачи, или, как их еще называют, цветовые модели. Самая популярная из них – это модель RGB (Red Green Blue – красный, зеленый, синий), в которой цвета получаются смешением трех основных цветовых компонентов: красного, зеленого и синего. Каждый цвет характеризуется его интенсивностью или насыщенностью. Количество каждого цвета может лежать в диапазоне от 0 до 255.

Помимо RGB в работе также используются модели YCbCr и HSV.

Модель YCbCr содержит информацию базовой яркости Y и двух цветовых компонент Cb (сине-желтая цветность) и Cr (красно-зеленая цветность). Буква Y означает светимость, которая вычисляется как усредненное компонент R, G и B. Остальные компоненты по существу определяются в виде разностей между светимостью Y и компонентами R, G и B: $Cb = B - Y$; $Cr = R - Y$. Cb это мера отклонения цвета от серого к синему или желтому, а Cr мера отклонения к красному или бирюзовому. Преимущество YCbCr перед RGB заключается в том, что Cb и Cr можно представить с меньшим разрешением чем Y. Это позволяет сократить объём информации без заметного ухудшения качества передачи цветовых оттенков изображения.

В модели HSV цвет описывается следующими параметрами: цветовой тон H (hue), насыщенность S (saturation), яркость, светлота V (Value). В этой модели используется цилиндрическая система координат (рисунок 1), а допустимые цвета представляют собой шестигранный конус, поставленный на вершину. Основание конуса представляет яркие цвета и соответствует $V = 1$. Тон H измеряется углом вокруг вертикальной оси OV. Величина S изменяется от 0 на оси OV до 1 на гранях конуса. Цвета, дополняющие друг друга до белого, находятся один напротив другого, их тона отличаются на 180 градусов. Процесс добавления белого цвета к выбранному выглядит как уменьшение насыщенности S, а процесс добавления черного цвета как уменьшение

яркости V . Ось OV соответствует ахроматическим цветам (серым тонам).

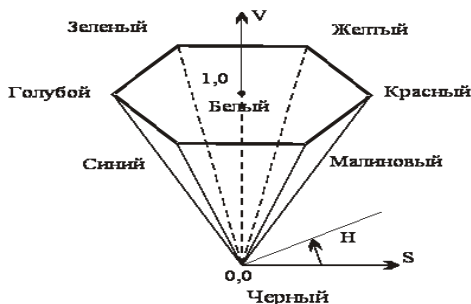


Рис. 1. представление модели HSV в цилиндрической системе координат [3]

2. Алгоритм Виолы-Джонса

В свое время алгоритм предложенный Полом Виолой и Майклом Джонсоном в области детектирования объектов был революционным. Суть его заключалась в объединении в единую конструкцию четырех подходов: Использование признаков Хаара, обучение классификаторов с помощью бустинга, использование слабых классификаторов и применение каскада классификаторов.

Признаки Хаара состоят из смежных прямоугольных областей черного и белого цвета (рисунок 2). Они позиционируются на изображении, после этого суммируются интенсивности пикселей в областях, далее вычисляется разность между суммами. Эта разность является значением определенного признака, определенного размера, определенным образом спозиционированного на изображении [4].

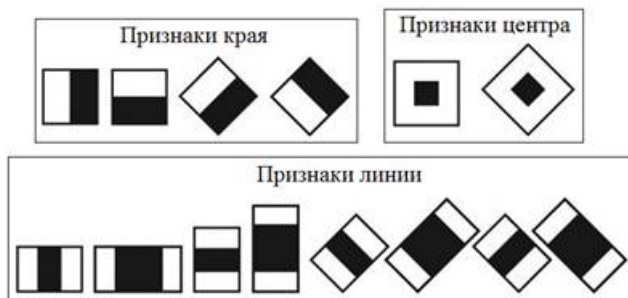


Рис. 2. Признаки Хаара [5]

Слабые классификаторы. Слабый классификатор представляет собой алгоритм принятия решения о классификации на основе значения, которое было получено при применении одного из признаков Хаара. Процесс можно описать так: ищется такое положение между двумя соседними отсортированными результатами применения признака Хаара, на основе которого строится данный слабый классификатор, к изображениям тренировочной выборки, и с учетом весовых коэффициентов будет делить тренировочные изображения на две группы [4].

Бустинг или усиление слабых классификаторов. Эта процедура заключается в комбинировании множества слабых классификаторов в один сильный. При обучении слабых классификаторов, выбирается наиболее точный из них и составляется линейная комбинация. При этом весовые коэффициенты каждый раз модифицируются, и следующий слабый классификатор при обучении выбирает те тренировочные изображения, с которыми не справились предыдущие [4].

Каскад классификаторов. Итоговый классификатор строится в виде каскада сильных классификаторов, описанных выше. Каждый слой каскада настраивается на распознавание как можно большего числа лиц, но учится отсеивать ложные положительные срабатывания, которые вернули предыдущие слои. В итоге каждый последующий слой каскада содержит большее количество слабых классификаторов, и запускается когда предыдущий, сообщит о возможности наличия в анализируемой области объект поиска [4].

3. Использование информации о цвете кожи в алгоритмах обнаружения лиц

Существуют методы, использующие информацию о цвете кожи лица для улучшения показателей работы стандартных алгоритмов распознавания. К таким относятся методы цветовой сегментации кожи на изображении: априорные, параметрические и непараметрические.

Цветовая сегментация – это процесс разделения цифрового изображения на несколько сегментов (на супер пиксели). Цель сегментации заключается в упрощении изображения для простоты его анализа. А результатом сегментации является множество сегментов, которые покрывают все изображение. Все пиксели в сегменте схожи по определенной характеристике, но соседние значительно отличаются по этой же характеристике [6, 7].

Априорные методы используют в качестве модели цвета кожи, заданные параметры в определенном цветовом пространстве. Метод имеет простоту реализации и отсутствие обучения, но при этом

приходится подбирать удобное цветовое пространство и адекватный набор правил.

Непараметрические методы используют идею конструирования вероятностной карты кожи. Каждому значению цвета, в цветовом пространстве ставится в соответствие вероятность принадлежности этого цвета коже. К этому классу методов относится классификатор байеса.

В параметрических методах, параметры модели вычисляются на основании тренировочных данных (изображения с выделенными областями кожи). Поэтому эти методы используют явное представление распределения цвета кожи в выбранном цветовом пространстве. Но эти методы склонны к высокому уровню ложного обнаружения.

4. Реализованный код программы

В проделанной работе используется алгоритм Виолы-Джонса. Основная идея заключается в обработке изображения на предмет нахождения лиц с помощью данного алгоритма, и осуществления дополнительной проверки на цвет кожи лица с использованием трех разных цветовых пространств.

Сначала активируется встроенный в Matlab каскадный детектор, реализованный на основе алгоритма Виолы-Джонса. Затем выбирается путь до исходных изображений, и путь, куда будут сохраняться обработанные изображения. Считывается содержимое папки с исходными изображениями, и для каждого из них осуществляется поиск лиц с использованием данного детектора.

После этого для каждого положительного результата детектирования производится дополнительная проверка на цвет кожи лица в трех цветовых пространствах: RGB, YCbCr и HSV. Для сегментации на области, содержащие и не содержащие кожу лица, используются параметры, предлагаемые в [6].

Далее, рассчитывается площадь, занимаемая пикселями, которые прошли проверку на цвет кожи лица по итогам сегментации, и делится на площадь всего положительного результата детектирования, и, затем, сравнивается с порогом. Если порог не был превышен, то результат детектирования отбрасывается.

В завершении производится запись в файлы изображений с отмеченными на них результатами детектирования.

5. Прделанный эксперимент

Эксперимент заключался в смене порога отношений площадей и проверки результатов детектирования при смене этого порога.

Пороговые значения изменялись в диапазоне от 0,1 до 1,0. На каждом этапе эксперимента, пороговое значение увеличивалось на 0,1 единицу.

Было проведено наблюдение, результатами которого являлось то, что количество ложных срабатываний уменьшалось на каждом шаге увеличения порога, но при этом и падал процент обнаруженных лиц.

Результаты экспериментов представлены на рисунке 3. Для ложных обнаружений за 100% бралось такое же количество обнаружений как и для оригинального детектора, но только без проверки на цвет кожи лица. Как видно из графика, количество ложных обнаружений снижается уже на четверть всего при пятипроцентном падении в точности обнаружения лиц.

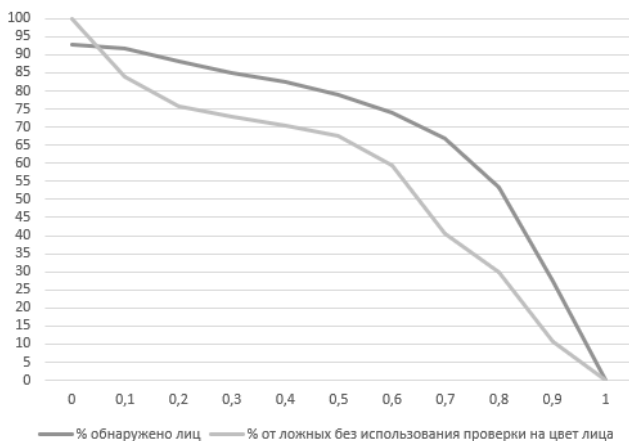


Рис. 3. Проценты правильно обнаруженных лиц и относительного числа ложных срабатываний в зависимости от порога как доли площади, занимаемой кожей лица

Заключение

В результате работы был реализован подход, позволяющий улучшить показатели работы алгоритма Виолы-Джонса. Для этого была добавлена дополнительная проверка на цвет кожи лица с использованием цветовых пространств RGB, YCbCr и HSV. Был проведен соответствующий эксперимент, на основании которого сделан вывод о том, что за счет падения на 5% в точности распознавания, можно снизить на четверть число ложных обнаружений.

Список литературы

1. Viola, P. Rapid Object Detection using a Boosted Cascade of Simple Features / P. Viola, M. Jones // Proceedings of the 2001 IEEE Conference on Computer Vision and Pattern Recognition 2001. – 2001. – Vol. 1. – P. 511-518.
2. Viola, P. Robust Real-Time Face Detection / P. Viola, M. Jones // International Journal of Computer Vision. – Netherlands: Kluwer Academic Publishers. – 2004. – № 57(2). – P. 137-154.
3. Компьютерная графика [Электронный ресурс]. – Режим доступа : <https://pandia.org/text/77/498/3202-5.php/> (дата обращения 16.05.2021).
4. Акимов, А.В. Разработка и исследование алгоритмов распознавания изображений на основе метода Виолы-Джонса с использованием технологии вычислений на графических процессорах CUDA / А.В. Акимов, А.А. Сирота // Вестник ВГУ (системный анализ и информационные технологии). – 2014. – № 3. – С. 100-108.
5. Друки, А.А. Применение алгоритма Виолы-Джонса, преобразования Хафа и искусственных нейронных сетей для распознавания символьной информации на сложном фоне / А.А. Друки, П.А. Каковкин, Д.С. Чернета // Научная визуализация. Электронный журнал открытого доступа [Электронный ресурс]. – 2016. – № 3. – Режим доступа : <http://sv-journal.org/2016-3/05/index.php?lang=en/> (дата обращения 16.05.2021).
6. Kaddouhi, S.E. Eye Detection based on Viola & Jones Detector, Skin Color, and Eye Template / S.E. Kaddouhi, S.E. Abderrahim. M. Abarkan. // International Journal of Control and Automation. – 2018. – Vol. 11(5). P.59-72.
7. Kaddouhi, S.E. A New Robust Face Detection Method Based on Corner Points / S.E. Kaddouhi, S. Abderrahim. M. Abarkan. // International Journal of Software Engineering and Its Applications. – 2014. – Vol. 8(11). P. 25-40.

Реализация модулей взаимодействия с различными типами приборов Arinst

Н. М. Жевандров

Студент магистр

П. С. Лысачев

Старший преподаватель

Введение

Имеется множество типов приборов линейки Arinst, часть из которых работает по протоколу, разработанному в ООО «Крокс Плюс». Для таких приборов потребовалось создание программного обеспечения (далее – ПО), которое работало бы на персональных компьютерах наших клиентов. Это ПО для всех таких приборов должно предоставлять возможности идентификации прибора (например, по идентификационному номеру) и обновления микропрограммы. Но для некоторых отдельных типов подобных приборов требуется особый функционал, дополняющий возможности прибора удобствами персонального компьютера (например, большой дисплей с большим разрешением). Поэтому было решено создать единое универсальное ПО, где требуемый функционал работы с данными приборами был бы представлен в виде различных модулей взаимодействия. Такие модули должны переключаться в автоматическом режиме (без манипуляций со стороны пользователя) во время работы ПО в зависимости от типа подключаемого прибора.

Данная работа посвящена разработке программного обеспечения, реализующего модули взаимодействия с различными типами приборов линейки Arinst.

1. Описание модулей взаимодействия

1. Модули взаимодействия состоят из 2 частей: backend и frontend. Backend реализует процессы обработки данных, которые меняются в зависимости от типа прибора. Frontend реализует различные графические элементы пользовательского интерфейса (графики и их компоненты, страницы главного меню), с которыми взаимодействует пользователь программного обеспечения.

2. Рассмотрим разницу модулей на примере приборов линеек Arinst VR (далее – VR-приборы) и Arinst PR (далее – PR-приборы). Приборы обеих этих линеек нацелены на измерение S-параметров [1]. Однако, VR-приборы в процессе сканирования могут измерить только коэффициенты S11, в то время как PR-приборам в процессе сканирования доступны для измерения коэффициенты S11 и S21, как по отдельности, так и одновременно. При этом PR-приборы имеют возможность прямой калибровки (Through) помимо однопортовой (Short, Open и Load) [2]. Кроме того, в аспекте графического интерфейса PR-приборам доступны новые виды графиков (в том числе развернутая фаза) и использование данных различных коэффициентов (S11 или S21) в качестве источника для графиков.

3. Модули бывают базовым (общий для всех типов приборов) и специализированными. Базовый модуль реализует лишь базовый функционал нашего протокола (идентификация и обновление микропрограммы) без учета каких-либо особенностей отдельных типов приборов. Такой модуль доступен для всех приборов линейки Arinst, работающих по нашему протоколу.

4. Специализированные модули разработаны с целью поддержки взаимодействия с конкретными типами приборов. Для всех специализированных модулей определена минимальная необходимая версия микропрограммы прибора.

5. Критерии выбора специализированного модуля:

- для данного типа прибора существует специализированный модуль. Например, PR-модуль для приборов линейки Arinst PR;
- версия микропрограммы данного прибора не меньше минимальной необходимой, определенной для этого модуля. В противном случае пользователь получит уведомление о необходимости обновления микропрограммы прибора, чтобы подходящий специализированный модуль мог быть загружен.

6. Если один из критериев не выполняется, выбор будет сделан в пользу базового модуля.

2. Реализация модулей взаимодействия

Теоретическое описание составных частей модулей взаимодействия было представлено в разделе 1. Теперь рассмотрим их практическую реализацию.

В менеджере устройств каждому типу прибора сопоставлен определенный специализированный модуль. При этом к одному типу прибора могут относиться разные виды приборов. Например, разным видам приборов Arinst VR 1-6200 и Arinst VR 23-6200, относящимся к

одному типу приборов – VR, соответствует специализированный модуль VR.

В backend процессы обработки данных (такие как сканирование, калибровка и трассы) выражены в виде сервисов. Ими являются взаимозаменяемые объекты, которые существуют в единственном экземпляре и основаны от общего родителя. Иерархия таких объектов представлена на рис. 1.

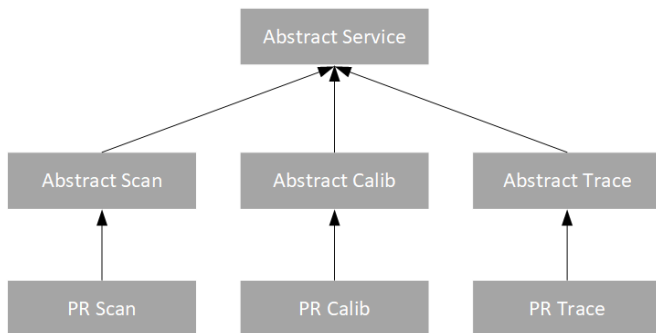


Рис. 1. Иерархия классов сервисов

Графическая сторона модулей взаимодействия (она же frontend) содержит в себе графики с их компонентами (маркеры и таблица маркеров) и контроллеры страниц главного меню (далее – контроллеры). И графики, и контроллеры имеют схожую иерархию классов и устройство с сервисами из backend.

За каждую сторону модулей (backend и frontend) отвечает соответствующий конфигуратор, который управляет «загрузкой» и «выгрузкой» взаимозаменяемых объектов, а также содержит в себе списки таких объектов для каждого модуля взаимодействия. У конфигураторов можно получить текущие активные объекты каждой стороны модуля.

Под «загрузкой» и «выгрузкой» взаимозаменяемых объектов понимается управление их ресурсами и обработчиками событий. Таким образом, все неактивные объекты на момент выгрузки должны освободить все свои ресурсы из памяти (различного рода буферы, компоненты), а активные объекты в момент загрузки, напротив, создают свои ресурсы, занимая место в памяти. Из-за управления обработчиками событий неактивные объекты «обесточиваются», погружаясь в буквально неактивное состояние.

На рис. 2 представлен пример интерфейса ПО с базовым модулем взаимодействия. В нем нет никаких графиков, а из пунктов главного меню есть только основные: информация о приборе, настройки ПО и информация о ПО.

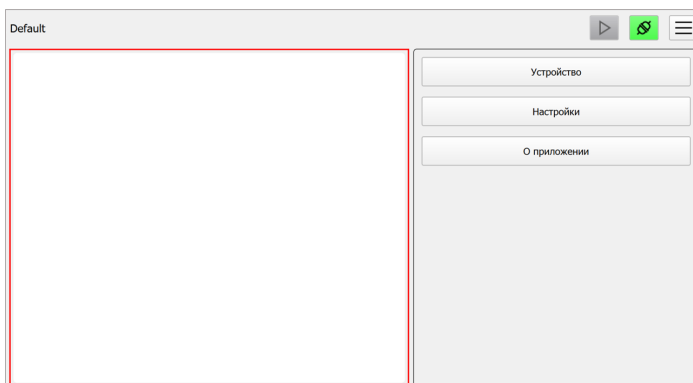


Рис. 2. Графический интерфейс ПО с базовым модулем

На рис. 3 представлен пример интерфейса ПО со специализированным модулем взаимодействия PR для приборов типа Arinst VNA-PR.

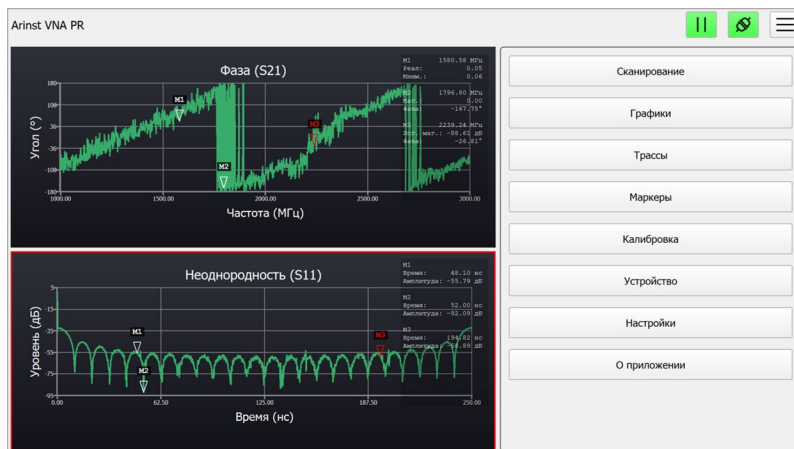


Рис. 3. Графический интерфейс ПО со специализированным модулем PR

В нем может быть от 1 до 4 графиков (на рис. 3 представлен вариант с двумя графиками), на каждом из которых могут быть маркеры (треугольники, направленные вниз, с подписью М и номером сверху) и соответствующая им таблица маркеров (в правом верхнем углу графиков). Пунктов главного меню намного больше. Помимо основных добавились пункты сканирования, настройки графиков, калибровок и прочие.

3. Подключение прибора к разработанному ПО

Блок-схема подключения прибора к разработанному ПО представлена на рис. 4.

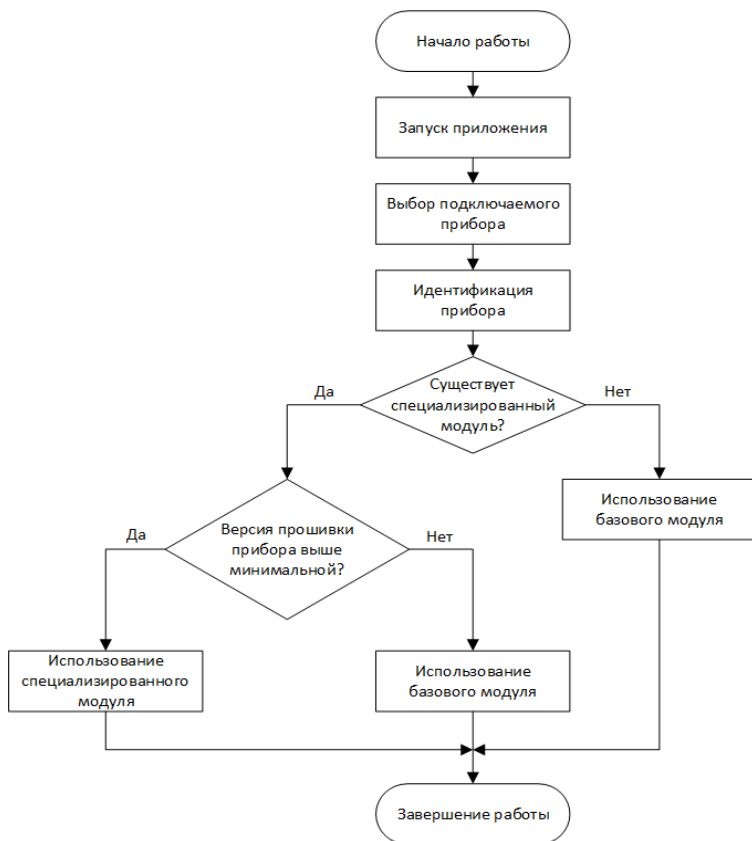


Рис. 4. Блок-схема подключения прибора к разработанному ПО

2. Приборы подключаются к персональному компьютеру посредством USB-кабеля по интерфейсу последовательного порта. Взаимодействие с нужным прибором осуществляется с помощью программного обеспечения Against Virtual Lab (далее – Against VL) при выборе пользователем в списке подключения прибора, с которым требуется установить связь. Взаимодействие с прибором осуществляется с помощью команд, определенных в нашем бинарном протоколе. При попытке установления связи Against VL отправляет прибору серию идентификационных команд, в результате которых становятся известны: идентификационный номер прибора, его тип и текущее состояние (режим загрузчика или основного приложения), и версия микропрограммы. Исходя из типа прибора и его версии микропрограммы, Against VL выбирает модуль взаимодействия в соответствии с критериями, указанными в разделе 1.

4. Переключение модулей взаимодействия

Если в данный момент уже был активен подходящий модуль взаимодействия, то смена модуля не требуется. В противном случае произойдет замена модуля. Независимо от этого произойдет сохранение и загрузка настроек приложения при подключении прибора. Это необходимо для актуализации настроек приложения.

Этапы алгоритма замены активного модуля взаимодействия:

- выгрузка сервисов активного модуля;
- замена активного модуля и загрузка сервисов требуемого модуля;
- выгрузка контроллеров бывшего активного модуля и общих;
- выгрузка графиков бывшего активного модуля;
- загрузка контроллеров и графиков требуемого модуля, общих контроллеров.

Блок-схема переключения модулей взаимодействия представлена на рис. 5.

Заключение

Данная работа посвящена разработке программного обеспечения, реализующего модуль взаимодействия с различными типами приборов линейки Against. Описаны составляющие модулей взаимодействия, механизм их реализации. Представлен алгоритм переключения модулей взаимодействия в автоматическом режиме. Результатом данной работы стала разработка единого универсального ПО, где требуемый функционал работы с данными приборами представлен в виде различных модулей взаимодействия.

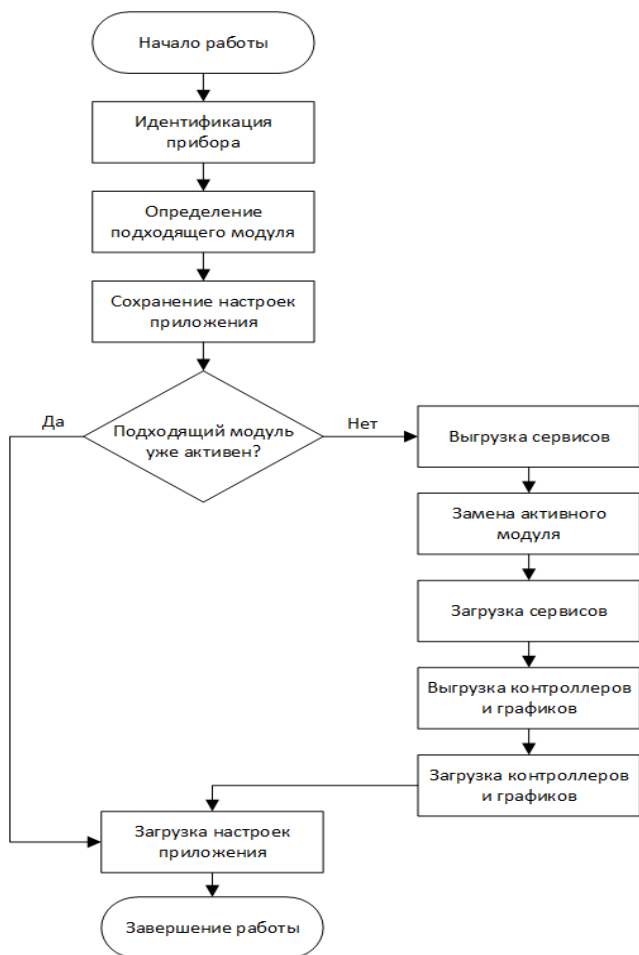


Рис. 5. Блок-схема переключения модулей взаимодействия

Список литературы

1. Белоус, А. И. СВЧ-электроника в системах радиолокации и связи. Техническая энциклопедия. Издание 2-е, дополненное. В 2-х книгах. Книга 2 / А. И. Белоус, М. К. Мерданов, С. В. Шведов. – М. : Техносфера, 2018. – 703 с.
2. Терешков, В. В. Современные методы и средства измерений на высоких и сверхвысоких частотах : учебное пособие / В. В. Терешков. – Ростов н/Д : Изд-во ЮФУ, 2018. – 113 с.

Разработка математического и программного обеспечения для решения задачи высокоточного обнаружения опорных точек на изображении лица человека

В. С. Жуков

Студент магистр

М. А. Дрюченко

Доцент

Введение

Современные стандарты мира кино и видеоигр требуют большой тщательности и высокой точности при воспроизведении внешности настоящих людей. Одно из важнейших мест в сфере компьютерной графики занимает создание 3D моделей человеческих лиц, которые должны быть максимально похожи на лица реальных людей. На данный момент, активно используются несколько способов создания цифрового дублера.

Один из наиболее перспективных способов заключается в помещении актера в фотограмметрическую установку и съемке необходимой сцены внутри. Для всех движений актера, мы получаем серию высокополигональных 3D моделей, которая полностью повторяет отыгранную актером сцену. Для дальнейшего использования необходимо преобразовать сканы в корректную совокупность ребер, вершин и полигонов. Эта проблема решается с помощью повторной ручной сборки топологии 3D художником каждого кадра из последовательности. Этот процесс требует долгой и кропотливой работы, но его можно автоматизировать. Для автоматизации используется алгоритм регистрации, в данной работе, для примера, будет рассмотрен алгоритм ICP (Iterative Closest Points) [1] – итеративный алгоритм ближайших точек, который позволяет деформировать заранее подготовленную модель с корректной топологией (базовая модель) под интересующую нас модель с иной топологией (целевая модель). Данный алгоритм при работе использует опорные точки базовой и соответствующие целевой модели. Но для высокого качества необходим алгоритм, способный с большой точностью найти такие точки. Именно разработке алгоритма

локализации ключевых точек для задачи регистрации посвящена данная дипломная работа.

1. Итеративный алгоритм ближайших точек

В качестве алгоритма регистрации используется алгоритм ICP или итеративный алгоритм ближайших точек. Как уже было сказано ранее, без подобного алгоритма процесс переноса мимики реального актера на его цифрового дублера является крайне длительной и ресурсоемкой задачей, так как требует огромного количества ручной работы высококвалифицированного 3D художника. Основная суть алгоритма ICP заключается в следующем, он пытается итеративно сопоставить, два набора точек, один из которых принадлежит опорному изображению, а другой целевому. Прежде всего, на базовой и целевой модели необходимо указать точки соответствия, благодаря которым, деформер сможет трансформировать базовую модель к целевой (рис. 1).

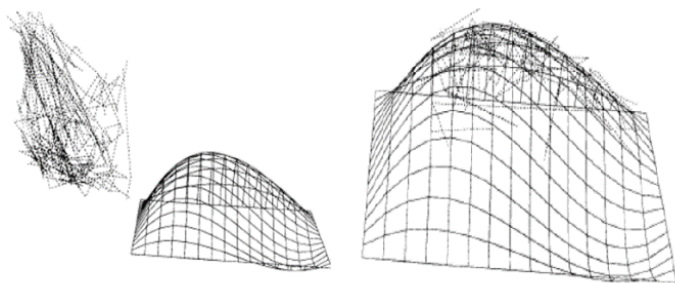


Рис. 1. Результат работы алгоритма ICP

Алгоритм ICP итеративно выполняет этапы поиска и деформации (рис. 2) пока не будет достигнут заданный порог ошибки, вычисляемой как среднееквадратичное отклонение между начальными и целевыми точками.

$$\frac{1}{N_p} \sum_{i=1}^{N_p} \|\bar{x}_i - R(\bar{q}_R) \bar{p}_i - \bar{q}_T\|^2, \quad (1)$$



Рис. 2. Схема работы алгоритма ICP

2. Активные модели внешнего вида

Активные модели внешнего вида (Active Appearance Models, AAM) [2] — это статистические модели изображений, которые путем разного рода деформаций могут быть подогнаны под реальное изображение. Этот алгоритм может быть использован для поиска ключевых точек. Перед использованием модель должна быть обучена на множестве заранее размеченных изображений. Каждая метка имеет свой номер и определяет характерную точку, которую должна будет находить модель во время адаптации к новому изображению. Процедура обучения активных моделей внешнего вида начинается с нормализации положения всех форм для того, чтобы компенсировать различия в масштабе, наклоне и смещении (рис. 3).



Рис. 3. Нормализация контрольных точек

После того, как все формы нормированы, из составляющих их точек формируется матрица:

$$S = [s_1, s_2, \dots, s_M] \quad (2)$$

$$s_m = [x_1^m, \dots, x_N^m, y_1^m, \dots, y_N^m] \quad (3)$$

После выделения главных компонент [3] указанной матрицы получаем выражение для синтезированной формы:

$$s = s_0 + \Phi_s b_s.$$

Здесь s_0 — форма, усредненная по всем реализациям обучающей выборки (базовая форма), Φ_s — матрица главных векторов, b_s — параметры формы. Приведенное выражение означает, что форма s может быть выражена как сумма базовой формы s_0 и линейной комбинации собственных форм, содержащихся в матрице Φ_s . Изменяя вектор параметров b_s мы можем получать разного рода деформации формы для подгонки ее под реальное изображение (рис. 4).

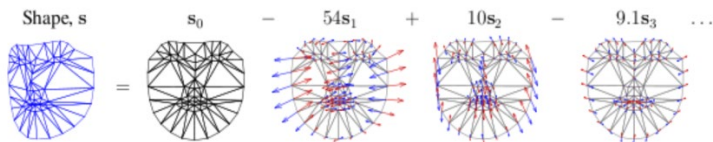


Рис. 4. Разложение модели на базовую и преобразования (Синие и красные стрелки – направления главных компонент)

Результатом обучения такой модели является регрессор ключевых точек (рис. 5).

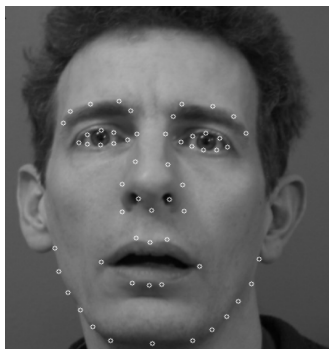


Рис. 5. Пример работы ААМ

3. Нейронные сети с координатными свертками

Для решения поставленной задачи было решено использовать архитектуру, которая включает в себя так называемые координатные свертки [4]. Сверточные нейронные сети пользуются огромным успехом в качестве ключевого инструмента для обеспечения эффективности глубокого обучения в сфере обработки изображений.

Одно из свойств сверточных слоев это инвариантность к смещению входа – смещение входного изображения на один пиксель вправо приведет к смещению выхода на один пиксель вправо (без учета шага свертки и паддинга).

Это свойство очень полезно при решении задач классификации/сегментации. Однако, это же свойство вредит при решении задач, требующих регрессии конкретных координат, потому что сети требуется каким-то образом превратить пространственную активацию фильтра в конкретное число. Для облегчения этой задачи предлагается использовать так называемые координатные свертки (рис. 6).

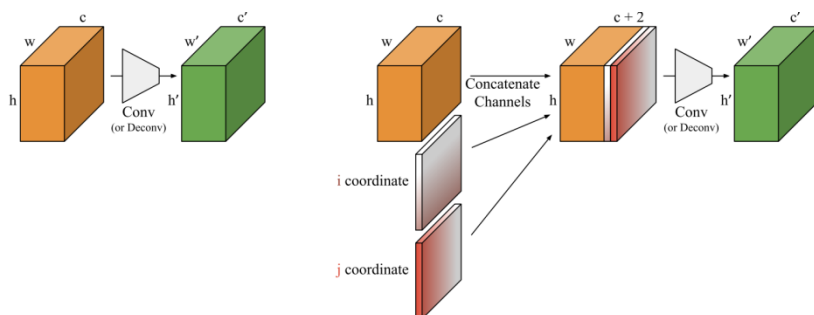


Рис. 6. Пример координатной свертки

Слои с координатными свертками обладают преимуществами обычных сверточных слоев такими как:

- Относительно мало тренируемых параметров;
- Быстро вычисляются на современных графических процессорах.

Однако, слои с координатными свертками не обладают инвариантностью к смещению входа, что делает их подходящими для нашей задачи.

Координатный слой реализован как расширение стандартной свертки, в которой дополнительные каналы создаются и заполняются информацией о координатах, после чего эти две матрицы становятся

дополнительными каналами и подаются на вход стандартному сверточному слою (рис. 6). Канал координат i представляет собой матрицу ранга 1 размером $x \times w$, первая строка которой заполнена нулями, вторая строка единицами, третья двойками и так далее. Для свертки над двумя измерениями достаточно двух координат, если измерений больше, можно аналогичным образом добавлять подобные матрицы.

4. Обучение нейронной сети

Реализованная нейронная сеть принимает на вход изображение человеческого лица, а на выходе выдает пары x и y для каждой точки контура на изображении.

В качестве модели нейронной сети был выбран ResNet 18 с полносвязанным слоем в виде головы сети.

Нейронная сеть разработана на базе фреймворка PyTorch, который позволяет строить граф вычислений «на лету», что в значительной мере ускоряет его работу. В качестве оптимизатора использовался Adam, а функции ошибки MSE.

В результате первых экспериментов было обнаружено, что собранный набор данных слишком одинаков и идеален, в связи с чем нейронная сеть достаточно быстро переобучается на обучающей выборке. Исправить эту ситуацию помогла аугментация данных. Она реализует сильные аффинные преобразования изображения, а также большое случайное изменение цветовой модели HSV изображения, которая позволяет изменять яркость и цветовой тон изображения, в результате чего решилась проблема переобучения на исходных данных.

5. Тестирование разработанного программного модуля

На данном этапе работы был получен детектор, который достаточно хорошо справляется с задачей поиска ключевых точек даже на изображениях с экстремальными эмоциями. Помимо этого, разработанный алгоритм был применен в качестве детектора для алгоритма ICP, используемого для задачи регистрации 3D персонажей. На рисунках 4.10 и 4.11 наглядно демонстрируется разница между работой алгоритма регистрации цифрового дублера без детектора и с детектором, использующим разработанный алгоритм. Также группой специалистов был проведен ряд испытаний, в ходе которых, путем зрительного анализа, было отмечено значительное увеличение качества результатов работы алгоритма ICP с детектором относительно работы алгоритма без детектора (рис. 7).

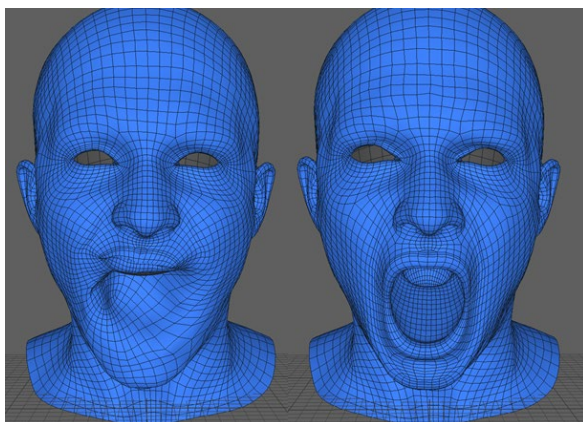


Рис. 7. Пример работы алгоритма регистрации без детектора и с детектором

Заключение

Данная статья посвящена разработке и реализации программного модуля для поиска ключевых точек на изображении лица человека, который позволяет во многом упростить задачу регистрации 3D персонажей. Описаны алгоритмы поиска ключевых точек и создание самого разрабатываемого программного модуля. В итоге данной разработки получился полностью рабочий программный модуль, реализованный в виде обученной модели. Использование данного решения позволяет значительно упростить задачу регистрации опорных точек при создании цифровых дублеров.

Список литературы

1. Rusinkiewicz, S. Efficient Variants of the ICP Algorithm / S.Rusinkiewicz, M. Levoy // 3D Digital Imaging and Modeling —2001. — С. 123–198
2. Cootes, T. Active appearance models. In Proceedings of the European Conference on Computer Vision / T. Cootes, G Edwards, C. Taylor // Active appearance models. In Proceedings of the European Conference on Computer Vision. — 1998. —Т.2. — С. 484–498.
3. Пao, С. Р. Линейные статистические методы и их применения / С. Р. Пао – Москва: Наука (Физматлит), 1968. — 548 с.
4. An intriguing failing of convolutional neural networks and the CoordConv solution /R. Liu [et al.] // Uber AI Labs — 2018.

Имитационное моделирование радиодеструкции полимеров

С. А. Зайцев

Студент магистр

М. А. Глушков

Студент магистр

М. Е. Семенов

Профессор

Введение

Одним из непосредственных атрибутов хозяйственной деятельности человека является производство полимерных материалов. С середины прошлого века общемировое производство пластмасс растет по экспоненте, и в настоящее время годовой объем производства составляет 350–400 миллионов тонн. Общую динамику можно проследить на рис. 1. Общее же количество произведенного пластика составляет около 6–8 миллиардов тонн. Из этого количества было переработано около 9%, а сожжено 12% [1].

Главное достоинство пластмасс – долговечность, – за которым так гнались изобретатели пластика в начале прошлого столетия, сегодня обернулось недостатком. Чем больше пластмассы мы используем, тем быстрее растет количество отходов, срок разложения которых в естественных условиях многократно превышает тот предел, в котором окружающая среда не несет ущерба. Помимо попадания в естественные пищевые цепочки, отходы, разлагаясь в течение столетий, могут выделять вещества, влияние которых на биосферу непредсказуемо. Также стоит принимать во внимание наличие добавок в готовых изделиях из полимеров – пластификаторов, красителей и т. д., – которые принимают непосредственное участие в реакциях разложения. Таким образом, поиск альтернативных способов переработки отходов данного вида становится значимой задачей для современного человечества. Для достижения данной цели могут быть применены не только химические способы, но и радиационные (так называемый радиолиз).

Также в развитых странах активно развиваются технологические процессы полимеризации и вулканизации с применением радиационной

обработки сырья. Так, например, в Японии 90% шин производится методом радиовулканизации. В России активно разрабатываются технологии повторного использования бутилрегенерата в производстве кровельных материалов [2, 3].

Важным шагом для внедрения радиолиты в промышленное производство (как и для развития технологий радиационной переработки) является создание компьютерных моделей, позволяющих достоверно моделировать процессы, происходящие при облучении молекулярных структур полимеров. Рассмотрим имитационное моделирование процессов построения молекулярной структуры и ее радиационного разложения.

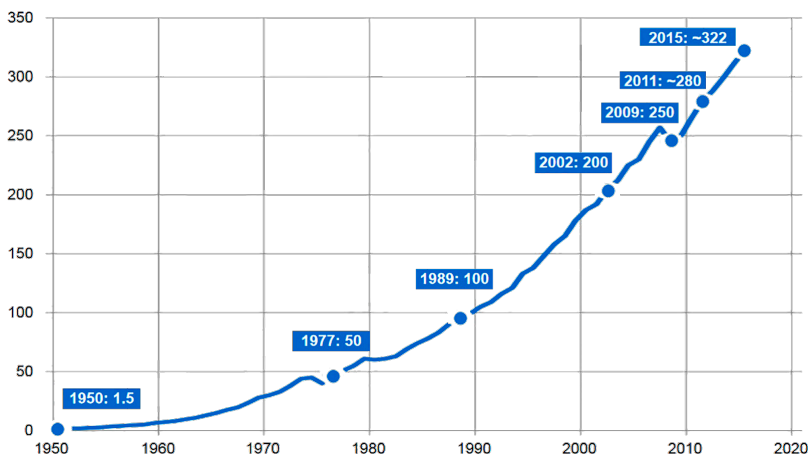


Рис. 1. Общемировое производство пластика по годам в млн. тонн

2. Моделирование пространственной структуры

В простейших статистических теориях полимерную молекулу моделируют в виде бестелесной свободно-сочлененной цепи, отдельные звенья которой подвергаются хаотическому тепловому движению. Тогда построение полимерной цепи осуществляется последовательно, с помощью углов Эйлера:

$$R = \begin{pmatrix} \cos(\alpha) \cos(\gamma) - \cos(\beta) \sin(\alpha) \sin(\gamma) & -\cos(\gamma) \sin(\alpha) - \cos(\alpha) \cos(\beta) \sin(\gamma) & \sin(\beta) \sin(\gamma) \\ \cos(\beta) \cos(\gamma) \sin(\alpha) + \cos(\alpha) \sin(\gamma) & \cos(\alpha) \cos(\beta) \cos(\gamma) - \sin(\alpha) \sin(\gamma) & -\cos(\gamma) \sin(\beta) \\ \sin(\alpha) \sin(\beta) & \cos(\alpha) \sin(\beta) & \cos(\beta) \end{pmatrix} \quad (2)$$

Находится вектор с началом в последнем узле молекулы, согласно заданным углам поворота, затем на векторе откладывается отрезок, равный длине межатомной связи, его конец – это координаты

следующего узла полимерной цепи. В случае цепи с фиксированными валентными углами по сжюему алгоритму применяется матрица поворота вокруг произвольной оси, где $\bar{v} = (x, y, z)$ – единичный вектор, задающий ось вращения, а θ – угол поворота.

$$M(\bar{v}, \theta) = \begin{pmatrix} \cos(\theta) + (1 - \cos(\theta))x^2 & (1 - \cos(\theta))xy - \sin(\theta)z & (1 - \cos(\theta))xz + \sin(\theta)y \\ (1 - \cos(\theta))yx + \sin(\theta)z & \cos(\theta) + (1 - \cos(\theta))y^2 & (1 - \cos(\theta))yz - \sin(\theta)x \\ (1 - \cos(\theta))zx - \sin(\theta)y & (1 - \cos(\theta))zy + \sin(\theta)x & \cos(\theta) + (1 - \cos(\theta))z^2 \end{pmatrix} \quad (3)$$

Так как в ансамбле есть несколько цепочек, после расчета каждого конечного мономера последующий мономер, то есть начало цепочки, задается случайным образом. Возможны случаи, когда мономер выходит за границы области моделирования, или оказывается расположенным слишком близко к уже имеющимся в модели мономерам. Данные ситуации также учитываются.

Блок-схема процесса построения молекулярного ансамбля представлена на рис. 2. Пример ансамбля из 3000 мономеров в ячейке пространства $20*20*20$ (в единицах длины связи между мономерами) представлен на рис. 3. Молекулы имеют вид цепочки с валентным углом в 75 градусов.

3. Моделирование процесса полимеризации

Вторым этапом в построении пространственной модели молекул является моделирование образования межмолекулярных связей между отдельными молекулами ансамбля. Данный процесс имеет место быть на этапе вулканизации или полимеризации эластомеров и полимеров, в результате образуется единая связанная пространственная структура, вторичные связи имеют на порядки более маленькие энергетические характеристики относительно первичных связей, но влияние их на эксплуатационные характеристики изделий из того или иного полимера очень и очень значительно.

Алгоритм построения вторичных связей состоит из двух этапов: выявление всех потенциальных пар мономеров, которые могут образовать вторичные связи (с учетом максимального расстояния между мономерами, при превышении которого связь не образуется), и непосредственно добавление связей в модель (с заданной вероятностью образования оных, которая может быть обоснована особенностями полимеризации моделируемого полимера) на основе потенциальных пар, рассчитанных на первом этапе.

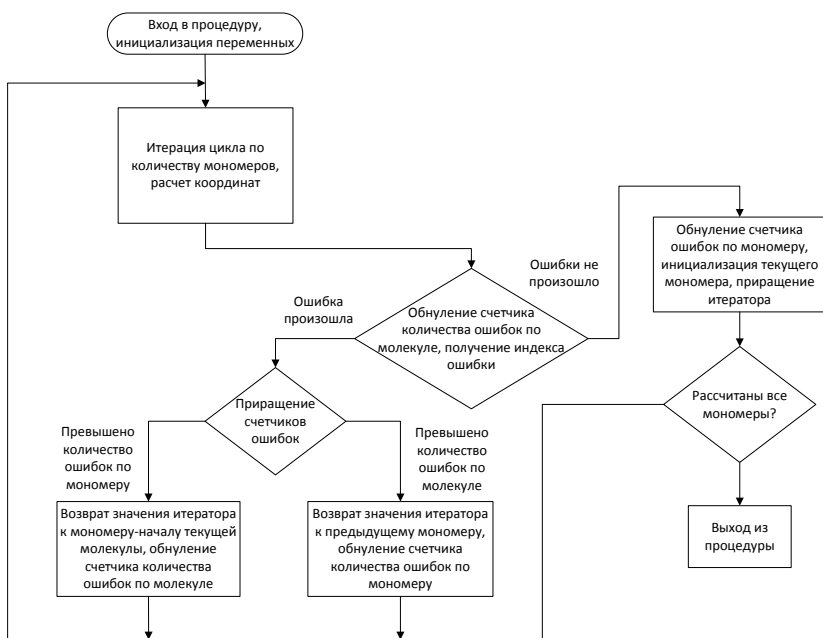


Рис. 2. Блок-схема процесса построения молекулярного ансамбля

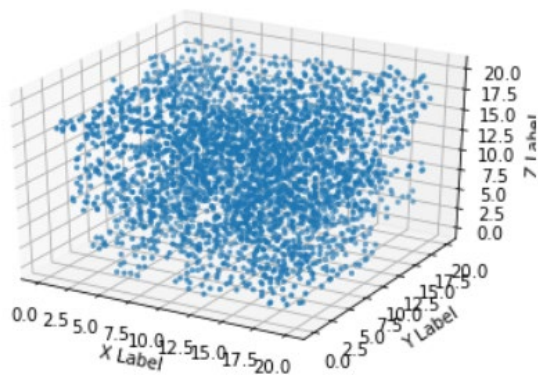


Рис. 3. Ансамбль из 3000 мономеров в 21 молекуле

На рис.4 представлена визуализация ансамбля без вторичных связей и с ними.

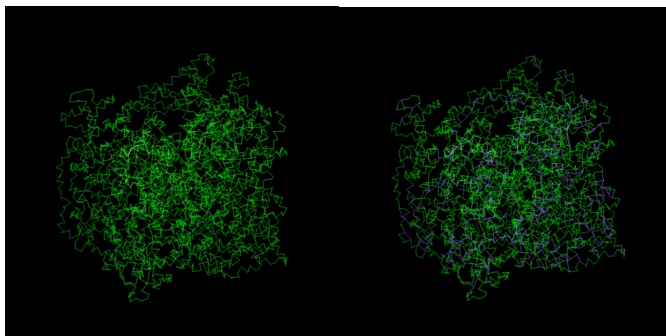


Рис. 4. Внешний вид ансамбля из 1000 мономеров до и после образования межмолекулярных соединений

Одной из важнейших характеристик для данного типа задач является молекулярно-массовое распределение (ММР, полидисперсность), позволяющее однозначно сопоставить текущее состояние внутренней структуры полимера и его эксплуатационные характеристики, степень содержания отдельных мономеров в образце и так далее. ММР представляет собой отношение соотношению количеств макромолекул различной молекулярной массы в данном образце полимера.

В качестве примера приведем модель ансамбля, изначально ММР которого было задано как набор коротких цепочек длиной до 500 мономеров. Всего в ансамбле 3000 мономеров, визуализация ММР до и после построения вторичных связей представлена на рис. 5 и рис. 6, соответственно.

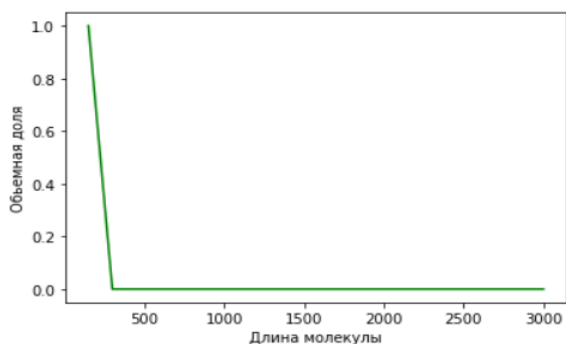


Рис. 5. ММР ансамбля до образования вторичных связей

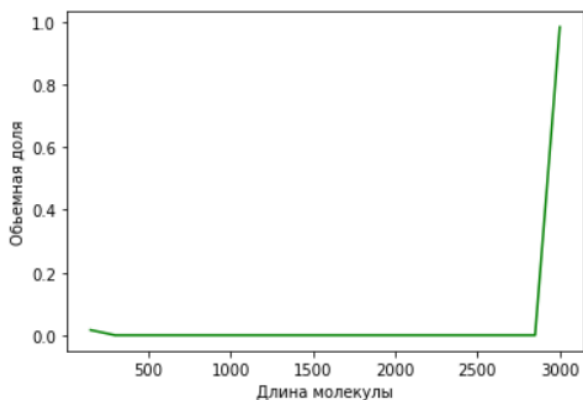


Рис. 6. ММР ансамбля после образования вторичных связей

Далее было произведено моделирование образования вторичных связей, вероятность образования связи из каждой потенциальной пары была задана как 98%. Как и следует ожидать из практики, ансамбль теперь представляет собой почти целую макромолекулу, за исключением незначительной доли непрореагировавших низкомолекулярных цепочек.

4. Моделирование процесса радиолиза

Дальнейшим этапом является моделирование расщепления связей первого и второго типа под воздействием ионизирующего излучения. В качестве первого приближения взаимодействие заряженных частиц с ансамблем можно представить как одновременное прохождение нескольких траекторий заряженных частиц сквозь молекулярный ансамбль. Для расчета отклонения частиц от узлов молекулярной структуры применяется стохастическая аппроксимация упругого рассеяния по Резерфорду. Ионизация рассчитывается стохастическим приближением формулы Бете–Блоха. В качестве примера для моделирования радиолиза был взят тот же ансамбль из пункта выше и проведено моделирование его разрушения при использовании 1000 траекторий заряженных частиц. ММР ансамбля после воздействия представлено на рис. 7. Характерно, что ММР оказалось сдвинуто в сторону низкомолекулярных соединений.

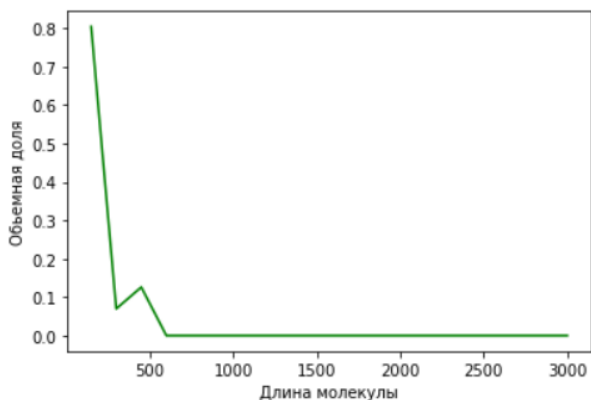


Рис. 7. ММР ансамбля после моделирования процесса радиолитза

Заключение

На основе представленных алгоритмов и разработанного программного обеспечения на языке Python планируется создать программный комплекс, позволяющий проводить моделирование процессов радиополимеризации и радиодеструкции широкого профиля полимерных материалов и предсказывать их физические свойства.

Список литературы

1. Geyer, R. Production, use, and fate of all plastics ever made / R. Geyer, J. R. Jambeck, K. L. Law // *Science Advances*. – 2017. – Vol. 3, №7.
2. Modeling of the destruction process of butyl rubber / S.G Tikhomirov [et al.] // *Radiation physics and chemistry*. – 2019. – Vol. 158 – P. 205–208.
3. Оптимизация свойств радиационного регенерата, применяемого в эластомерных кровельных материалах / Шутилин Ю. Ф. [и др.] // *Вестник ВГУИТ*. – 2017. – №4 (74).
4. Контроль и прогнозирование параметров качества полимеров в процессе их деструкции в растворе / С.Г.Тихомиров [и др.] // *Теоретические основы химической технологии*. – 2018. – Т. 52, № 4. – С. 466–472.

Алгоритм устранения неопределенных значений в нормализованных реляционных базах данных

Г. В. Зинькевич

Студент магистрант

В. Ю. Храмов

Профессор

Введение

Термин «база данных с неполной информацией» впервые был введен В. Липским [1], однако до сих пор он не имеет однозначного толкования. По отношению к информации, зафиксированной в базе данных (БД), наряду со словом «неполнота» употребляются также слова «неопределенность», «недоопределенность», «переопределенность» (противоречивость), что отражает различие взглядов на содержание и поведение динамической модели предметной области, поддерживаемой автоматизированной информационной системой (АИС). До сих пор никто не попытался определить, что такое база данных с полной информацией, однако несомненно, что термины «полнота» и «неполнота» относятся не к общему представлению об адекватности модели, а характеризуют принятый в информационной системе способ представления информации, в силу которого некоторые данные в системе или отсутствуют вообще, или не могут быть выданы, хотя имеются в системе, или определены недостаточно точно.

Проблема наличия неопределенных значений в базе данных становится особенно актуальной при принятии решений в различных предметных областях, когда для этого разрабатываются автоматизированные системы поддержки принятия решений. В данных системах часто решаются задачи, связанные с классификацией (распознаванием) объектов предметной области и отнесения определенных образов к эталонным классам. В этом случае часто информация о распознаваемом объекте является неполной (отсутствие информации по некоторым признакам, описывающим конкретный образ) или недостоверной (информация о значении признака имеется, но она не достоверна).

1. Алгоритм синтеза схемы реляционной базы данных в оптимальной третьей нормальной форме

Входными данными для алгоритма являются: конечное и непустое множество атрибутов предметной области $U = \{A_1, \dots, A_n\}$ и заданное на U множество ФЗ $F = \{L_i \rightarrow R_i \mid L_i, R_i \subseteq U; i = 1, m\}$ [2].

Алгоритм включает пять шагов.

1. Построить для заданного множества функциональных зависимостей \mathbb{F} редуцированное слева покрытие F' .

2. Построить для F' его расширение $F'' = \{L_i \rightarrow ((L_i)^+ \setminus L_i)\}$ и найти для F'' избыточное покрытие F_0 . Если замыкание одной из левых частей $(L_i)^+ = U$, то перейти на шаг 3, иначе – к множеству F_0 добавить функциональную зависимость $Z \rightarrow U$, где $Z \in U$.

3. Построить для F_0 редуцированное справа покрытие F^* .

4. Разбить F^* на классы эквивалентности и построить для F^* оптимальное покрытие G .

5. Из каждого класса эквивалентности E_G для G сформировать схему отношения R_i , которая включает множество атрибутов данного класса. Если в одной из полученных схем отношений R_i содержится атрибут Z , введенный на шаге 2, то удалить его из данной схемы. Оставшиеся атрибуты образуют ключ R_i .

Рассмотрим следующий пример, демонстрирующий работу алгоритма. Пусть $U = \{A, B, C, D, E\}$ и $F = \{ABC \rightarrow D, AB \rightarrow E, E \rightarrow AB\}$ на выходе алгоритма получим схему базы данных $\mathbb{R} = (R_1, R_2)$,

где: $S_1 = ECD, K_1 = \{E\}$; $S_2 = ABE, K_2 = \{AB, E\}$.

Представленный алгоритм синтеза порождает схемы отношений, которые находятся в сильной форме, чем ЗНФ в том смысле, что ни один из ключей, формируемых из левых частей ФЗ, входящих в класс эквивалентности, не зависит транзитивно ни от одного ключа.

Основной областью применения разработанного алгоритма синтеза схемы реляционной базы данных в оптимальной третьей нормальной форме является восходящее проектирование баз данных. Алгоритм целесообразно использовать и при нисходящем проектировании: на этапе преобразования высокоуровневых сущностей и связей в модели «сущность-связь» в низкоуровневые сущности и связи, при

нормализации низкоуровневых сущностей, у которых проблемы размерности атрибутов уже не существует.

2. Существующие алгоритмы устранения неопределенных значений в базах данных реляционного типа на основе функциональных зависимостей и их недостатки

Некоторый способ проверки допустимости частичных отношений. Оказывается, \mathbb{F} -зависимости не только накладывают ограничения на возможные пополнения, но могут использоваться также для фактического замещения неопределенных значений определенными. Может даже случиться, что для данного неопределенного значения существует единственное пополнение, удовлетворяющее множеству \mathbb{F} -зависимостей (табл. 1, табл. 2).

Таблица 1

Отношение «отпуск»

Отпуск	Служащий	Стаж	Годовой-Отпуск	Накопление
t_1	Удолл	3	21	\perp
t_2	Удолл	\perp	\perp	17
t_3	Антэнк	3	\perp	\perp

Таблица 2

Частичное пополнение отношения «отпуск»

Отпуск'	Служащий	Стаж	Годовой-Отпуск	Накопление
$t_1 = t_2$	Удолл	3	21	17
t_3	Антэнк	3	21	\perp

В последующих разделах приводятся случаи, когда необходимо использовать комбинацию нескольких отношений для заполнения неопределенных значений.

Однако, в некоторых случаях введение одного неопределенного значения \perp не позволяет осуществить устранения неопределенности для нескольких функциональных зависимостей. Для этого вводят помеченные неопределенные значения. При этом с помощью индексирования мы получаем бесконечный набор различных неопределенных значений: $\{\perp_1, \perp_2, \perp_3, \dots\}$. Предполагается, что

помеченные неопределенные значения различны, только если они имеют различные индексы. Пусть строки t_1 и t_2 со схемой R содержат помеченные неопределенные значения. Для атрибута A в R строки t_1 и t_2 согласованы на A , если выполнено одно из следующих условий:

1. $t_1(A) \downarrow$ и $t_2(A) \downarrow$ и $t_1(A) = t_2(A)$;
2. $t_1(A) = \perp_i$, $t_2(A) = \perp_j$ и $i = j$.

Строки согласованы на множестве атрибутов X в R , если они согласованы на каждом атрибуте из R . До конца этого раздела все неопределенные значения в частичных отношениях предполагаются помеченными. Каждое отношение с непомеченными неопределенными значениями может быть превращено в отношение с помеченными путем придания различных индексов непомеченным неопределенным значениям. Считается, что два отношения из $Rel \uparrow (R)$ совпадают, если одно может быть получено из другого взаимнооднозначным переименованием помеченных неопределенных значений.

Алгоритм устранения неопределенностей с использованием помеченных неопределенных значений включает следующие шаги:

Пусть даны $r \in Rel \uparrow (R)$ и множество F \mathbb{F} -зависимостей в R .

Пусть строки t_1 и t_2 в r участвуют в слабом нарушении \mathbb{F} -зависимости $X \rightarrow A$ из F .

1. Если $t_1(A) \downarrow$ и $t_2(A) = \perp_i$, то заменяем каждое вхождение \perp_i в r на $t_1(A)$.
3. Если $t_2(A) \downarrow$ и $t_1(A) = \perp_i$, заменяем каждое вхождение \perp_i в r на $t_2(A)$.
4. Если $t_1(A) = \perp_i$, $t_2(A) = \perp_j$ и $i < j$, то заменяем каждое вхождение \perp_j на \perp_i ; если $i > j$, то заменяем каждое вхождение \perp_i на \perp_j .

Основным недостатком данного алгоритма является его использование только для устранения неопределенных значений в одном отношении (универсальной схеме). Однако, БД может представлять из себя совокупность нормализованных отношений, когда

для устранения неопределенностей необходимо осуществлять соединение данных отношений.

Стратегия проверки допустимости состоит в заполнении как можно большего числа неопределенных значений с помощью информации, доставляемой \mathbb{F} -зависимостями.

3. Разработка алгоритма устранения неопределенных значений в базах данных реляционного типа на основе функциональных зависимостей

Основу приводимого далее алгоритма устранения неопределенных значений в базе данных составляет алгоритм построения замыкания множества атрибутов относительно множества ФЗ. Поэтому вначале рассмотрим данный алгоритм, приведенный в редакции [3].

Алгоритм 1

Вычисление замыкания множества атрибутов относительно множества функциональных зависимостей

Вход : $U = \{ A_1, \dots, A_n \}$ - универсальная схема;

$F = \{ Li \rightarrow Ri \mid Li, Ri \subseteq U; i = \overline{1, m} \}$ - множество ФЗ;
 $A \subseteq U$.

Выход : $A(+)$ - замыкание.

НАЧАЛО

Pred:= \emptyset ; New:=A;

ПОКА New \neq Pred делать

НАЧАЛО

Pred:=New;

ДЛЯ каждой \mathbb{F} -зависимости $Li \rightarrow Ri$ делать

ЕСЛИ $Li \subseteq$ New ТО New:=New \cup Ri

КОНЕЦ

КОНЕЦ {Замыкание}

Пример.

$U = \{ A, B, C, D, E, F \}$;

$F = \{ AC \rightarrow B, ABC \rightarrow D, ABE \rightarrow BCF, CF \rightarrow AE \}$;

$A = \{ C, F \}$.

Найти $A(+)$.

По алгоритму 2.1 построим :

$A(0) = \{ C, F \}$;

$A(1) = \{ A, C, E, F \}$;

$A(2) = \{ A, B, C, E, F \}$;

$A(3) = \{ A, B, C, D, E, F \}$;

$A(4) = \{ A, B, C, D, E, F \}$.

Отсюда, $A(+)$ = $\{ A, B, C, D, E, F \} = U$.

Докажем, что временная сложность приведенного алгоритма 1 имеет порядок $O(nm^2)$, где n - число атрибутов, m - число ФЗ. Время

выполнения цикла ДЛЖ в наихудшем случае будет иметь порядок $O(nm)$. Временная сложность оператора присваивания $\text{Pred} := \text{New}$ имеет порядок $O(1)$. Таким образом, по правилу сумм [1] временная сложность тела цикла ПОКА имеет порядок $\max(O(1), O(nm)) = O(nm)$. Прежде, чем New станет равным Pred , можно m раз реализовать цикл ПОКА. Окончательно получаем, что временная сложность алгоритма 1 имеет порядок $O(nm^2)$.

Алгоритм устранения неопределенных значений основан на том предположении, что в автоматизированной системе поддержки принятия решений для определенной предметной области имеется база данных, описывающая эталонные образы, по которым осуществляется распознавание некоторого входного образа. Отношение БД, в котором хранятся эталонные образы, является полным, в то время как отношение, в котором хранятся записи о распознаваемом образе, является частичным, т.е. содержит неопределенные значения. На атрибутах «эталонного» отношения заданы функциональные зависимости. С учетом данного предположения алгоритм устранения неопределенных значений в БД выглядит следующим образом.

Алгоритм 2.2

Устранение неопределенных значений в нормализованных базах данных реляционного типа на основе функциональных зависимостей

Шаг 1. Построить для заданного множества функциональных зависимостей замыкания всех левых частей ФЗ по алгоритму 2.1.

Шаг 2. Для каждого атрибута проверить наличие неопределенных значений \perp .

Для каждого неопределенного значения \perp найти вхождение атрибута с данным неопределенным значением в вычисленные на шаге 1 замыкания атрибутов. При этом данный атрибут не должен принадлежать левым частям ФЗ.

Для соответствующей найденному замыканию ФЗ, если значение атрибутов, стоящих в ее левых частях определены, заменить неопределенное значение в частичном отношении значением из полного отношения.

Временная сложность предложенного алгоритма определяется сложностью шага 1 и имеет порядок $O(nm^2)$.

В качестве примера рассматривается работа алгоритма устранения неопределенных значений в базе данных реляционного типа на основе функциональных зависимостей (рис. 1, рис. 2).

A	B	C
1	2	3
2	3	4
3	4	5
A	B	C
1	⊥	⊥
⊥	3	4
3	4	⊥

$$A \rightarrow B, C \rightarrow A$$

Рис. 1. Входные данные для работы алгоритма

1 шаг:

A	B	C
1	⊥	⊥
2	3	4
3	4	⊥

2 шаг:

A	B	C
1	2	⊥
2	3	4
3	4	⊥

Рис. 2. Выполнение алгоритма

В отличие от представленных в разделе 1 алгоритмов, разработанный алгоритм позволяет отказаться от помеченных неопределенных значений, имеет меньшую временную сложность и может использоваться для устранения неопределенных значений не в одном отношении, а во всей базе данных на основе выполнения операции соединения [3].

Заключение

Данная статья посвящена разработке алгоритма устранения неопределенных значений в нормализованных реляционных базах данных. Проведен анализ существующих алгоритмов устранения неопределенных значений в базах данных реляционного типа на основе функциональных зависимостей. Показано, что данные алгоритмы не всегда способны устранять неопределенные значения. Разработан алгоритм устранения неопределенных значений в нормализованных базах данных реляционного типа на основе функциональных зависимостей, в основу которого положено понятие замыкания множества атрибутов относительно множества функциональных зависимостей. Показано, что в отличие от существующих алгоритмов разработанный алгоритм позволяет отказаться от помеченных неопределенных значений, имеет меньшую временную сложность и может использоваться для устранения неопределенных значений не в одном отношении, а во всей базе данных на основе выполнения операции соединения.

Список литературы

1. Аю А. Структуры данных и алгоритмы / А. Аю, Д. Хопкрофт, Д. Ульман. – М.: Издательский дом «Вильямс», 2003. – 384 с.
2. Храмов, В. Ю. Методы и средства проектирование баз данных: монография / В. Ю. Храмов, А. И. Кустов, Э. Б. Ханов // Воронежский ЦНТИ – филиал ФГБУ "РЭА" Минэнерго России. – Воронеж, 2015. – С. 58-61.
3. Мейер Д. Теория реляционных баз данных / Д. Мейер. – М.: Мир, 1987. – 608 с.

Алгоритм распределения канальных ресурсов в сетях когнитивного радио

М. И. Игнатов

Студент магистр

А. Ю. Савинков

Профессор

Введение

Развитие радиовещания, цифрового телевидения и беспроводных систем связи происходит столь стремительно, что все более остро встает вопрос об эффективности использования частотного спектра.

Современные сетевые технологии обычно не имеют способность адаптироваться к различным изменяющимся условиям, что может привести к неоптимальной производительности. Чтобы рационально распределить нагрузку на каналы, оптимизировать работу сети и повысить коэффициент использования спектра необходимо применить в радиосистеме особую технологию, которая сможет обеспечить постоянную занятость каналов связи таким образом, чтобы не пострадало качество связи абонентов которые привязаны к определенному частотному диапазону.

В настоящее время в связи с бурным развитием технологий беспроводной связи, а также существующей потребностью организации устойчивой мультисервисной сети связи из большого количества радиостанций, имеется ряд актуальных задач: во-первых, возникает проблема генерации настроечных данных для радиостанций таким образом, чтоб обеспечить электромагнитную совместимость радиостанций, во-вторых – ввода этих настроечных данных в радиостанции за приемлемое время и с минимальными усилиями со стороны пользователей и администраторов сети связи, в-третьих – существует проблема с тем, чтобы при выходе из строя одного из узлов системы остальные узлы продолжали функционировать и в-четвертых использования ресурса пропускной способности сети для совместной передачи трафика различного вида.

Распределение спектра основывается на выделении конкретного диапазона частот для конкретной услуги. Однако, большая часть

выделенного диапазона радиочастот используется от случая к случаю, что приводит к неэффективному использованию частотного ресурса. Проблема неэффективного использования спектра может быть решена за счет новой системы доступа к лицензированным полосам частот в которых работают существующие (так называемые первичные) пользователи. Такой новой системой доступа является динамический доступ к спектру [1]. Основной технологией, использующей динамический доступ к спектру, является когнитивное радио, которое дает возможность получить доступ к беспроводному каналу наравне с первичными пользователями. Термин «когнитивное радио» был впервые введен Митолой и Магуайром в [2]. В данной статье когнитивное радио определяется как радио, которое воспринимает окружение, в котором оно находится, и в результате может адаптировать параметры связи в соответствии с этим пониманием. В настоящее время, термин «когнитивное радио» приобрел более широкое значение.

1. Модель сети

Для того, чтобы получить доступ к радиочастотному спектру, в котором работают лицензированные (первичные) пользователи, не нарушая их прав, с требуемым качеством обслуживания, каждый пользователь когнитивного радиоустройства в сетях когнитивного радио должен:

- определить доступную часть спектра;
- выбрать лучший из доступных каналов;
- скоординировать доступ к этому каналу с другими пользователями;
- освободить канал, когда возобновит работу лицензированный пользователь.

Одной из основных задач реализации такого подхода является выбор канала приёма/передачи.

Рассмотрим представленную на рисунке 1 модель беспроводной сети, представляющую собой совокупность приемников и передатчиков, образующих пары.

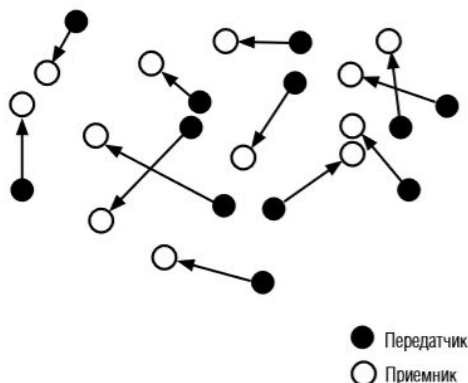


Рис. 1. Беспроводная модель приемопередающих пар

В данной модели все приемопередающие пары находятся в одном кластере, т. е. передача информации от передатчика к приемнику осуществляется напрямую без использования промежуточных узлов. В сети может использоваться определенное количество каналов связи для передачи информации от передатчика к приемнику. При этом каждая приемопередающая пара может использовать не более одного канала связи.

Нужно адаптировать приемопередающие пары таким образом, чтобы каналы связи для конкретных пар были оптимальными. Для этого используем предложенную в [3] целевую функцию:

$$U_i(a_i, a_{-i}) = \sum_{k=1, k \neq i}^n p_k g_{ki} f(a_k, a_i) + \sum_{k=1, k \neq i}^n p_i g_{ik} f(a_i, a_k) \quad (1)$$

где a_i - канал связи, выбранной i -той парой, a_{-i} - каналы связи, выбранные остальными парами, p - мощность передатчика, g - коэффициент передачи радиочастотного тракта между каким-либо передатчиком и приёмником, f - функция, принимающая значение, равное 1, если её аргументы – выбранные каналы связи двух различных пар приёмников и передатчиков – совпадают, и равное 0, если не совпадают. Таким образом, целевая функция (1) для i -той пары представляет собой сумму помех, наводимых остальными передатчиками на i -м приёмнике, и помех, наводимых i -м передатчиком на остальных приёмниках. Критерием выбора канала связи для данной целевой функции является её минимум.

2. Алгоритм и результаты моделирования

Для имитационного моделирования распределения канальных ресурсов в сети приемопередающих пар на основе модели с целевой функцией (1) была разработана и реализована программа (рисунок 2) на языке Python.

Алгоритм заканчивает работу и распределение каналов приемопередающих пар считается окончательным тогда, когда в очередном раунде ни одна из пар не поменяет выбранный в предыдущем раунде канал связи.

Моделирование проводилось при следующих значениях параметров: количество приемопередающих пар $n = 5 - 50$; количество каналов связи $m = 3 - 10$; приемопередающие пары распределены случайным образом на плоскости размером $400 \times 400 \text{ м}^2$; мощности всех передатчиков равны 1 Вт; исходное распределение каналов связи между приемопередающими парами также случайно.

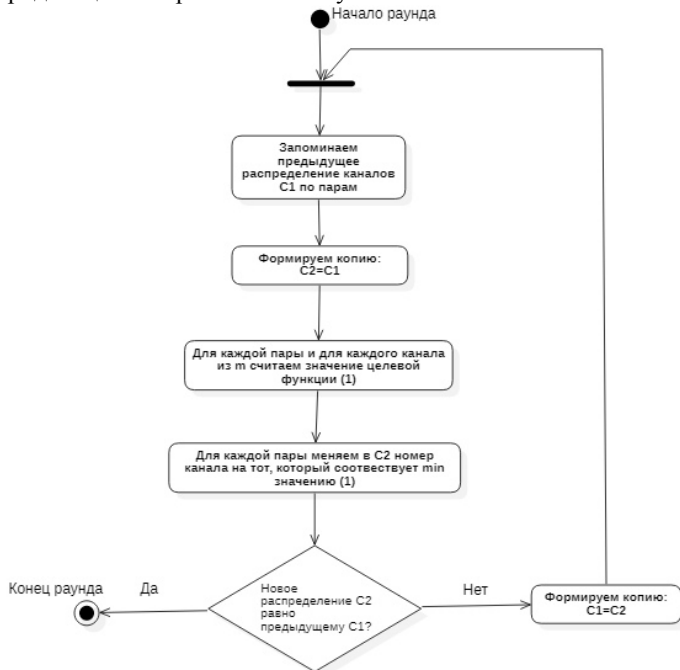


Рис. 2. Диаграмма состояний алгоритма распределения каналов.

На рисунке 3 представлены распределения каналов связи среди приемопередающих пар до и после работы алгоритма для случая $n = 10$, $m = 4$. Оси отражают расположение приемопередающих пар на плоскости, первичный и адаптированный каналы связи указаны в скобках у каждой пары.

Для качественной оценки адаптации каналов приемопередающих пар в соответствии с предложенным алгоритмом использовалось отношение сигнал/шум для приёмника каждой пары (в дБ). Исходное распределение каналов для 10 пар и соответствующие отношения сигнал/шум (левые столбцы), а также отношения сигнал/шум после адаптации каналов (правые столбцы) представлены на рисунке 4. Можно отметить, что при заданных условиях моделирования отношение сигнал/шум практически для каждой пары выросло в среднем на 3-5 дБ, а работа алгоритма заканчивается через 3 раунда.

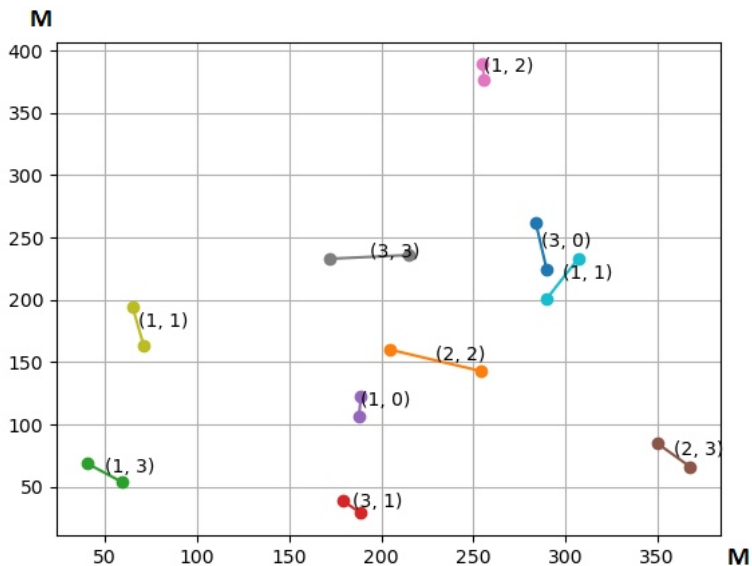


Рис. 3. Распределение каналов связи приемопередающих пар на плоскости

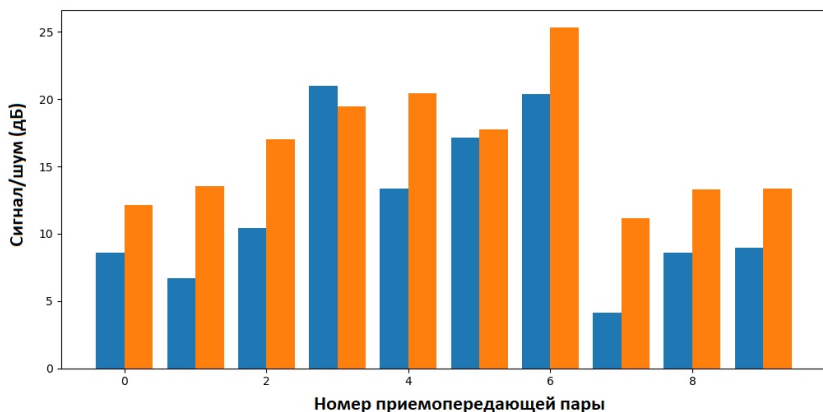


Рис. 4. Отношение сигнал/шум до и после работы алгоритма

Заключение

В данной статье рассмотрен алгоритм распределения каналов связи в сетях приемопередающих пар, в основе которого лежит минимизация уровня помех на приёмниках. Так же приведены результаты моделирования распределения каналов связи на основе рассмотренного алгоритма. Реализация алгоритма и последующее моделирование позволили выявить следующие аспекты его функционирования: алгоритм чувствителен к первичному распределению каналов, что отражается в виде увеличения/уменьшения отношения сигнал/шум после адаптации каналов связи; при больших значениях n (более 15) и m (более 5-8) для некоторых приемопередающих пар наблюдается значительное ухудшение отношения сигнал/шум (15-30 дБ); для n не более 30 алгоритм заканчивает работу в среднем за 3-6 раундов.

Рассмотренный алгоритм может быть использован в совокупности с другими методами оценки и выбора каналов связи, в частности, в перспективных направлениях построения сложных алгоритмов обработки информации с помощью аппарата искусственных нейронных сетей (НС) [4, 5].

Список литературы

1. Doyle L. Essentials of Cognitive Radio. Cambridge University Press, 2009. — 252 p.
2. Joseph Mitola III. Cognitive Radio. An Integrated Agent Architecture for Software Defined Radio. Dissertation ... of Doctor of Technology. Sweden, Stockholm: Royal Institute of Technology (KTH), 2000 – 313 p.

3. Ошмарин Д.В. Распределение канальных ресурсов в сетях когнитивного радио на основе теории игр.- Бизнес-информатика. – 2010. – №4 (14). – С. 38-45.

4. Кирсанов Э. А., Сирота А. А. Обработка информации в пространственно-распределенных системах радиомониторинга: статистический и нейросетевой подходы. – М. : Физматлит, 2012. – 344 с.

5. Фомин Л. А., Будко П. А., Шлаев Д. В. Нейрофизиологическая модель управления распределенной системой // Нейрокомпьютеры: разработка, применение. – 2008. №7. – С. 62-68.

Система контроля обработки производственных заказов

Ю. И Лещева

Студент магистр

И. В. Илларионов

Доцент

Введение

Система SAP состоит из набора прикладных модулей, которые поддерживают различные бизнес-процессы компании и интегрированы между собой в масштабе реального времени. Границы модулей в значительной степени условны, между ними происходит обмен данными, могут быть сделаны общие настройки и отчеты [1].

Модуль SAP PP является одним из основных модулей системы SAP ERP. SAP PP расшифровывается как производственное планирование. Этот модуль SAP ERP предназначен для планирования и выполнения производственных бизнес-процессов в организации, а также контроля производственной деятельности предприятия. Модуль SAP PP обрабатывает основные данные, необходимые для операции спецификаций, рабочего центра и маршрутизации, и хранит их в отдельном компоненте [2].

Многие системы могут быть основаны на стандартном программном обеспечении SAP R/3. Рассмотрим систему MA-VL, в которой задействованы модули SAP PP и MM. Система MA-VL – это полностью автоматическая система, контролирующая обработку заказа, в которую пользователь не вмешивается. Большая часть работы данной системы происходит за счет вызова транзакций модуля PP [3, 4].

1. Основные данные модуля SAP Production Planning

Модуль SAP PP обрабатывает основные данные, необходимые для операции спецификаций, рабочего центра и маршрутизации, и хранит их в отдельном компоненте. Моделирование производственных процессов в SAP PP начинается с основных данных [5]. Без основных данных модуля PP нельзя выполнять и хранить какие-либо транзакции. Основными данными являются:

1. Ведомость материалов.
2. Мастер материалов.

3. Рабочий центр.
4. Маршрутизация.

Рассмотрим ключевые этапы выполнения производственного заказа. Он начинается с планового заказа и заканчивается поступлением материала против производственного заказа:

1. Преобразовать плановый заказ в производственный заказ. Первым шагом является преобразование планового заказа в производственный заказ. При создании производственного заказа тип определяется в системе SAP PP.

2. Выпустить производственный заказ. Чтобы начать производственный процесс, необходимо оформить производственный заказ. Пока не выпущен производственный заказ, выполнение производственного процесса не может начаться.

3. Выпуск товаров для производственного заказа. Чтобы выполнить производственный заказ, необходимо оформить товар. После выпуска товара номер документа может быть обновлен в системе.

4. Подтверждение производственного заказа. Все подпроцессы выполняются в соответствии с необходимыми операциями для подтверждения производства согласно производственному заказу.

5. Хорошие поступления по производственному заказу. По завершении выполнения производственного заказа товары, произведенные по производственному заказу, принимаются и помещаются в зону хранения.

2. Интеграция модуля SAP PP и модуля Material Management

Для выполнения операций планирования и контроля производства SAP PP тесно интегрирован с модулем MM. SAP MM – один из модулей, разработанный для облегчения следующих бизнес-процессов на предприятии:

1. Основные данные поставщика и основные данные материала.
2. Планирование на основе потребления.
3. Покупка.
4. Управление запасами.
5. Оценка материалов.
6. Подтверждение счета.

SAP PP и SAP MM:

- включает в себя перемещение товаров, получение товаров;
- включает создание резервирования для производственного заказа / планирования;
- повышение заказов на покупку;
- котировки и выбор поставщиков.

3. IDOC

IDOC – это просто контейнер данных, используемый для обмена информацией между любыми двумя процессами, которые могут понимать синтаксис и семантику данных. Другими словами, IDOC похож на файл данных с указанным форматом, которым обмениваются две системы, которые знают, как интерпретировать эти данные.

Структура IDOC состоит из 3 частей:

- административная часть (англ. Control Record) – которая содержит тип IDOC контейнера, тип сообщения, текущий статус, отправителя, получателя и так далее. Это называется контрольной записью;
- данные приложения (англ. Data Record) – которые содержат данные. Они называются записями данных или сегментами;
- информация о статусе (англ. Status Record) – предоставляет информацию о различных этапах, через которые прошел IDOC.

4. Автоматическая система обработки заказов MA-VL

Система MA-VL – это полностью автоматическая система, контролирующая обработку заказа, в которую пользователь не вмешивается. MA-VL принимает и обрабатывает заказы от CRM-систем.

Как видно на рис. 1 CRM-система отправляет готовый к производству заказ в MA-VL. Заказ состоит из любого количества позиций заказа. Каждая позиция заказа относится только к одному рыночному продукту.

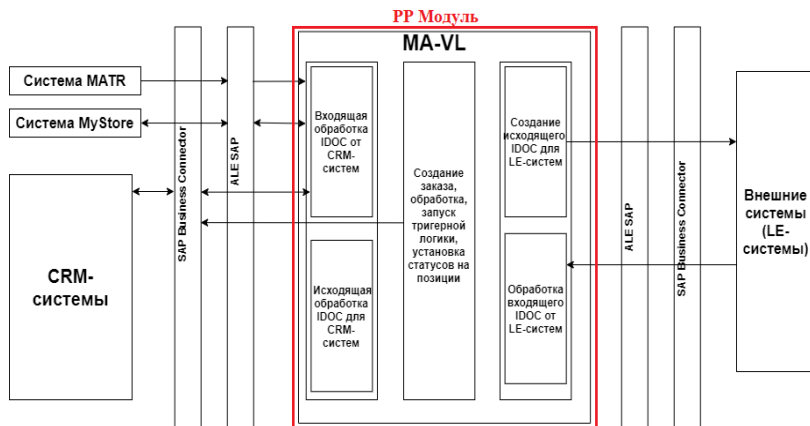


Рис. 1. Архитектура системы MA-VL

MA-VL принимает и обрабатывает заказы от CRM-систем. Система преобразует рыночные продукты, переданные от CRM-систем, в готовый заказ для клиента, связывает предварительные продукты и передает заказы соответствующим внешним системам. Результат обработки отправляется в CRM-систему в виде IDOC. Если какой-то элемент заказа или весь заказ не могут быть созданы в MA-VL, в CRM-систему заказов отправляется отрицательное сообщение в виде IDOC. В этом случае соответствующие позиции заказа не доступны в MA-VL. Полученный IDOC имеет статус ошибки. Если MA-VL смогла создать свои заказы, в CRM-систему отправляется положительное подтверждение для каждой позиции заказа.

MA-VL отправляет заказ в различные внешние системы ровно один раз через WebSphere MQ. WebSphere MQ (англ. Message Oriented Middleware) – промежуточное ПО обмена сообщениями. Оно позволяет независимым и, возможно, работающим не одновременно, приложениям в распределённой системе обмениваться данными друг с другом. Принимающая система несет полную ответственность за дальнейшую обработку, а также повторную обработку заказа. Сотрудники бэк-офиса не должны вносить изменения в позиции заказов, поступившие во внешние системы через MA-VL. Исключением здесь является, например, желаемая дата клиента. Нельзя гарантировать, что последовательность заказов в системах будет поддерживаться. Однако MA-VL обнаруживает неправильные последовательности и сообщает об ошибке. Информация, которая передается в LE-системы, зависит от определения продукта в системе MATR. В этой системе для каждого материала определяется, какие LE-системы участвуют в выполнении заказа. Взаимодействие между MA-VL и LE-системами сопровождается различными сообщениями, которые информируют о состоянии заказов в системах. В MA-VL установлен порядок управления вызова LE-систем, называемый триггером, на основе ответов между системами. Сообщения передаются в системы по мере необходимости.

MATR – это центральная система, хранящая данные о продуктах и об их поставщиках. Доставка данных продукта из MATR в MA-VL происходит через соединение SAP-SAP через ALE. SAP ALE (англ. Application Link Enabling) – технология обмена данными, разработанная компанией SAP AG. Технология, потому что это набор инструментов, протоколов, форматов, которые позволяют обмениваться данными в режиме реального времени или оффлайн режиме между SAP и не SAP-системами. Формирование и структура продуктов, поставляемых MATR, определяют процессы в MA-VL. Следовательно, базовая структура

структуры продукта не должна изменяться, поскольку такие изменения могут помешать обработке в MA-VL.

Система MyStore используется для поддержки внедрения и работы с CRM-системами. С помощью этого инструмента заказы могут быть технически выполнены в SAP-системах или заказы можно откатить. Инструмент MyStore реализован как портал SAP и, таким образом, может быть вызван как веб-приложение в браузере.

Система MA-VL использует следующие основные данные:

1. Составные части материала – это компоненты, из которых состоит продукт, пришедший в заказе.

2. Рабочие места – это список систем, куда должны быть отправлены материалы для обработки.

3. Рабочий план показывает, какие действия нужно сделать с компонентами, чтобы получить заказанный продукт. Каждый рабочий план состоит из нескольких компонентов основного материала в заказе. В рабочем плане с помощью рабочего места записывается, какая конкретная внешняя система должна быть запущена.

5. Взаимодействие системы MA-VL и модуля PP

Рассмотрим общий режим взаимодействия модуля PP, CRM-систем и системами, исполняющими заказ. LE-системы получают заказы от MA-VL. На рисунке представлена схема взаимодействия модуля PP с интерфейсами, обрабатывающими IDOCs от систем и общие процессы, происходящие в модуле.

Из рис. 2 видно, что заказы обрабатываются во входящих интерфейсах и сохраняются в модуле SAP R / 3 PP-Core как производственные заказы или сети производственных заказов.

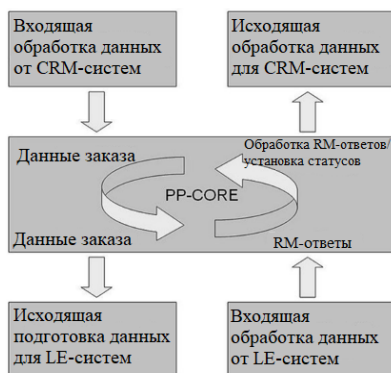


Рис. 2. Процессы модуля PP, используемые в MA-VL

После успешной установки PP-ядро запускает исходящие интерфейсы (интерфейсы LE-систем). Входящие интерфейсы запускаются входящими пакетами данных, выходящие интерфейсы запускаются ядром PP. В зависимости от соглашения об интерфейсе и стадии заказа LE-системы отправляют соответствующие сообщения. Ответы, в свою очередь, обрабатываются в модуле PP и могут запускать определенные действия. CRM-система может быть проинформирована об этих ответах. Как только LE-системы начинают или завершают обработку своих материалов, которые входят в заказ, происходит установка статусов в заголовке заказа или на уровне процесса заказа в PP модуле.

Заключение

Не все компании решают внедрить модуль SAP PP, поскольку он необходим только в том случае, если компания выполняет производственную деятельность. Однако SAP PP предоставляет возможности для эффективного управления производственными объектами и процессами. Система MA-VL эффективно осуществляет обработку различных заказов на основе модуля PP. Получив заказ, система, используя возможности модуля PP решает, как обрабатывать позиции, делит их на сети и отправляет позиции заказа в разные LE-системы. Системы обмениваются ответами в виде IDOC. Каждая система сообщает MA-VL может ли она обработать материалы, используя различные виды ответов. Таким образом происходит полностью автоматическая обработка и контроль выполнения заказа, в которую пользователь не вмешивается.

Список литературы

1. Вольфганг В. Разработка приложений SAP R 3 на языке ABAP4 / В.Вольфганг, Р.Кречмер – М.:Лори, 1998. – 335 с.
2. Вивек К. Внедрение SAP R/3: Руководство для менеджеров и инженеров / К. Вивек. – М. : Компания АйТи, 2006. – 511 с.
3. Kühnhauser K. Discover ABAP, 2nd edition / K. Kühnhauser, T. Franz. – SAP Press, 2012. – 533 p.
4. Akhtar J. Production Planning and Control with SAP ERP, 2nd edition / J. Akhtar. – SAP Press, 2016. – 940 p.
5. Murray M. Materials Management with SAP ERP: Functionality and Technical Configuration, 2nd edition / M. Murray – SAP Press, 2006. – 561 p.

Алгоритм построения скелета с помощью графа смежности для бинарных изображений

А. Д. Ильина

Студент магистр

В. В. Фертиков

Доцент

Введение

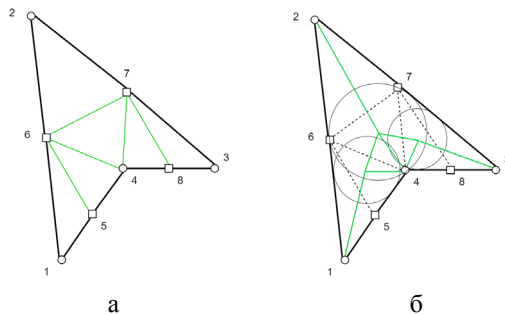
Рассматривается задача построения скелета многоугольной многосвязной фигуры – замкнутой области, граница которой составлена конечным числом простых многоугольников. Исходные данные алгоритма представлены бинарным изображением, которое может содержать произвольное число объектов-фигур. В результате решения задачи для каждой из фигур мы получаем непрерывный скелет – множество срединных осей фигуры.

Данный алгоритм основан на изложенном в [1] подходе, где предлагается строить скелет с помощью двойственного к диаграмме Вороного графа смежности сайтов, образующих границу фигуры. Оценка сложности алгоритма составляет $O(n \log n)$ в худшем случае, где n – общее число вершин фигуры.

1. Граф смежности многоугольной фигуры

Для дальнейшего описания предлагаемого алгоритма требуется определить некоторые основные понятия. Отличительной особенностью алгоритма является применение графа смежности сайтов (графа Делоне), который, в свою очередь, необходим для построения диаграммы Вороного фигуры.

Граница фигуры может быть разделена на множество объектов, называемых сайтами. *Сайтом* фигуры является любая её вершина (сайт-точка) или сторона (сайт-сегмент). Граф смежности – это планарный граф, у которого множество вершин состоит из всех сайтов фигуры, а множество рёбер содержит пары смежных сайтов. Два сайта являются *смежными*, если они имеют непустое пересечение либо общую пустую окружность [2]. На рис. 1а продемонстрирован пример графа смежности; цифрами обозначены сайты фигуры.



а – граф смежности, б – диаграмма Вороного

Рис. 1. Пример построения графа смежности и диаграммы Вороного фигуры

Между графом смежности и диаграммой Вороного существует взаимно однозначное соответствие [3]. Для получения диаграммы Вороного достаточно построить максимальные пустые окружности всех пар сайтов в графе смежности, а затем соединить их центры, как показано на рис. 1б. При этом известно, что соединительными линиями могут быть отрезки прямых или квадратичных парабол. В свою очередь, скелет является подмножеством диаграммы Вороного. Таким образом, для решения конечной задачи построения скелета достаточно определить граф смежности фигуры.

2. Обзор алгоритма

В ходе выполнения предлагаемого алгоритма возникает необходимость решения следующих подзадач:

- Прослеживание границ и выделение сайтов фигуры.
- Построение графа смежности.
- Переход от графа смежности к диаграмме Вороного.
- Построение на основе диаграммы Вороного непрерывного скелета фигуры.

Для решения первой подзадачи производится просмотр всего изображения, а затем симплексное прослеживание границы фигуры на основе граничного коридора [1], в результате чего граница представляется как множество вершин и соединяющих их прямых отрезков. Эти вершины и отрезки принимаются за сайты фигуры.

Далее осуществляется построение графа смежности методом «разделяй и властвуй», как будет показано ниже. Имея граф смежности фигуры, можно за линейное время перейти к её диаграмме Вороного

при помощи построения пустых окружностей и анализа вида смежных сайтов.

Последним шагом является переход непосредственно к скелету изображения, что также осуществляется за линейное время. Всё это позволяет сделать оценку сложности алгоритма в $O(n \log n)$, поскольку именно такой сложностью обладает используемый алгоритм построения графа Делоне [2].

3. Построение графа смежности

Решение данной проблемы является ключевой точкой всего алгоритма. Оно основывается на разбиении множества сайтов фигуры на более мелкие подмножества, для которых можно легко построить графы смежности. При этом выполняются следующие действия:

1. Множество всех сайтов фигуры разделяется на элементарные цепочки. Под элементарной цепочкой будем иметь в виду любую выпуклую вершину многоугольника, стороны, имеющие только выпуклые соседние вершины, а также цепочки из вогнутых вершин и инцидентных им сторон. На рис. 2а такими цепочками являются множества сайтов $\{1\}$, $\{2\}$, $\{3\}$, $\{6\}$, $\{7\}$, $\{4, 5, 8\}$.

2. Поскольку цепочки сайтов рассматриваются последовательно друг за другом, последний сайт каждой цепочки является смежным первому сайту следующей цепочки. Поэтому на втором шаге эти сайты соединяются ребром (рис. 2б). По завершении обхода количество цепочек уменьшается в два раза.

3. На последующих этапах работы алгоритма выполняется итеративное построение графов смежности из соседних пар цепочек сайтов, для чего применяется условие Делоне. Условие Делоне проверяется с помощью описанных окружностей [4]. Если пара сайтов удовлетворяет условию, они соединяются ребром. Кроме того, на каждой новой эпохе условие Делоне проверяется для уже существующих рёбер, и при его нарушении конфликтующее ребро разрушается.

Работа алгоритма останавливается, когда остаётся лишь одна цепочка из сайтов (рис. 2 г). В этом случае граф смежности построен для всей фигуры, и мы можем перейти к диаграмме Вороного, как уже было описано выше.

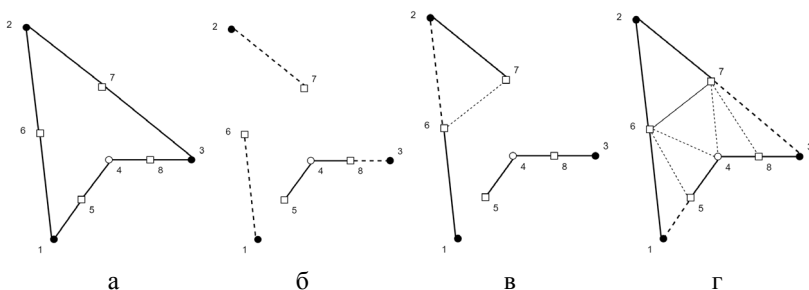


Рис. 2. Эпохи построения графа смежности

4. Переход к непрерывному скелету

Наконец, имея диаграмму Вороного фигуры, мы можем построить её скелет. Для этого достаточно лишь убрать из диаграммы «лишние» элементы – а именно, все рёбра, проходящие через вогнутые вершины фигуры (те, для которых угол между сторонами больше 180°). В примере на рис. 3 показан непрерывный скелет фигуры. Для его получения из диаграммы Вороного (рис. 1б) были убраны рёбра, инцидентные вогнутой вершине 4.

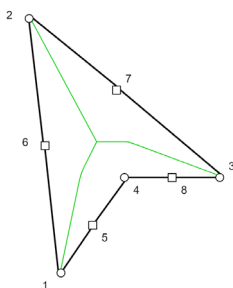


Рис. 3. Непрерывный скелет фигуры, полученный из диаграммы Вороного

5. Развитие алгоритма

Скелеты фигур имеют весьма обширный круг применений, подробно описанный в [1]. Для большинства из них могут быть использованы уже имеющиеся результаты работы алгоритма. Тем не менее, интерес также представляют некоторые направления развития изложенного здесь подхода.

Первое из них состоит в построении циркулярных фигур (циркуляров) на основе скелетов. Для получения циркуляра необходимо

в каждой точке скелета определить дистанционную функцию, значение которой соответствует расстоянию от точки скелета до границы фигуры. Такое описание поможет оценивать не только форму фигуры, но и её размеры.

Второе направление заключается в обобщении алгоритма для трёхмерных изображений. Это достаточно перспективная область, поскольку подобный алгоритм позволит адекватно оценивать форму объёмных объектов, обладая при этом достаточно высокой вычислительной эффективностью.

Заключение

Описанный в данной статье алгоритм является эффективным средством, позволяющим решить поставленную задачу – построение скелета бинарного изображения. Результаты работы алгоритма могут быть применены в более сложных прикладных задачах, таких как распознавание рукописных текстов, идентификация биометрических данных, маршрутизация. Дальнейшее развитие алгоритма можно осуществлять в двух основных направлениях: переход к циркулярным фигурам и расширение алгоритма на трёхмерный случай.

Список литературы

1. Местецкий Л.М. Непрерывная морфология бинарных изображений: фигуры, скелеты, циркуляры. – М.: ФИЗМАТЛИТ, 2009. – 288 с.
2. Местецкий Л.М. Скелетизация многосвязной многоугольной фигуры на основе дерева смежности её границы // Сиб. журн. вычисл. математики / РАН. Сиб. отд-ние. – Новосибирск, 2006. – Т. 9, № 3– С. 299-314
3. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение: Пер. с англ. С.А. Вичеса, М.М. Комарова п. ред. Ю.М. Баяковского– М.: Мир, 1989. – 478 с., с. 255-257
4. Скворцов А.В., Мирза Н.С. Алгоритмы построения и анализа триангуляции. – Томск: Изд-во Том. Ун-та, 2006. – 168 с.

Модуль сбора, анализа и загрузки информации как подсистема ситуационного центра

К. А. Казаков

Студент магистр

Н. К. Самойлов

Старший преподаватель

Введение

В последнее время вследствие бурного развития социально-политических отношений в управленческой сфере (в частности, в органах государственной власти) повысились объем и динамика информационного обмена, а также проявилась необходимость в быстром и своевременном реагировании на возникающие ситуации. Таким образом, можно сделать вывод о том, что проблема повышения эффективности решения управленческих задач на сегодняшний день является весьма актуальной. Идея комплексного управления с опорой на системы автоматизации обусловлена выработкой комплексного подхода к различным управленческим вопросам и необходимостью принятия решений в условиях дефицита времени. Воплощением этой идеи стало информационно-аналитическое и организационное решение, получившее название «Ситуационный центр» [1].

Создание ситуационных центров, предназначенных для использования в органах государственной власти Российской Федерации, является одной из приоритетных задач по модернизации государственного управления. Для таких ситуационных центров характерны следующие особенности: масштабность решаемых задач, уклон в стратегическое управление, необходимость интеграции большого количества информационных источников, наличие обширной нормативной базы, высочайшие требования к уровню информационной безопасности. Целью настоящей работы является рассмотрение подхода к построению автоматизированной системы «Ситуационный центр» для регионального уровня управления, в рамках которого для системы в целом присущи все вышеперечисленные особенности, при этом модулю сбора, анализа и загрузки информации отведена важная и основополагающая роль.

1. Анализ существующих решений

В рамках реализации стратегии информационного развития Российской Федерации для различных регионов страны были успешно разработаны и сданы в эксплуатацию ситуационные центры органов государственной власти. В число этих регионов входят Сахалинская область, Республика Саха (Якутия).

Согласно [2], в конфигурацию системы «Ситуационный центр губернатора Сахалинской области» входят следующие компоненты: подсистема анализа и структурирования информации, подсистема сбора информации, подсистема передачи информации, подсистема визуализации информации, подсистема хранения информации, подсистема обеспечения безопасности информации. При этом стоит отметить, что наибольшее внимание системный интегратор (компания Polymedia – <https://www.polymedia.ru/>) уделил подсистеме визуализации информации, вследствие чего в систему было внедрено программное обеспечение «Визуализация информации на распределенных дисплеях» (внутренний продукт Polymedia), на основе которого решаются задачи разработки сценариев отображения информации, оперативного управления сценариями и режимами отображения, а также демонстрации аналитических и медиа-данных в удобном для восприятия виде [3].

Согласно [4], системы «Ситуационный центр Главы Республики Саха (Якутия)» входят следующие компоненты: подсистема загрузки данных, подсистема представления данных, подсистема формирования аналитических панелей, подсистема администрирования. При этом стоит отметить, что основной опцией системы является построение отдельных веб-страниц для визуализации информации в табличном, графическом и картографическом виде; в системе также реализована возможность подключения геоинформационных ресурсов с таких платформ, как ArcGIS (<https://www.arcgis.com/>), Панорама (<https://gisinfo.ru/>), GeoServer (<http://geoserver.org/>).

Для вышеперечисленных программных продуктов характерны определенные минусы. Так, первым из недостатков описываемых решений является то, что процессы сбора, анализа и загрузки данных реализованы в различных модулях ситуационного центра, что делает его архитектуру более сложной, громоздкой, менее отказоустойчивой. Также к минусам данных информационно-аналитических систем можно отнести большие затраты (как временные, так и трудовые) на должное обеспечение информационной безопасности системы. Ещё одним недостатком является необходимость в реализации большого количества способов загрузки данных, так как формы этих данных могут быть

различными (при этом стоит отметить, что это также сказывается на возможностях системы к масштабированию). Таким образом, весьма удобным и практичным представляется решение, в рамках которого управление информационными потоками в ситуационном центре осуществляется в рамках одной подсистемы.

2. Функционирование ситуационного центра при наличии модуля сбора, анализа и загрузки информации

В общем виде модуль сбора, анализа и загрузки информации представляет собой реализацию методов по переносу исходных данных из источников информации в специальное аналитическое приложение или хранилище данных. Этот модуль способствует преобразованию исходных данных в определенный универсальный формат, получению из этих данных обобщенной информации. Как следствие, значительно упрощается последующий процесс аналитической обработки полученной информации, она становится пригодной для визуализации (представления в виде, удобном для человеческого восприятия).

Описываемый модуль должен обеспечивать корректное выполнение следующих этапов процесса переноса данных в систему: **извлечение данных** из одного или нескольких источников, **преобразование данных** (конвертация к требуемому формату данных, их обобщение и очистка), **загрузка данных** (запись в соответствующую систему хранения). Перемещение данных в этом процессе можно представить в виде функциональной схемы, представленной на рис. 1.

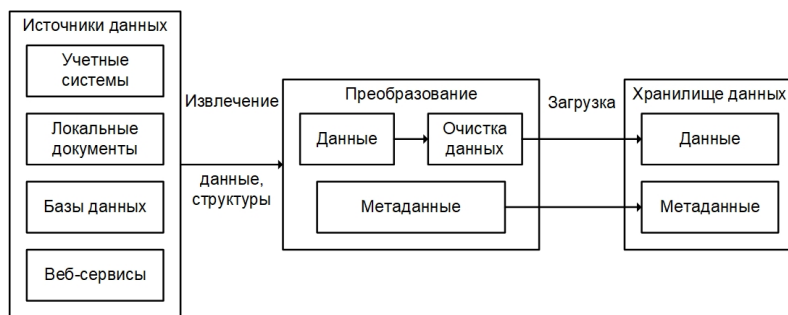


Рис. 1. Схема функционирования модуля сбора, анализа и загрузки информации

Согласно [2], к важнейшим правилам корректного построения и функционирования ситуационных центров относятся принципы комплексного единства, комплексной безопасности, наглядности представления информации. При этом каждая из компонент системы

(в том числе и модуль сбора, анализа и загрузки информации) должна учитывать требования всех остальных компонент и не мешать их развитию и масштабированию. Таким образом, основополагающими компонентами ситуационного центра являются модуль администрирования (обеспечения безопасности информации), модуль представления данных, модуль сбора, анализа и загрузки информации, модуль сбора отчетности, а также отдельная база данных (в рамках вышеописанного процесса переноса информации она является ничем иным, как хранилищем данных). Исходя из вышесказанного, функционирование ситуационного центра можно представить в виде контекстной диаграммы, отображенной на рис. 2.

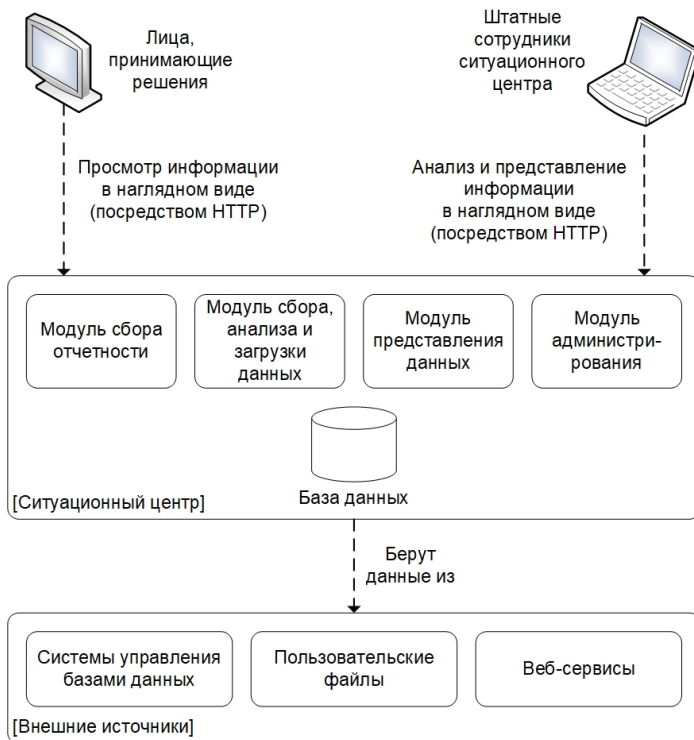


Рис. 2. Контекстная диаграмма системы «Ситуационный центр»

Описываемая архитектура ситуационного центра обладает достаточно большой степенью гибкости и масштабируемости (то есть

при необходимости в систему можно добавить новые модули, тем самым существенно расширив ее функциональность), соотносится с принципом стандартизации (в рамках системы используется единый формат данных), обеспечивает замкнутость информационного контура и должный контроль над протекающими в системе информационными процессами, а также позволяет значительным образом сократить временные и трудовые затраты на разработку ситуационного центра.

Заключение

В рамках данной статьи была рассмотрен подход к построению ситуационных центров, идея которого заключается в работе модуля сбора, анализа и загрузки информации. Реализация данного модуля в рамках ситуационного центра позволяет должным образом контролировать протекающие в системе информационные процессы, а также повысить эффективность работы системы.

Список литературы

1. Трунович, А. С. Ситуационный центр как инструмент управления / А. С. Трунович // Jet Info. – 2017. – №7-8 (284). – С. 26-31.
2. Ильин, Н. И. Ситуационные центры. Опыт, состояние, тенденции развития / Н. И. Ильин, Н. Н. Демидов, Е. В. Новикова. – Москва : МедиаПресс, 2011. – 336 с.
3. Вахмянин, И. С. Разработка модели алгоритма управления информационными потоками в ситуационных центрах органов государственной власти / И. С. Вахмянин, Н. И. Ильин, Е. В. Новикова. – Бизнес-информатика. – 2011. – № 1 (15). – С. 3-10.
4. Руководство по работе в информационно-аналитической системе «Ситуационный центр Главы Республики Саха (Якутия)» [Электронный ресурс]. – Режим доступа : http://src-sakha.ru/medias/docs/is/manual_sc.pdf

Новые аспекты колебаний балки с учетом распределенных гистерезисных свойств

Е. А. Карпов

Студент магистр

М. Е. Семенов

Профессор

Введение

В настоящее время проектирование несущих конструкций различных инженерных сооружений, а также разработка адекватных физических моделей является неотъемлемой частью строительного процесса. Известно, что одним из базовых и важнейших элементов любых несущих конструкции является балка [1].

Одним из важнейших этапов моделирования динамики балки является анализ ее поведения под воздействием внешних возбуждений (нагрузок) [2]. Обычно исследуемые нагрузки классифицируются следующим критериям: по характеру приложения (сосредоточенные и распределенные), по продолжительности во времени (переменные и постоянные), по характеру действия (статические и динамические). В зависимости от типа нагрузки, проявляются те или иные свойства балки, являющиеся, в свою очередь, следствием внутренних особенностей используемых материалов. Поэтому важно еще на этапе проектирования и моделирования исследовать физические особенности балок, изготовленных из разных материалов.

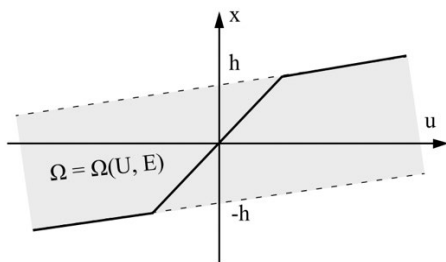
Один из подходов, позволяющих моделировать особенности материалов, основывается на теории упругопластического гистерезиса [3]. Обычно для этих целей рассматриваются следующие конструктивные модели: преобразователь Прандтля, преобразователь упор, преобразователь Прандтля–Ишлинского. Их моделирование обычно осуществляется на языке нелинейных операторов, разработанных в фундаментальной монографии М. А. Красносельского и А. В. Покровского [4].

Настоящая работа посвящена исследованию модели колебаний балки с учетом гистерезисных свойств, формализуемых в рамках конструктивного подхода моделью Прандтля–Ишлинского, а также срав-

нению реакций классической и гистерезисной балок на нагрузку сейсмического типа.

1. Уруголастический гистерезис: преобразователь Прандтля–Ишлинского

Одним из самых простых гистерезисных преобразователей в рамках теории уруголастического гистерезиса является преобразователь Прандтля (см. рис. 1). В основном он используется для моделирования уруголастических волокон материала, где нет необходимости учитывать дислокации, возникающие в кристаллической структуре материала. Кроме того, преобразователь Прандтля используется как элементарная составляющая в сложных континуальных гистерезисных моделях.



преобразователь Прандтля с пороговым значением равным h и областью определения h в виде полосы с углом наклона, зависящим от коэффициента E

Рис. 1. График зависимости выхода $u(t)$ от входа $x(t)$

Преобразователь Прандтля имеет две характеристики: h – пороговое значение – и E – угол наклона (также интерпретируемые как модуль упругости). Математически этот преобразователь формализуется следующим образом:

$$\sigma(t) = \Gamma_P[\varepsilon(t)] = \min\{h, \max\{-h, E[\varepsilon(t) - \varepsilon(t - \tau)] + \sigma(t - \tau)\}\}, \quad (1)$$

где $t - \tau$ характеризует значение в предыдущий момент времени. Полагая, что есть некоторый набор (в общем случае бесконечный) преобразователей Прандтля, каждый из которых имеет собственное пороговое значение, перейдем к более сложному преобразователю – Прандтля–Ишлинского. Он является компиляцией модели материала Ишлинского и преобразователя Прандтля и имеет вид:

$$\sigma(t) = \Gamma_{PI}[\varepsilon(t)] = \int_0^{\infty} \Gamma_P[h, \varepsilon(t)] d\Xi(h), \quad (2)$$

Одна из его особенностей – возможность учета дислокаций в материале посредством функции распределения пороговых значений h . Преобразователи с такой структурой обычно относят к классу сложных гистерезисных систем, описываемых посредством непрерывных аналогов преобразователей, состоящих из конечного числа элементарных носителей гистерезисных свойств – упоров. Блок-схема соответствующего преобразователя показана на рис. 2.

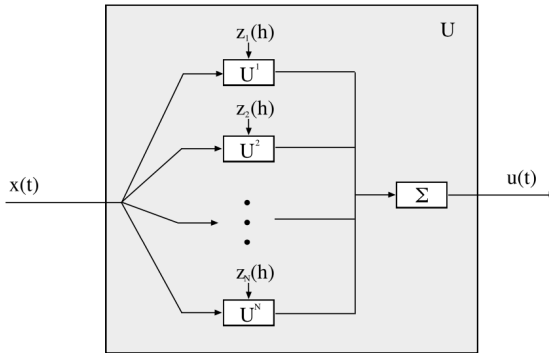


Рис. 2. Блок-схема параллельного соединения преобразователей Прандтля в модели Прандтля–Ишлинского

2. Модель колебаний классической и гистерезисной балок

Математические модели колебаний балок восходят к теории Эйлера–Бернулли [1] и Тимошенко [1]. Основное различие между указанными подходами заключается в поведении поперечного сечения (а именно, в положении поперечного сечения относительно нейтральной оси) балки при ее деформациях. Под классической моделью балки в настоящей работе будет пониматься модель, полученная с использованием теории Эйлера–Бернулли в предположении, что балка жестко закреплена в двух концах с нулевыми начальными условиями (балка покоится и не имеет начальной скорости):

$$\begin{cases} u_{tt} + a^2 u_{xxxx} = g(x,t), x \in [0, L], t > 0 \\ u(x, 0) = u_t(x, 0) = 0 \\ u(0, t) = u(L, t) = u_{xx}(0, t) = u_{xx}(L, t) = 0 \end{cases}, \quad (3)$$

где $u = u(x, t)$ – отклонение балки от положения равновесия, индексы обозначают частные производные, $g(x, t)$ есть функция, характеризующая нагрузку, a – величина, пропорциональная плотности $\rho(x) = const$ и обратно пропорциональная площади поперечного сечения, $t > 0$ – время моделирования, L – длина балки.

Предположим, что нормальное напряжение и деформация связаны следующим соотношением [5]:

$$\sigma(x, t) = \Gamma_{PI}[\varepsilon(x, t)], \quad (4)$$

Тогда, используя соотношение (4) с учетом (1) и (2) во время вывода уравнений колебаний балки, получим модифицированное уравнение (3) – модель колебаний балки с учетом распределенных гистерезисных свойств:

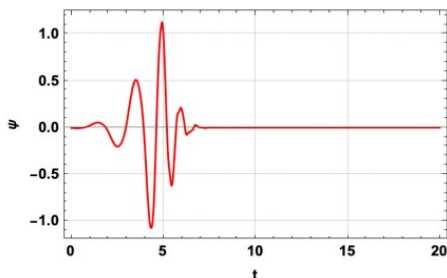
$$\begin{cases} u_{tt} + a^2 \Gamma[u_{xx}]_{xx} = g(x,t), x \in [0, L], t > 0 \\ u(x, 0) = u_t(x, 0) = 0 \\ u(0, t) = u(L, t) = u_{xx}(0, t) = u_{xx}(L, t) = 0 \end{cases}. \quad (5)$$

3. Моделирование колебаний классической и гистерезисной балок под нагрузкой сейсмического типа

В настоящем разделе приводятся результаты моделирования колебаний классической и гистерезисной балок. Моделирование производилось с использованием численных методов, а именно, посредством явной разностной схемы, со следующими параметрами: $h_t = 0.00125$ – шаг по времени, $h_x = 0.05$ – шаг по пространственной координате, $L = 1$, $T = 20$, $a^2 = 0.5$; пороговые значения преобразователя Прандтля–Ишлинского распределялись в интервале $(0, 100)$ с шагом 0.5. В качестве внешнего воздействия использовалась нагрузка сейсмического типа, представленная в виде функции материнского вейвлета (см. рис. 3):

$$\Psi_{a,b}(t) = \Psi\left(\frac{t-b}{a}\right), \quad (6)$$

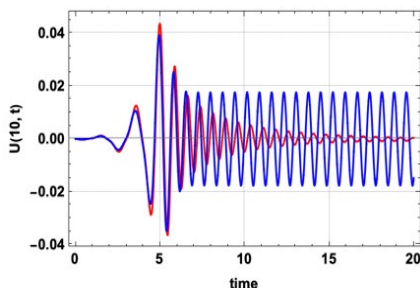
где a и b являются параметрами растяжения и сдвига материнской функции, соответственно. Под вейвлет-функциями, в рамках настоящей работы, понимается хорошо известное семейство вейвлетов Добеши.



слева – вейвлет Добеши 6 порядка с параметрами: $a = 1$, $b = 4$.

Рис. 3. График материнской вейвлет-функции от времени

Из результатов моделирования, представленных на рис. 4, видно, что реакции классической и гистерезисной балок различны. А именно, на некоторых временных интервалах реакция гистерезисной балки демонстрирует меньшую амплитуду, нежели классической, а на некоторых – большую. Тем не менее, через примерно 1.5 единиц модельного времени балка с учетом гистерезисных свойств стабилизируется в положение равновесия, отличное, однако, от исходного, что, в свою очередь, согласуется с упругопластическими свойствами гистерезисного материала.



синяя кривая соответствует решению уравнения колебаний классической балки, красная – гистерезисной

Рис. 4. Динамика дискретной точки балки с координатами 0.1 от положения равновесия в зависимости от времени

Заключение

Настоящая статья посвящена исследованию колебаний классической балки и балки с учетом распределенных гистерезисных свойств, формализуемых преобразователем Прандтля–Ишлинского, под воздействием нагрузки сейсмического типа. Приведены результаты численного моделирования колебаний, полученные с помощью численного метода – метода явной разностной схемы. Установлено, что балка с учетом гистерезисных свойств демонстрирует совершенно иную динамику, в сравнении с классической моделью. Полученный результат может быть полезен ученым и исследователям, имеющим дело с моделированием и проектированием несущих конструкций.

Настоящая работа выполнена при поддержке гранта РФФИ №19-08-00158-а.

Список литературы

1. Timoshenko, S. History of strength of materials / S. Timoshenko ; McGraw-Hill. – New York, 1953. – 452 p.
2. Карпов, Е. А. Моделирование поведения балки под нагрузкой / Е. А. Карпов, М. Е. Семенов // Сборник студенческих научных работ факультета компьютерных наук ВГУ. – 2020. – С. 128–133.
3. Stabilization of Chaos Via Strong Nonlinearities: The Lorenz-Malkus Wheel Under Coulomb and Hysteresis Frictions / M. E. Semenov [et al.] // Vibration Engineering and Technology of Machinery. – 2021. – P. 3–36.
4. Красносельский, М. А. Системы с гистерезисом / М. А. Красносельский, А. В. Покровский. – М. : Наука, 1983. – 271 с.
5. Krejci, P. Reliable solutions to the problem of periodic oscillations of an elastoplastic beam / P. Krejci // International Journal of Non-Linear Mechanics. – 2002. – V. 37. – P. 1337–1349.

Разработка информационной системы общения с детектированием эмоционального состояния человека

М. А. Карпова

Студент магистр

Я. А. Туровский

Доцент

Введение

С развитием информационных технологий люди больше времени проводят в социальных сетях, активно общаются в мессенджерах. По данным отчета Digital 2020 организации We Are Social [1], опубликовавшей данные в феврале 2020 года, свыше 4,5 млрд человек на Земле пользуются интернетом, то есть почти 60% населения планеты имеют возможность выходить в онлайн. В своем исследовании организация ссылается на статистику таких аналитических компаний, как GlobalWebIndex, Statistica, GSMA Intelligence, App Annie, SimilarWeb, LocoWise.

Среди типов мобильных приложений, которые интересуют пользователей больше всего, лидируют мессенджеры — ими пользуются 89% пользователей. Столько же процентов приходится на социальные сети. То, что люди так много времени общаются с помощью информационных технологий, приводит к тому, что уровень эмоционального интеллекта снижается.

Эмоциональный интеллект - сумма навыков и способностей человека распознавать эмоции, понимать намерения, мотивацию и желания других людей и свои собственные, а также способность управлять своими эмоциями и эмоциями других людей в целях решения практических задач.

Деграция эмоционального интеллекта означает, что человек хуже выражает свои эмоции и хуже воспринимает эмоции окружающих, в связи с этим он теряет способности самоконтроля, мотивации, интерпретации эмоций во взаимоотношениях и т.д. Стикеры и эмодзи, которые используются в переписках, являются плохой заменой для выражения эмоций людей, поскольку они не содержат в себе объективную информацию об эмоциональном состоянии человека.

Для того, чтобы максимально приблизить критерии при общении с использованием информационных технологий к критериям при общении лично необходимо создать систему, которая смогла бы детектировать эмоции пользователей и передавать их от одного пользователя к другому. В качестве решения указанной проблемы предлагается создать десктопные приложения и сайт для передачи сообщений, включающих в себя объективную информацию об эмоциональном (стресс-) состоянии человека. Для детектирования эмоционального состояния необходимо разработать новую систему цифровой обработки сигнала, позволяющую на коротких отрезках ВСР получать информацию о стресс-состоянии пользователя.

В ходе данной работы были реализованы серверная часть, сайт и десктопное приложение-чат, информация о стресс-состоянии пользователя передается в виде цвета букв. В дальнейшем планируется разработать алгоритм, позволяющий на коротких отрезках вариабельности сердечного ритма детектировать эмоциональное состояние пользователя, и реализовать приложение с использованием разработанного алгоритма для получения информации о вариабельности сердечного ритма с устройства-пульсометра, ее обработки и получения результирующего эмоционального состояния, которое затем передается в приложение-чат и сайт.

Разработанные сайт и приложение передают объективную информацию об эмоциональном состоянии пользователя в виде цвета букв сообщений. Поскольку приложения для детектирования стресс-состояния пользователя пока не реализовано, система осуществляет работу с определенным набором цветов:

- DarkOrange;
- Magenta;
- DodgerBlue;
- MediumSeaGreen;
- RosyBrown;
- Red.

1. Разработка серверной части

Разработанная серверная часть обеспечивает работу следующего функционала:

- Регистрация;
- Логинизация;
- Создание беседы;
- Получение списка пользователей, состоящих в беседе;
- Поиск бесед по названию;

- Добавление беседы в список бесед пользователя и выход из беседы;
- Получение и отправление сообщений.

Серверная часть была реализована с помощью фреймворка Flask [2] в среде разработки PyCharm.

При разработке серверной части были использованы следующие библиотеки:

- Для отправления сообщений всем подключенным пользователям была использована библиотека Flask-SocketIO [3].

- Для управления пользовательскими сессиями была использована библиотека Flask-Login.

- Для обработки совместного использования ресурсов между источниками (CORS) была использована библиотека Flask-CORS.

- Для хэширования паролей была использована утилита werkzeug.security.

- Для работы с базой данных была использована библиотека Flask-SQLAlchemy [4].

Серверная часть взаимодействует с базой данных SQLite, работа с которой происходит с помощью SQLAlchemy. Это обеспечивает реализацию технологии программирования ORM (Object-Relational Mapping), которая связывает базы данных с концепциями объектно-ориентированного программирования.

Обработка запросов происходит с помощью функционала, реализованного в файле app.py. Для работы базы данных были реализованы классы User, Room, Message в файле models.py. На рис.1 показана ER-диаграмма базы данных.

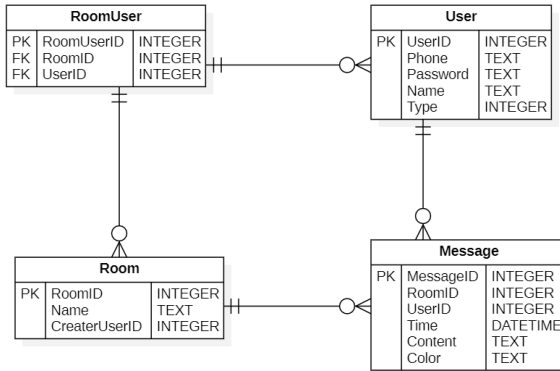


Рис. 1. ER-диаграмма базы данных

2. Разработка сайта

Разработка сайта происходила с помощью JavaScript фреймворка Vue.js [5]. При разработке клиентской части были использованы следующие библиотеки:

- Набор CSS-стилей и JavaScript-скриптов Bootstrap для оформления сайта.

- Библиотека для передачи сообщений Vue-Socket.io [6].

Разработанная клиентская часть состоит из следующих компонентов:

- Registration.vue – для регистрации пользователя
- SignIn.vue – для логинизации пользователя
- Chat.vue – для поиска, добавления, удаления, создания бесед и отправления сообщений с информацией об эмоциональном состоянии в виде цвета

На рис. 2 представлен компонент для регистрации.

Регистрация

Рис. 2. Компонент Registration.vue

На рис. 3 представлен компонент для логинизации.

Вход

Запомнить меня

[Нет аккаунта? Зарегистрироваться](#)

Рис. 3. Компонент SignIn.vue

На рис. 4 представлен компонент для взаимодействия между пользователями.

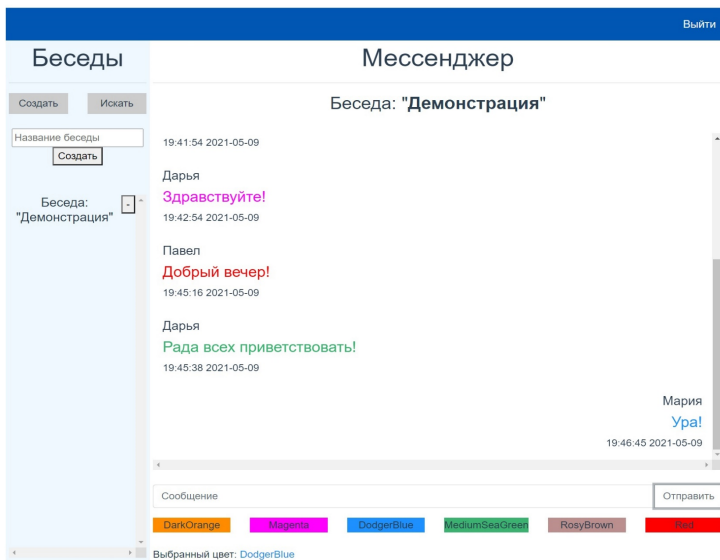


Рис. 4. Компонент Chat.vue

Информация об эмоциональном состоянии пользователя передается с помощью цвета букв сообщений. Выбор цвета определяется нажатием на кнопку соответствующего цвета.

3. Разработка десктопного приложения-чата

Десктопное приложение-чат является Windows Form Application, реализовывалось на языке C# [7] в среде разработки Visual Studio 2017. При разработке серверной части были использованы следующие библиотеки:

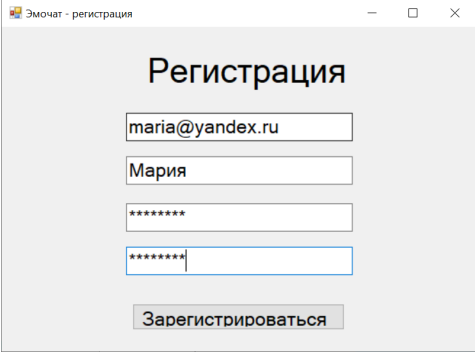
- Quobject.SocketIoClientDotNet.Client – для общения с сервером
 - Newtonsoft.Json – для сериализации в JSON и десериализации
- Приложение включает в себя следующие формы:

- EmochatRegistrationForm - для регистрации пользователя
- EmochatLoginForm - для логинизации пользователя
- EmochatForm - для поиска, добавления, удаления, создания бесед и отправления сообщений с информацией об эмоциональном состоянии в виде цвета

Для работы приложения были реализованы следующие классы:

- Классы User, Message, Room – для сериализации в JSON и десериализации
- Класс ValidateCredentials – для проверки данных при регистрации и логинизации

На рис. 5 показана форма для регистрации.

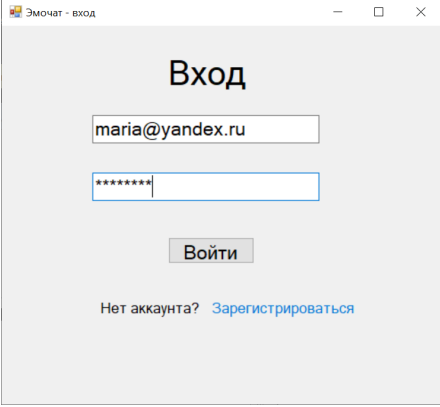


Эмочат - регистрация

Регистрация

Рис. 5. Форма для регистрации

На рис. 6 показана форма для входа в приложение.



Эмочат - вход

Вход

Нет аккаунта? [Зарегистрироваться](#)

Рис. 6. Форма для логинизации

На рис. 7 представлена форма для взаимодействия между пользователями.

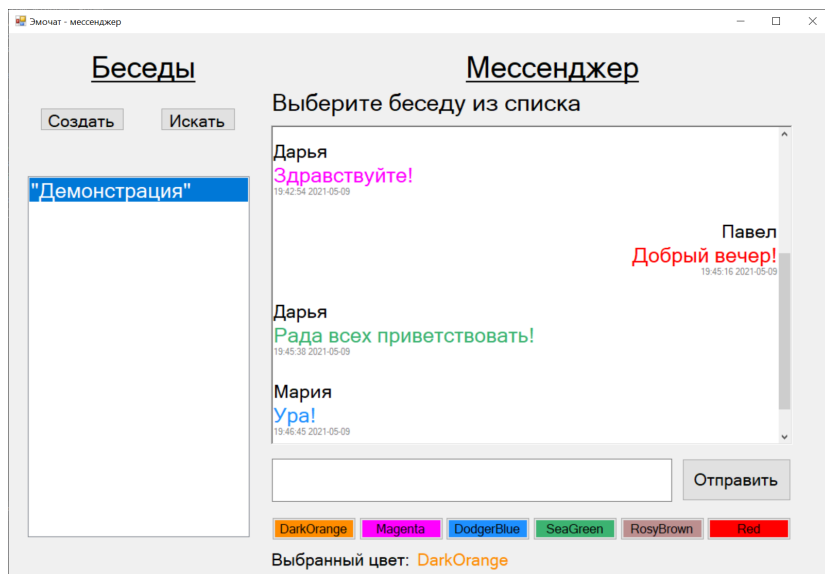


Рис. 7. Форма чата

Выводы

В ходе работы была разработана часть системы общения с детектированием эмоционального состояния пользователя.

Были реализованы:

- серверная часть
- сайт
- десктопное приложение-чат

В дальнейшем планируется:

- Разработка алгоритма анализа вариабельности сердечного ритма для детектирования эмоционального состояния пользователя
- Разработка десктопного приложения для детектирования эмоционального состояния пользователя

Разрабатываемая система общения может быть востребована в психологических исследованиях, индустрии компьютерных игр, социальных сетях.

Список литературы

1. Отчет Digital 2020 организации We Are Social [Электронный ресурс]: база данных. – Режим доступа : <https://wearesocial.com/digital-2020>

2. Документация по фреймворку Flask [Электронный ресурс]: база данных – Режим доступа : <https://flask.palletsprojects.com/en/1.1.x/>
3. Документация по программной библиотеке Flask-SocketIO [Электронный ресурс]: база данных – Режим доступа : <https://flask-socketio.readthedocs.io/en/latest/>
4. Документация по программной библиотеке Flask-SQLAlchemy [Электронный ресурс]: база данных – Режим доступа : <https://flask-sqlalchemy.palletsprojects.com/en/2.x/>
5. Документация по фреймворку Vue.js [Электронный ресурс]: база данных – Режим доступа : <https://vuejs.org/>
6. Документация по программной библиотеке Vue-SocketIO [Электронный ресурс]: база данных – Режим доступа : <https://www.npmjs.com/package/vue-socket.io>
7. Документация по языку программированию C# [Электронный ресурс]: база данных – Режим доступа : <https://docs.microsoft.com/ru-ru/dotnet/csharp/>

Разработка системы учёта заболеваемости «COVID-19» по Воронежской области

И. В. Ключкова

Студент магистрант

И. В. Абрамов

Доцент

Введение

В сложившейся в результате быстрого распространения нового типа коронавируса SARS-CoV-2 (2019-nCoV) ситуации необходимо было действовать быстро. Принимались различные меры по выявлению и борьбе с вирусом SARS-CoV-2, в частности, в кратчайшие сроки было организовано тестирование людей с помощью определения РНК нового коронавируса методом полимеразной цепной реакции (ПЦР). Затем стало проводиться и тестирование крови на антитела классов Ig (G, A, M) различными методами.

Встал вопрос и о программном обеспечении этого процесса. Данные о пациентах и результатах исследований необходимо особым образом хранить, систематизировать, анализировать, наглядного представлять с различных точек зрения. Необходимо также предоставить регистраторам, лаборантам, контролирующим органам и другим работникам удобные механизмы, ПО для электронного учёта данных, который в данной ситуации просто необходим. Решение данной проблемы на территории Воронежской области и будет освещено в данной статье.

1. Выбор способа реализации

Для учёта заболеваемости COVID-19 на территории области ДЗ ВО было принято решение не создавать новый программный продукт, а доработать уже существующий. Доработка уже существующей и успешно функционирующей системы регионального уровня несомненно требовала меньше времени и ресурсов, чем создание новой системы с нуля. Тем более, что подходящий продукт в области был.

Выбрана для доработки была медицинская информационная система (МИС) АУЗ ВО «ВОККДЦ». Данный программный продукт уже успешно используется, как для обеспечения работы самого

«ВОККДЦ», так и в других мед. организациях (МО), производящих запись пациентов на услуги в рамках обязательного или добровольного медицинского страхования в «ВОККДЦ».

2. Проектирование решения

Наиболее глобальным процессом в проектируемой системе является процесс, начинающийся регистрацией направления на исследование на SARS-CoV-2, а заканчивающийся получением его результата и соответствующими дальнейшими действиями.

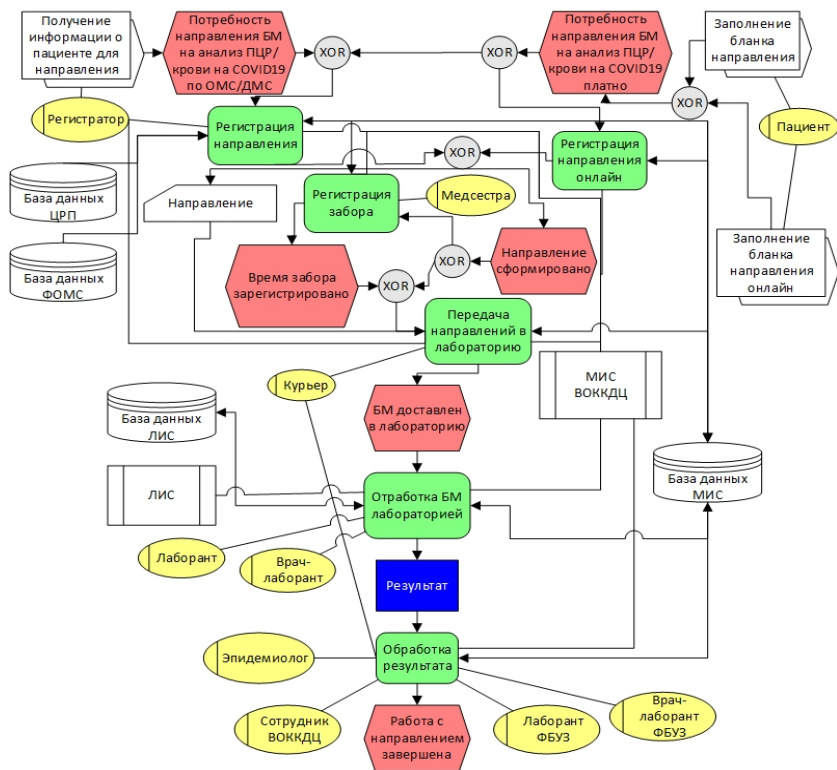


Рис. 1. Диаграмма процесса жизненного цикла направления на исследование на SARS-CoV-2

На рис. 1 представлена диаграмма процесса жизненного цикла направления. Отправной точкой данного процесса является событие возникновения потребности регистрации направления на исследование либо в рамках платных услуг, либо в рамках ОМС или ДМС. Указанным

событиям предшествуют процессы получения необходимой для формирования направления информации о пациенте. В случае ОМС и ДМС данные получает регистратор направляющей МО или регистратуры ДМС соответственно, в ином случае пациент заполняет бланк на сайте «ВОККДЦ» или же вручную. Далее следует регистрация направления посредством регистратора или же через сайт «ВОККДЦ». После формирования направления при необходимости возможна регистрация фактического времени забора биоматериала. Когда направление готово, происходит его передача в лабораторию вместе с биоматериалом пациента. Затем непосредственно происходит работа лаборатории, после чего следуют все необходимые после получения результата манипуляции. В ходе данного процесса повсеместно используется МИС «ВОККДЦ» и её база данных. Также при регистрации направлений в рамках ОМС применяются базы ЦРП и ФОМС, а при отработке направления лабораторией «ВОККДЦ» применяется (лабораторная информационная система) ЛИС «ВОККДЦ» и её база.

Рассмотрим декомпозиции функций рассматриваемого процесса. На рис. 2 представлена детализация функции «Регистрация направления». Процесс начинается с авторизации пользователя в программе, в соответствии с чем открывается либо интерфейс регистратора внешней МО, либо один из интерфейсов регистратуры «ВОККДЦ»: ОМС, платных услуг, ДМС или кол-центра. Далее регистратор ищет нужного пациента с помощью своего интерфейса. Есть возможность и внесения нового пациента в систему. После того, как пациент выбран, вносятся непосредственно данные направления и происходит его запись. Если какие-то необходимые данные не внесены или некорректны, программа указывает на них и не записывает направление до устранения погрешностей. В зависимости от данных направления программа автоматически определяет лабораторию выполнения соответствующего исследования, и, когда направление сформировано, оно автоматически распечатывается, если лабораторией является ФБУЗ. В ином случае пользователь имеет возможность отказаться от печати направления.

Записаться на исследование на SARS-CoV-2 в рамках платных услуг можно также и через сайт «ВОККДЦ». Рис. 3 иллюстрирует этот процесс, представляя собой декомпозицию функции процесса жизненного цикла направления «Регистрация направления онлайн». После заполнения формы анкеты на сайте пациентом система создаёт документ записи на соответствующие услуги. Когда документ успешно создан, данные из формы анкеты сохраняются в буферной таблице базы

данных. Теперь система ожидает оплаты заказа, после поступления которой из буферной анкеты формируется полноценное направление, а временная анкета удаляется из базы.



Рис. 2. Диаграмма процесса регистрации направления

На рис. 4 представлена декомпозиция функции «Передача направлений в лабораторию». На входе данный процесс имеет направление в базе МИС. Исполнителем исследования в направлении может значиться лаборатория «ВОККДЦ» или лаборатория внешней МО.

Регистратор, направляющий биоматериал, может воспользоваться механизмом формирования реестра направлений. Реестр представляет собой приложение в виде таблицы списка передаваемых материалов, содержащей основные данные направлений, в том числе их штрих-коды в удобном для сканирования виде. Биоматериал доставляется в лабораторию «ВОККДЦ» или какую-либо другую, на чём процесс передачи направления завершается.



Рис. 5. Диаграмма процесса отработки биоматериала лабораторией

Функция «Отработка БМ лабораторией» представлена рис. 5. Данный процесс начинается с события доставки биоматериала либо в

лабораторию «ВОККДЦ», либо в лабораторию внешней МО. В первом случае результат исследования вносится в ЛИС, а затем автоматически передаётся в МИС. Во втором случае лаборант сначала регистрирует получение биоматериала в МИС, а затем в неё же вносится результат исследования. Причём, если регистрация или внесение результата произвела лаборатория, не указанная в направлении как исполнитель данного исследования, таковая автоматически заменяется на текущую лабораторию с сохранением истории изменений.

Последней функцией процесса жизненного цикла направления является функция «Обработка результата», декомпозиция которой изображена на рис. 6.

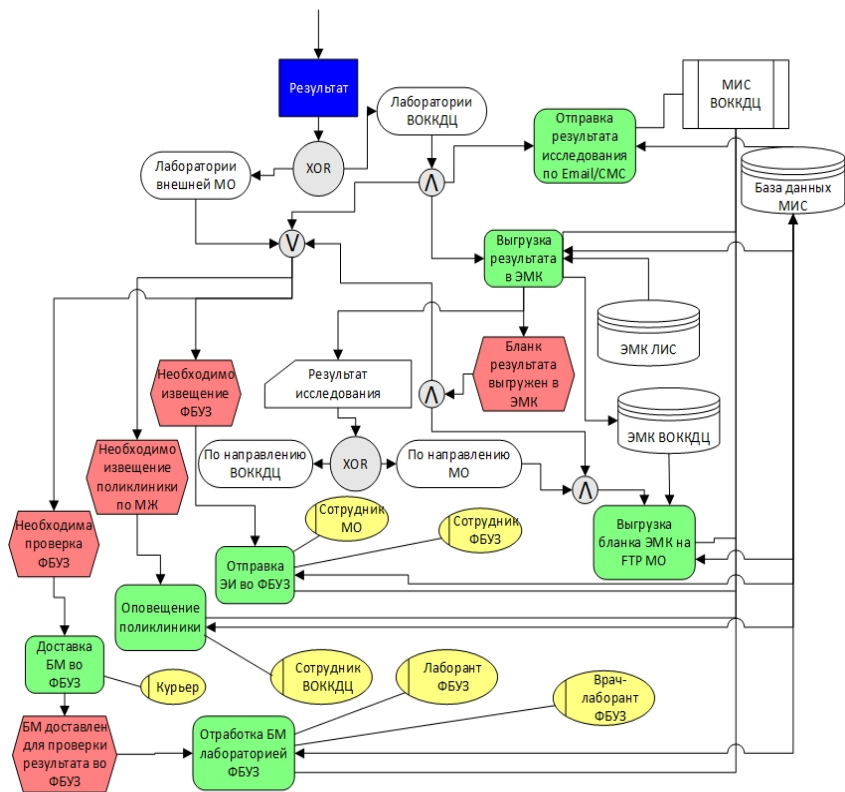


Рис. 6. Диаграмма процесса обработки результата

По принципу и подобию регистрации направлений на лабораторные исследования должна быть реализована регистрация КТ с типичной картиной для COVID-19.

Заключение

Данная статья посвящена разработке системы учёта заболеваемости «COVID-19» по Воронежской области. Спроектированная система будет хранить данные о пациентах и результатах исследований. В ней будут реализованы механизмы, которые на основе полученных данных будут систематизировать, анализировать и наглядно представлять их с различных точек зрения. Система предоставит регистраторам, лаборантам, контролирующим органам и другим работникам удобные механизмы, ПО для электронного учёта данных.

Список литературы

1. Воробьев А. А. Медицинская микробиология, вирусология и иммунология / А. А. Воробьев [и др.]. – М.: Медицинское информационное агентство, 2012. – 704 с.
2. Быкова, Т. А. Документационное обеспечение управления (делопроизводство) / Т.А. Быкова, Т.В. Кузнецова, Л.В. Санкина – М.: ИНФРА-М, 2020. – 304 с.
3. Филиппов Е. В. Настольная книга 1С: Эксперта по технологическим вопросам / Е. В. Филиппов. – М.: 1С-Паблишинг, 2014. – 247 с.

Решение ресурсоемких задач с помощью технологий программирования графических процессоров

А. С. Коновской

Студент магистр

С. В. Борзунов

Доцент

Введение

В настоящее время технологии высокопроизводительных вычислений используются во многих областях науки и техники. С распространением графических процессоров значительно расширяется спектр задач, которые могут быть решены с использованием различных методов распараллеливания. В силу этого исследование сравнительных особенностей различных технологий программирования графических процессоров представляется актуальной задачей.

Целью работы является сравнение технологий программирования графических процессоров CUDA и OpenACC на основе их применения к решению известных ресурсоемких задач.

1. CUDA

CUDA – программно-аппаратная архитектура, которая может значительно увеличить производительность вычислений за счет использования графических процессоров Nvidia. Представляет собой C-подобный язык программирования с компилятором и вычислительными библиотеками на GPU [1].

CUDA использует большое количество отдельных потоков для расчетов. Все они группируются в иерархию – grid / block / thread. Grid является начальным уровнем в модели – отвечает ядру и объединяет все потоки, которые выполняет данное ядро. Grid – одномерный или двумерный массив блоков (block). Каждый блок представляет собой полностью независимый набор скоординированных между собой потоков. Каждый поток обладает своей памятью (local), недоступной другим потокам. Внутри блока потоки имеют разделяемую память (shared) и синхронное исполнение. Потоки из разных блоков не взаимодействуют, но всем потокам доступна общая глобальная память.

2. OpenACC

OpenACC – это программный стандарт для параллельного программирования, разрабатываемый совместно компаниями Cray, CAPS, Nvidia и PGI.

Основным режимом использования OpenACC являются директивы, точно так же как и в OpenMP 3.x или более раннем OpenHMP. Библиотека поддержки предоставляет несколько вспомогательных функций, описанных в заголовочных файлах “openacc.h” для C/C++ и “openacc_lib.h” для Fortran.

Есть два основных ядра распараллеливания: 1) kernels, распараллеливание при помощи ядра, которое в автоматическом режиме подберет, что можно безопасно распараллелить, а что оставить в основном процессе; 2) parallel, распараллеливает выбранную область.

3. Задача перемножения матриц

Задача заключается в нахождении матрицы $C[M * K]$, полученной путем перемножения двух заданных прямоугольных матриц: $A[M * N]$ и $B[N * K]$ – с вещественными коэффициентами.

4. Реализация на CUDA

Вариант реализации основан на использовании глобальной памяти (реализация приведена в листинге 1).

Листинг 1

Перемножение матриц на CUDA

```
__global__ void kernel_global ( float *a, float *b, int n,
float *c )
{int bx = blockIdx.x; // номер блока по x
int by = blockIdx.y; // номер блока по y
int tx = threadIdx.x; // номер нити в блоке по x
int ty = threadIdx.y; // номер нити в блоке по y
float sum = 0.0f;
int ia = n * ( BLOCK_SIZE * by + ty ); // номер строки из A'
int ib = BLOCK_SIZE * bx + tx; // номер столбца из B'
int ic = ia + ib; // номер элемента из C'
// вычисление элемента матрицы C
for ( int k = 0; k < n; k++ ) sum += a[ia + k] * b[ib + k *
n];
c[ic] = sum;
}
```

Схема работы алгоритма заключается в следующем:

1. Матрицы A и B разбиваются на блоки заданного размера.

2. В каждом блоке функционируют потоки, количество которых зависит от размера блока. Координаты каждого потока определяются координатами внутри блока.

3. Каждый поток читает одну строку из матрицы A , один столбец из матрицы B и вычисляет соответствующий элемент матрицы C .

5. Реализация на OpenACC

Вариант реализации основан на сочетании директив и выделении памяти для каждого массива (программная реализация приведена в листинге 2). К основной реализации функции подключается прагма ACC `parallel` которая выполняет распараллеливание цикла. Остальные прагмы показывают, как вести себя с данными.

Листинг 2

Перемножение матриц на OpenACC

```
void Matrix_Mul ( float *restrict c, float *a, float *bT, int
N )
{
#pragma acc parallel loop present (c[0:N], a[0:N], bT[0:N])
for ( int n = 0; n < N; n++ )
{
for ( int m = 0; m < N; m++ )
{
float sum = 0.f;
for ( int k = 0; k < N; k++) sum += a[k + n * N] * bT[k
+ m * N];
c[m + n * N] = sum;
}
}
}
```

6. Задача о кратчайшем пути. Алгоритм Дейкстры

Задача заключается в нахождении кратчайшего пути из каждой вершины u в каждую вершину v .

Начальным весом всех вершин является большое положительное число (максимальное, больше того, что может быть в данном графе), в алгоритмах его принято обозначать бесконечностью. Одна из вершин обозначается 0 и является началом обхода графа. Массив флагов заполняется нулями, и запускается основной цикл.

На каждом шаге цикла мы ищем вершину v с минимальным расстоянием и флагом, равным нулю. Затем мы устанавливаем в ней флаг в 1 и проверяем все соседние с ней вершины u . Если в них (v и u) расстояние больше, чем сумма расстояния до текущей вершины и длины ребра, то уменьшаем его. Цикл завершается, когда флаги всех вершин становятся равны 1 [2].

7. Реализация на CUDA

Основной особенностью организации алгоритма Дейкстры заключается в том, что вычисление новых весов заключается в два этапа:

1. Происходит вычисление всех соседних весов, если текущая вершина не участвует в подсчетах.
2. После подсчета данная вершина освобождается и соседние пересчитываются.

Реализацию (листинг 3) рассмотрим с использованием глобальной памяти устройства.

Листинг 3

Алгоритм Дейкстры на CUDA

```
__global__ void closestNodeCUDA(data_t* node_dist, int*
visited_node, int* global_closest, int num_vertices) {
    data_t dist = INF_DIST + 1;
    int node = -1;
    int i;

    for (i = 0; i < num_vertices; i++) {
        if ((node_dist[i] < dist) && (visited_node[i] != 1)) {
            dist = node_dist[i];
            node = i;
        }
    }

    global_closest[0] = node;
    visited_node[node] = 1;
}
```

8. Реализация на OpenACC

Данная реализация (листинг 4) отличается от реализации на CUDA лишь подключением отдельных прагм, которые меняют способ распараллеливания. В случае OpenACC мы явно даем понять компилятору, как нужно распараллеливать данные циклы. При этом `graph` и `node_dist` у нас являются общими для всех, а `new_dist` и `next` являются приватными, каждый для своего потока.

Листинг 4

Алгоритм Дейкстры на OpenACC

```
int i, next;
for (i = 0; i < num_vertices; i++) {
    int curr_node = closestNodeOMP(node_dist,
visited_node, num_vertices); //closest node
    visited_node[curr_node] = 1;
    int new_dist;
    #pragma acc parallel
```

```

    {
        #pragma acc for private(new_dist,next)
        for (next = 0; next < num_vertices; next++) {
            new_dist = node_dist[curr_node] +
graph[curr_node*num_vertices + next];
            if ((visited_node[next] != 1)
&& (graph[curr_node*num_vertices + next] != (data_t)0)
&& (new_dist < node_dist[next])) {
                node_dist[next] = new_dist;
                parent_node[next] = curr_node;
            }
        }
    }
}

```

9. Сравнение результатов работы

В результате перемножения матриц при больших значениях размеров матриц мы можем увидеть, что для языка OpenACC затраченное время работы меньше, чем для CUDA. При маленьких значениях выигрывает во времени CUDA (см. рис 1).

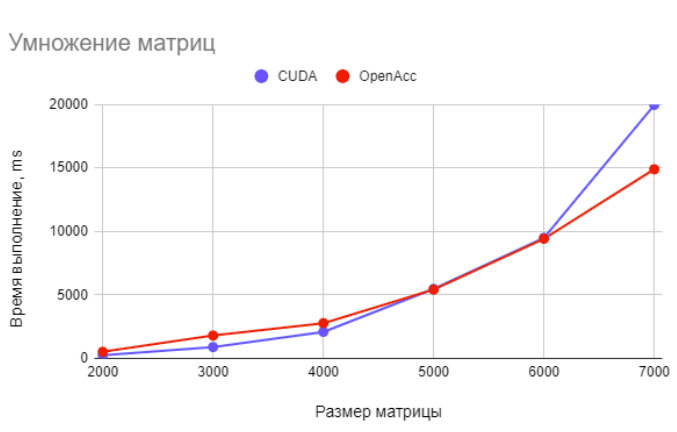


Рис. 1. Время работы программы перемножения матриц

При выполнении алгоритма Дейкстры результаты вычислений показали очень большой разброс. При больших значениях реализация на OpenACC показала лучший результат (см. рис 2).

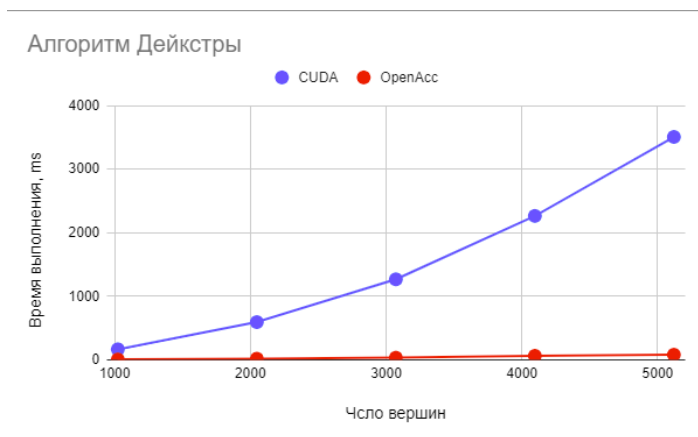


Рис. 2. Время работы программы, выполняющей алгоритм Дейкстры

Заключение

В работе проведено сравнение популярных технологий программирования графических процессоров CUDA и OpenACC. Был реализован программный код для решения рассматриваемых задач с помощью указанных технологий. Реализации алгоритмов решения ресурсоемких задач, использующие указанные технологии, показали сравнимое время матричных вычислений, но для алгоритма Дейкстры технология OpenACC демонстрирует несколько меньшее время выполнения.

Список литературы

1. NVIDIA CUDA Programming Guide [Электронный ресурс]. – Режим доступа: <https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>
2. Алгоритмы: Построение и анализ / Т. Х. Кормен [и др.]. – 3-е изд. – Москва : Вильямс, 2013.

Разработка игры-квеста для психологического анализа

А. Е. Копылова

Студент магистр

Е. А. Киселев

Доцент

Введение

На данный момент информационные технологии прочно вошли в жизнь современного человека и затронули практически все сферы жизни. Одной из наиболее популярных сфер применения информационных технологий является индустрия разработки игрового программного обеспечения. Однако игры могут создаваться не только с целью развлечения. Наблюдая и анализируя поведение игрока, можно сделать выводы о его психологическом портрете, так как именно в такого рода деятельности, легче всего отследить психоэмоциональные реакции.

Целью данной работы является разработка игры-квеста, с помощью которой, можно бы было отследить реакции человека и, анализируя его игровое поведение, определить тип личности, а также уровень развития у человека таких познавательных процессов как внимание и логическое мышление.

Данная работа посвящена разработке основных игровых механик и элементов, обеспечивающих возможность игрока передвигаться по игровому пространству, взаимодействовать с объектами мира, решать головоломки и давать ответы на задачи и вопросы.

1. Создание игрового пространства

Игровое пространство состоит из 3D-моделей. Процесс создания и подготовки для импорта в игру каждой из моделей включает в себя следующие этапы [1]:

1. Создание высоко-полигональной модели (high-poly). Высоко-полигональная модель – это 3D модель с, как правило, большим количеством полигонов, которая демонстрирует конечный вид объекта игрового пространства. Данная модель используется для запекания (подробнее в пункте 4 этого списка).

2. Создание низко-полигональной модели. Низко-полигональная модель отражает лишь основные черты и формы высоко-полигональной модели. Данный вид 3D-моделей используется в игровом пространстве вместо высоко-полигональных для экономии ресурсов на визуализацию. Экономия происходит за счёт уменьшения количества полигонов, которые нужно просчитывать видеокарте или процессору.

3. Развертка низко-полигональной модели. Данное действие необходимо для того чтобы корректно наложить текстуры.

4. Запекание низко-полигональной модели. Данный процесс позволяет сделать так, чтобы низко-полигональная модель выглядела высоко-полигональной. Это достигается за счёт переноса информации о нормалях полигонов высоко-полигональной модели на полигоны низко-полигональной модели.

5. Наложение текстур. Данный пункт включает в себя назначение материалов и их настройку.

6. Импорт модели в игровой движок. В данный пункт входят следующие этапы: настройка коллизии, наложение текстур и расположение на карте мира.

Пример игрового пространства первого уровня представлен на рис. 1.



Рис. 1. Пример игрового пространства

2. Персонаж

Данный игровой элемент содержит в себе скрипты, обеспечивающие игроку возможность передвигаться по игровому

пространству [2], также здесь происходит обработка событий взаимодействия игрока с миром, для чего используется информация о нажатии кнопок клавиатуры и движение мыши.

В рамках данного игрового элемента реализована система инвентаря. Его реализацию можно разделить на два условных блока: интерфейсная часть, в которой игрок может рассматривать предметы в увеличенном виде, и программная часть, обеспечивающая возможность добавлять предметы в инвентарь. На рис. 2 представлена интерфейсная часть инвентаря.

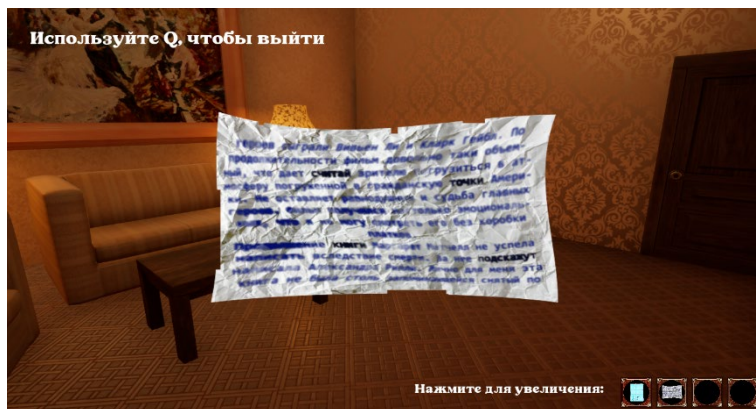


Рис. 2. Интерфейсная часть инвентаря

3. Панели ввода

Рассмотрим игровые механики, контролирующие возможность открыть дверь, исходя из правильности введенного кода. В рамках данной работы существует три вида: панель с фигурами, кодовый замок и панель с буквами.

Панель с фигурами представляет собой набор из пяти кнопок, с меняющимися при нажатии на них изображениями. После каждого нажатия проверяется, правильная ли последовательность изображений выставлена на кнопках. Если последовательность правильная, то происходит открытие двери.

Последние два вида имеют общую структуру: поле для отображения текста и набор кнопок, который для каждого вида панелей разный. При нажатии на кнопки в поле для отображения текста выводится нажатый символ. Нажатие на кнопку «Enter» вызывает проверку верность введенного кода. После чего на экран выводится

сообщение о правильности ответа. В случае верного ответа происходит открытие двери.

Реализацию данных игровых механик также можно разделить на два условный блока: интерфейсная часть, представленная на рис. 3, и часть, отвечающая за воплощение панелей в игровом пространстве и обеспечивающая возможность начала взаимодействия, вызывая интерфейсную часть [3].

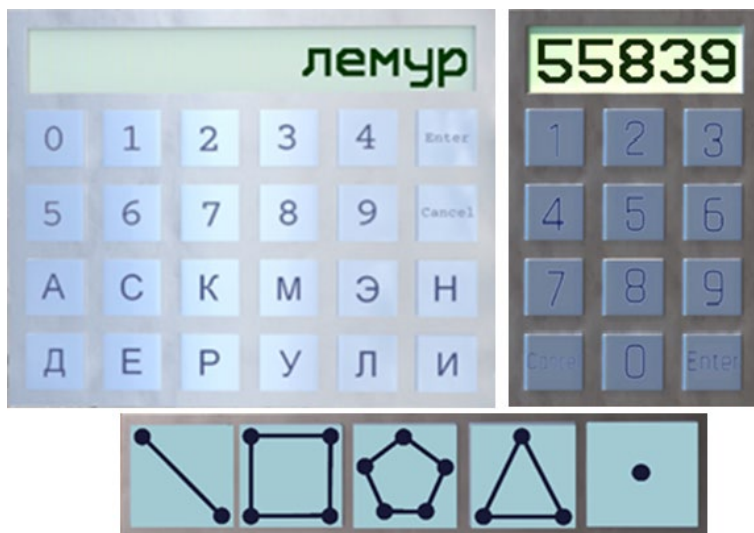


Рис. 3. Интерфейсная часть панелей для ввода

4. Лабиринты

Лабиринт представляет собой игровую механику, которая включает в себя необходимость переключать управление игрока с персонажа на другой объект. В рамках данной работы существует два вида лабиринтов:

- **Круглый лабиринт.** В данной игровой механике игроку необходимо управлять вращением лабиринта, чтобы довести статичный шарик до финиша

- **Прямоугольный.** В данной игровой механике игроку необходимо пройти лабиринт, управляя шариком.

Переключение управлением происходит после начала взаимодействия с объектом. Для обратного переключения требуется закончить взаимодействия, следуя подсказкам управления. Рис. 4

иллюстрирует внешний вид данной механики, а именно ее воплощение в игровом пространстве.

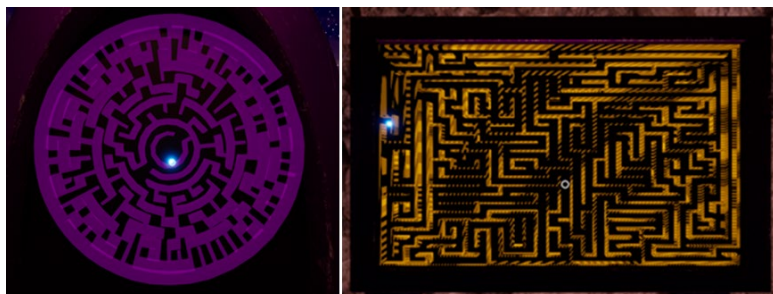


Рис. 4. Внешний вид лабиринтов

Заключение

Данная статья посвящена разработке и реализации игровых механик и элементов. На основе данных механик спроектирована и реализована квест-игра. В течение игрового процесса игроку предоставляются следующие возможности:

– перемещаться по 3D пространству мира (игровому пространству);

- добавлять предметы в инвентарь;
- просматривать содержимое инвентаря в увеличенном виде;
- взаимодействовать с объектами мира;
- решать загадки и головоломки.

В дальнейшем планируется внедрить в данную игру отслеживание метрик, и их на основе произвести анализ поведения игрока и протестировать развитость его познавательных процессов: внимания и логическое мышления.

Список литературы

1. Куксон А. Разработка игр на Unreal Engine 4 за 24 часа / А. Куксон, Р. Даулингсока, К. Клампер. – Москва: Эксмо, 2019. – 528 с.

2. Игровое моделирование или весь процесс создание 3D модели для игры [Электронный ресурс] : сайт. – Режим доступа : <https://dtf.ru/gamedev/92904-igrovoye-modelirovanie-ili-ves-process-sozdanie-3d-modeli-dlya-igry>

3. Принципы разработки игрового интерфейса [Электронный ресурс] : сайт. – Режим доступа : <https://dtf.ru/gamedev/70438-principy-razrabotki-igrovogo-interfeysa>

Разработка информационной системы учёта посещаемости факультета

В. К. Кушнеренко

Студент магистр

А. А. Вахтин

Доцент

Введение

На данный момент во многих учебных заведениях распространены различные документы по учету посещаемости. Они используют данные, которые каждую неделю отмечаются самими преподавателями или предоставляются старостами, что не способствует получению своевременной информации о посещаемости. Документы по ее учёту содержат информацию об отсутствии или присутствии человека на занятиях.

Проверка посещаемости требует немаленьких затрат: сопровождается значительным объёмом документов, отнимает много времени как у старост, так и у преподавателей.

Принимая во внимание вышеописанные факты, мы можем сформулировать ряд конкретных проблем:

- Необходимость затрат большого количества времени на заполнение документов;
- Неструктурированность документов по учету посещаемости;
- Старосты не всегда добросовестно заполняют посещаемость;
- Необходимо хранить все документы по учету в бумажном виде;
- Сложность в просмотре и ведении статистики по каждому студенту;

Отдельно рассмотрим учёт посещаемости через нашу систему.

Разработка системы для учета посещаемости позволит решить указанные проблемы процесса его проведения, который используется в настоящее время, сократить затраты времени на заполнение документов по учету, работать под разными ролями.

1. Постановка задачи

Целью данной работы является разработка системы учета посещаемости.

Конечными пользователями системы являются:

- Студенты, которые будут иметь возможность просматривать статистику по посещениям;
- Старосты, которые смогут заполнять посещаемость вместо преподавателей, если это будет необходимо;
- Преподаватели, которые смогут также следить за статистикой и заполнять данные;
- Администраторы, которые будут иметь возможность делать все действия в системе;

К разрабатываемому продукту предъявляются следующие требования:

- Удобный просмотр и редактирование учета посещаемости;
- Удобное взаимодействие с элементами системы;
- Работа под несколькими ролями в системе;
- Возможность работать с данными, связанными с посещаемостью;

Для создания качественного решения необходимо провести анализ существующих решений.

2. Анализ

Анализ предметной области

Процесс учета включает в себя множество действий, направленных на формирование документов о посещаемости студентов и сбора различной актуальной информации об этом.

При проверке посещаемости преподаватель получает список/журнал посещаемости. Затем он заходит в него и отмечает не пришедших на занятие студентов. После этого он может продолжать занятие. Имеется возможность для администратора изменять данные студентов, дисциплин, добавлять новые группы, занятия. Также можно провести поиск по конкретному студенту.

В данной работе будет проведён детальный разбор функционирования системы.

Анализ задачи

Рассмотрим задачу экспорта данных о посещаемости из файла. В настоящий момент имеются файлы с записями, общая структура которых представлена в таблице.

Упрощенное представление списка студентов для посещаемости.

ФИО студента/Дата проведения занятия	01. 05. 02	02.05.02	03.05.02
Беркутов И. С.	+	-	+
Меркулов С. В.	-	+	+
Петров П. В.	+	+	+
Сидоров В. И.	+	-	-

В 1 ячейке хранится параметр, обозначающий то что находится в ячейках внизу (ФИО студентов) и справа (даты занятия), в остальных ячейках находятся отметки - плюс или минус в зависимости от того был ли студент на занятии или нет.

Соответственно, экспорт в PDF данных студентов по посещаемости должен быть разработан с учетом структуры файла с формализованными требованиями.

Теперь проанализируем задачу разработки системы.

Для работы в системе предусмотрены разные роли. Роли могут принимать следующие значения:

- Студент;
- Преподаватель;
- Староста;
- Администратор.

Отметки посещаемости могут иметь свои статусы:

- Плюс - студент был на занятии;
- Минус - студент не был на занятии.

Система должна упрощать процесс своевременного получения данных, а также отслеживание статистики посещаемости, с учетом текущих данных.

Существующие решения

При проведении анализа были рассмотрены следующие готовые решения по учету посещаемости, в скобках приведены ссылки на официальные сайты существующих аналогов:

- Moodle (<https://edu.vsu.ru/>) Система позволяет не только отслеживать посещаемость, но и выставять оценки за курсы, общаться с преподавателями онлайн. Внешний вид сайта, представляющего систему представлен на рис. 1.

- iTeacherBook - журнал преподавателя (<https://ipadstory.ru/iteacherbook-zhurnal-prepodavatelya.html>) - рис. 2.

Этот сайт позволяет хранить информацию о студентах, их посещаемости, оценках и других различных данных.



Рис. 1. Официальный сайт системы Moodle

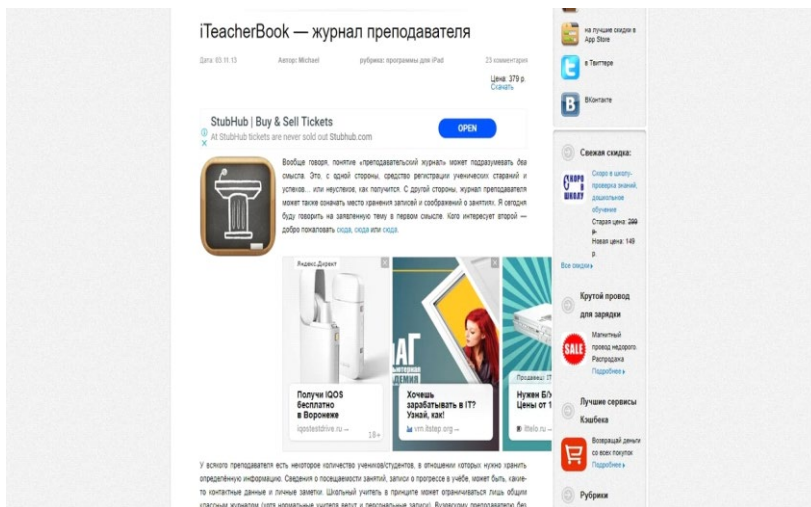


Рис. 2. Официальный сайт системы iTeacherBook

3. Реализация

Средства реализации

Для реализации проекта были выбраны следующие технологии и программные средства:

- Язык программирования Python 3.8 - обладает простотой и функциональностью, широко используется для разработки веб-приложений.

- Библиотека Flask [1] из языка программирования Python – обеспечивает быстрое и легкое создание веб-приложений и различных необходимых для его функционирования модулей.

- Библиотека Flask-Login - для реализации авторизации и аутентификации в системе по ролям.

- Шаблонизатор Jinja [2] – для обработки данных, поступающих на HTML-страницу.

- Библиотека SQLAlchemy [3] - для создания прослойки в виде классов, связанных с БД.

- Система контроля версий Git – обеспечивает возможность сохранения состояний проекта, получения текущей версии проекта и извлечение старых версий. Более того, система позволяет отслеживать этапы разработки решения.

Реализация классов

В результате анализа было принято решение о разработке системы, описание которой приводится в данной главе.

Структура проекта имеет вид:

- Файл `app.py` – содержит в себе все маршруты веб-приложения. Также здесь идет печать в PDF и авторизация.
- Файл `app_config.py` – конфигурация и настройки всего проекта.
- Файл `forms.py` – предоставляет все формы, необходимые для функционирования приложения.
- Файл `model.py` – представляет собой модели БД в виде их классов.
- `templates` – представляет собой папку с шаблонами html-страничек.

Диаграмма вариантов использования

При разработке системы составлена диаграмма вариантов использования, изображенная на рис. 3.

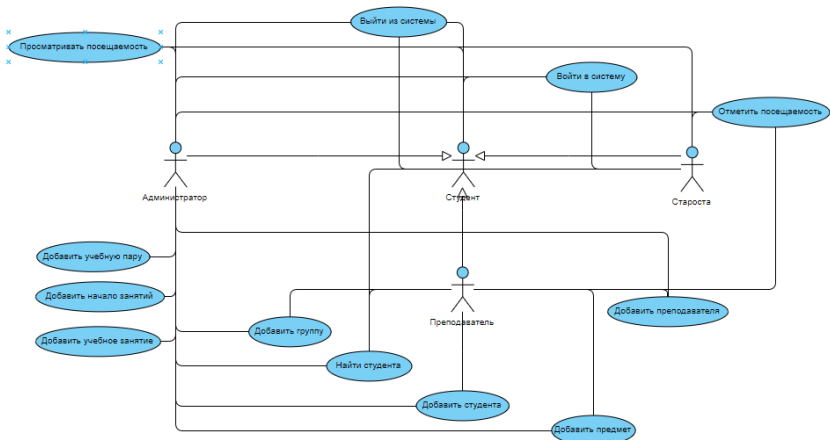


Рис. 3. Диаграмма вариантов использования

Диаграмма классов

При разработке системы составлена диаграмма классов, изображенная на рис. 4.

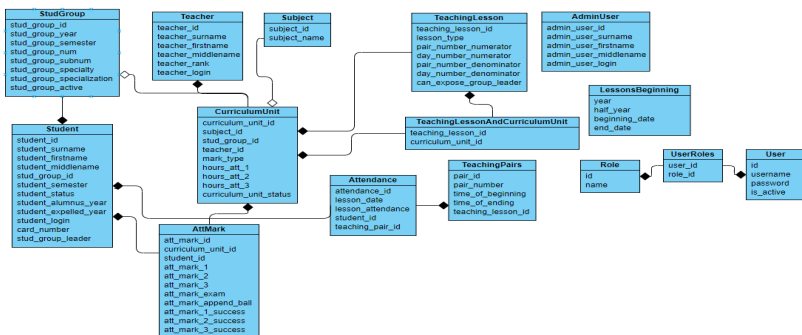


Рис. 4. Диаграмма классов для системы

Схема данных

При разработке системы составлена схема данных, изображенная на рис. 5.

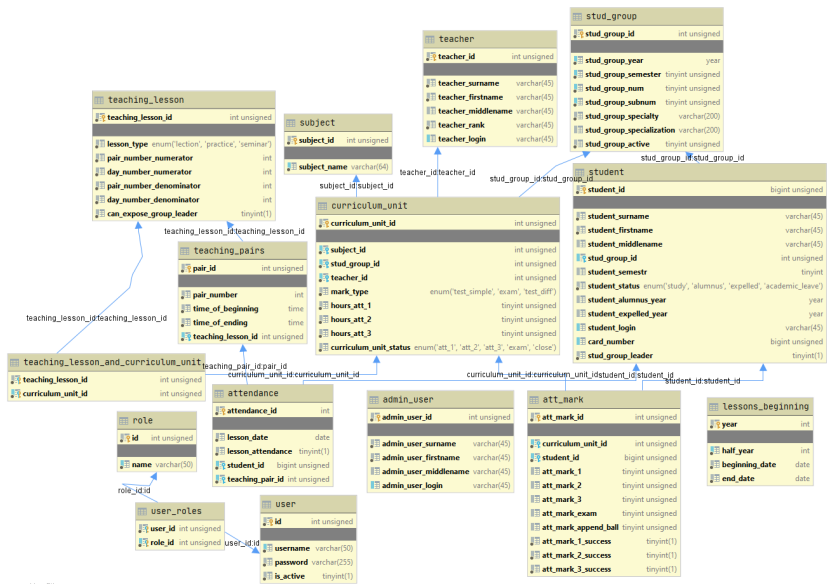


Рис. 5. Схема данных системы

Stud_group - таблица, содержащая группы студентов.

Student - таблица, содержащая в себе данные конкретного студента.

Teacher - таблица, содержащая данные о преподавателях в учебном заведении.

Curriculum_unit - таблица, содержащая единицы учебного плана.

Subject - таблица, содержащая сведения о предметах в учебном заведении.

Teaching_lesson - таблица, содержащая сведения о занятиях в учебном заведении.

Teaching_lesson_and curriculum_unit - связующая таблица между единицами учебного плана и занятиями в учебном заведении.

Attendance - таблица, содержащая сведения о посещаемости студента.

Att_mark - таблица, содержащая в себе аттестационные результаты студента.

Teaching_pair - таблица, содержащая в себе учебные пары на определенные дни.

Lessons_beginning - вспомогательная таблица для учета по первой неделе числителя/знаменателя.

User - таблица, содержащая информацию о пользователях в системе.

Role – таблица, содержащая информацию о ролях пользователя.

User_roles – связующая таблица между пользователями и их ролями.

Заключение

В ходе выполнения работы были проанализированы существующие решения для учета посещаемости, в результате анализа был получен вывод, что ни одно из них полностью не соответствует требованиям и не обладает необходимым функционалом.

Результатом выполнения работы является набор инструментов и веб-страниц, позволяющий управлять учетом посещаемости и возможность просмотра данных о ней.

Список литературы

1. Документация по Flask [Электронный ресурс]. – Режим доступа : <http://flask.pocoo.org/docs/1.0/>
2. Документация по Jinja [Электронный ресурс]. – Режим доступа : <http://jinja.pocoo.org/docs/2.10/>
3. Документация по SQLAlchemy [Электронный ресурс]. – Режим доступа : <https://www.sqlalchemy.org/>

Система анализа респираторной активности

П. О. Левчук

Студент магистрант

А. Ю. Иванков

Доцент

А. В. Максимов

Ассистент

Введение

Кашель является самым распространенным симптомом заболеваний органов дыхания и встречается у 20% больных. На данный момент времени диагностика кашля с помощью визуальных аналоговых шкал и специальных опросников отличается выраженной субъективностью.

В связи с этим было принято решение разработать систему анализа респираторной активности. В ходе предыдущих работ было разработано устройство для сбора кашлей, которое активируется при превышении порога звука и движении грудной клетки. Данная работа посвящена исследованию эффективности алгоритмов классификации собранных кашлей с использованием нейронных прямого распространения для использования в системе анализа респираторной активности.

1. Данные

Было собрано 1274 образца одиночных кашлей в формате .wav, частотой дискретизации 41КГц. Среди них 939 сухих кашля, 106 сухих ковидных кашлей, 163 влажных кашля, 172 свистящих кашля.

Для получения признаков для подачи на вход алгоритму мы будем использовать мел-кепстральные коэффициенты. Мел – психофизическая единица высоты звука, основанная на восприятии этого звука органами слуха человека. Это отражает тот факт, что люди гораздо лучше различают небольшие изменения высоты звука на низких частотах, чем на высоких. Так как это психофизическая величина, то существуют разные формулы. Одной из них является формула НТК (см. рис. 1):

$$M(f) = 1127 \ln(1 + f / 700)$$

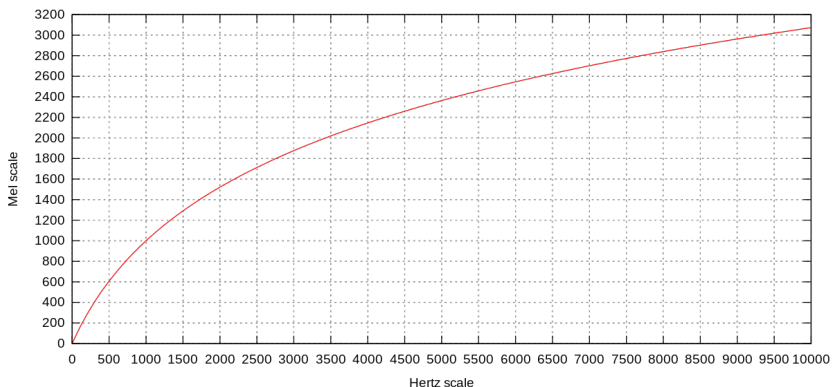


Рис. 1. Шкала перевода из Гц в мел

Для вычисления MFCC необходимо [1]:

1. Разделить сигнал на фреймы. Размер фрейма обычно около 20мс для распознавая речи и около 90мс для музыки. Чем меньше размер окна, тем проще различить импульсы, но сложнее различить тона.
2. Применить Short-time преобразование фурье на фреймы, получив спектры.
3. Получить мел-спектрограмму, перемножив спектры на треугольные окна (см. рис. 2).

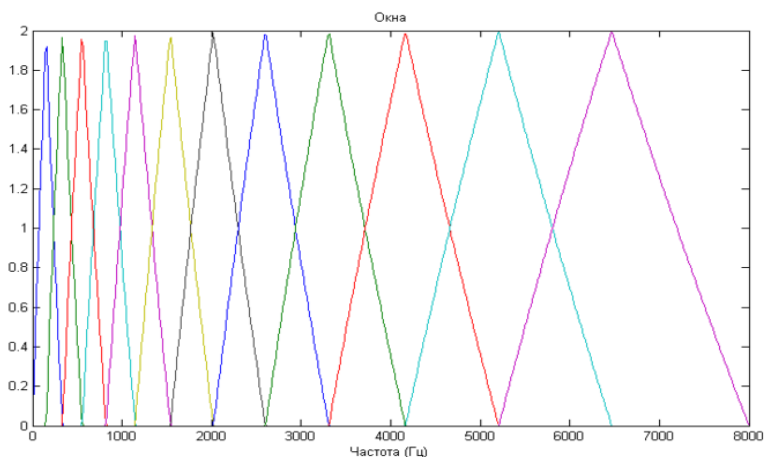


Рис. 2. Треугольные окна в шкале Гц

Далее нам необходимо получить мел-кепстр. Кепстр - один из видов обработки сигналов, функция обратного преобразования Фурье от логарифма спектра мощности сигнала. Позволяет описать график спектра относительно небольшим количеством значений.

4. Вычислить логарифм мощностей каждой мел частоты.
5. Применить дискретное косинусное преобразование.
6. Полученные амплитуды спектра это MFCC (см. рис. 3).

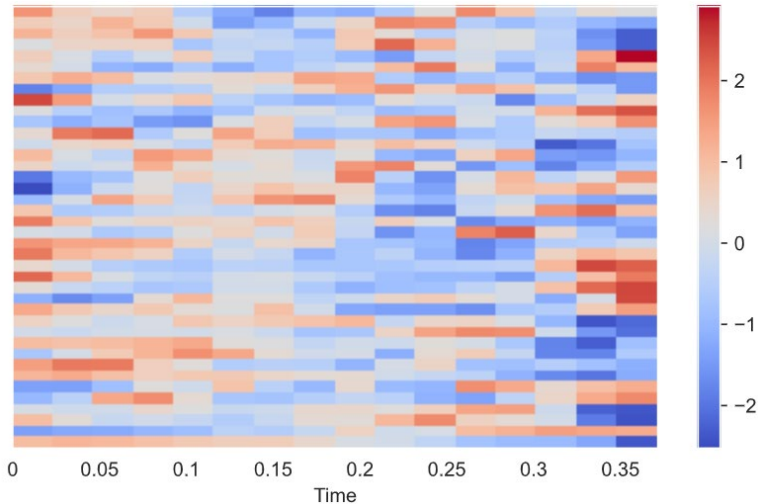


Рис. 3. MFCCs в нормализованном виде

Для нахождения MFCC использовалась библиотека librosa. После получения $[N_MFCC; M]$ массива, где N_MFCC – количество MFCC коэффициентов и M – количество фреймов. Вычисляем \min , \max , mean , median , std , kurtosis , skewness в строке для каждого коэффициента. Таким образом, получаем $7 * N_MFCC$ признаков для каждого образца.

2. Обучение

Для изучения эффективности нейронных сетей прямого распространения была разработана модель нейронной сети с помощью библиотеки pytorch. В качестве функции активации для задачи классификации была выбрана кросс-энтропия с весами 1.0 (Сухой), 3.0 (Мокрый), 3.0 (Свистящий), чтобы учесть несбалансированность выборки (листинг). Были использованы активация SELU и альфа-исключение [2], оптимизатор AdamW [3].

Архитектура нейронной сети

```

model_args = [
    torch.nn.Linear(feature_count, units),
    torch.nn.SELU(),
    torch.nn.AlphaDropout(dropout),
]
for layer_index in range(layers - 1):
    model_args.extend([
        torch.nn.Linear(units, units),
        torch.nn.SELU(),
        torch.nn.AlphaDropout(dropout)
    ])
model_args.append(torch.nn.Linear(units, n_classes))
return torch.nn.Sequential(*model_args)

```

При обучении использовались несколько случайных разделений выборки на 80% обучающую, 10% валидационную, 10% тестовую. Для увеличения объема обучающей выборки и улучшения устойчивости алгоритма на обучающую выборку был наложен белый гауссовский шум. При подборе гиперпараметров модели и признаков производилось обучение в течение 1000 эпох, из которых выбиралась с модель с наименьшим количеством потерь на валидационной выборке. Вычисления производились на рабочей станции с данными компонентами: AMD Ryzen 9 3950X, Geforce GTX 2070 Super, 2x16Гб 3200МГц ОЗУ, 1Тб SSD Samsung 970 Evo Plus. Подбираемые параметры:

1. Количество коэффициентов MFCC
2. Размер фрейма
3. Количество скрытых слоев
4. Количество нейронов в слое
5. Дропаут

Экспериментальным путем мы получили, что для получения подходящих необходим размер фрейма 25мс, количестве вычисленных MFCC коэффициентов от 30, от 5 скрытых слоев с 64 нейронами и дропаутом 0.1. График обучения нейронной сетт с наилучшими результатами изображена на рис. 4. Матрица запутывания этой сети изображены на рис. 5.

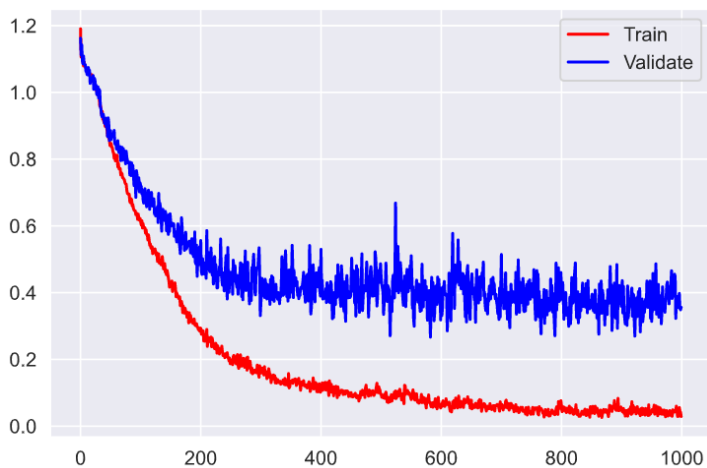


Рис. 4. Потеря/эпоха

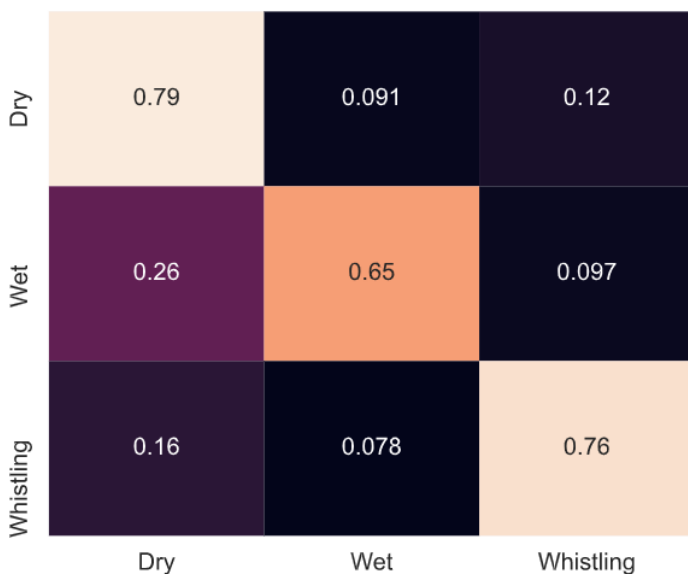


Рис. 5. Матрица запутывания

Заключение

Данная работа посвящена разработке классификатора кашля для использования в системе анализа респираторной активности. В итоге

данной разработки мы изучили эффективность нейронных сетей прямого распространения при классификации кашлей.

Дальнейшая работа включает в себя создание классификаторов не только по типу кашля, но и по полу и заболеванию, и поиск более эффективных алгоритмов.

Список литературы

1. Md. Sahidullah. Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition" / Md. Sahidullah, Goutam Saha // Speech Communication. Режим доступа: 10.1016/j.specom.2011.11.004
2. Günter Klambauer. "Self-Normalizing Neural Networks. Advances in Neural Information Processing Systems 30 / Günter Klambauer, Thomas Unterthiner, Andreas Mayr, Sepp Hochreiter // NIPS – 2017. Режим доступа: <https://arxiv.org/abs/1706.02515>
3. Ilya Loshchilov. Decoupled Weight Decay Regularization / Ilya Loshchilov, Frank Hutter // ICLR – 2019. Режим доступа: <https://arxiv.org/abs/1711.05101>

Нагрузочное тестирование микроконтроллера ESP8266

М. Б. Леонтьев

Студент магистр

П. С. Лысачев

Старший преподаватель

С. А. Зуев

Доцент

Введение

Микроконтроллер — это специальная микросхема, предназначенная для управления различными электронными устройствами. ESP8266 — китайский микроконтроллер от компании Espressif. Данные микроконтроллеры содержат в себе модуль IEEE 802.11, что позволяет создавать из них беспроводную сеть. По определению микроконтроллера должно производиться управление электронными устройствами. Для этого есть т.н. GPIO. На рис. 1 показана распиновка портов целевого микроконтроллера.

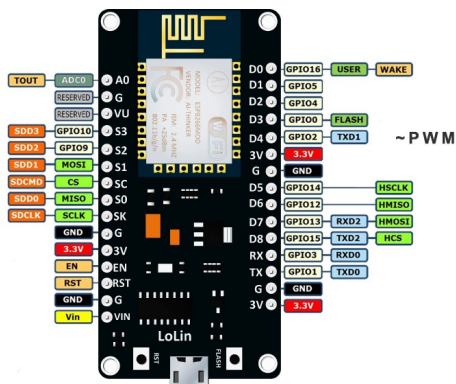


Рис. 1. Распиновка ESP8266

GPIO (от англ. General Purpose Input/Output) – универсальные порты ввода-вывода. Это порты микроконтроллера, которые могут

выступать в качестве источников контролирующих сигналов и управляться на уровне прошивки микроконтроллера. К примеру, на данном микроконтроллере присутствует 13 GPIO (GPIO 0-5, 9, 10, 12-16).

Микроконтроллер сам по себе особого интереса не представляет. Более интересно его применение в такой системе, как интернет вещей. В самом общем случае интернет вещей это набор микроконтроллеров, подключенный к серверу управления. Интернет вещей обычно предполагает наличие сети из «умных» устройств, взаимодействующих между собой, с возможностью создания сценариев для автоматизации в определенном смысле [1]. Интернет вещей должен обеспечивать отзывчивость при управлении и мониторинге. Теоретически, от объема данных при управлении микроконтроллерами может происходить увеличение задержки, что негативно сказывается на отзывчивости.

2. Постановка задачи

По некоторым главам теории автоматического управления [2] можно сделать вывод, что задержка между подачей управляющего сигнала и совершением микроконтроллером управляющего действия должна стремиться к минимуму. Применительно к рассматриваемому микроконтроллеру следует учитывать, что при интеграции его по IEEE 802.11 могут быть задержки между отправкой запроса и исполнением действия.

Если обратиться к datasheet на данный микроконтроллер [3], то откуда можно узнать, что из 13 GPIO эффективно доступны только 11. Это означает, что к микроконтроллеру можно подключить до 11 датчиков/актуаторов (устройств). Таким образом, может присутствовать зависимость между объемом пересылаемых между контроллером и сервером управления данных. Следует проверить, существует ли такая явная зависимость. Для этого необходимо протестировать данную систему.

3. Тестирование системы

Тестирование программного обеспечения – набор действий по планированию тестовых действий, их структуре, выполнению этих тестов и анализу полученных данных [4].

Выделяют две большие группы видов тестирования программного обеспечения по целям:

- функциональное тестирование;
- нефункциональное тестирование.

Функциональное тестирование – это группа тестирований, направленных на проверку соответствия имеющейся функциональности

системы поставленным функциональным требованиям, определяющим, какие именно действия пользователя должна поддерживать система, какие задачи она должна решать и то, насколько результаты её деятельности соответствуют реальности.

Нефункциональное тестирование – тестирование свойств, которые не относятся к функциональности системы. Данные свойства определяются нефункциональными требованиями, которые характеризуют программное обеспечение по определенным критериям. К примеру:

- надежность (реакция системы на непредвиденные ситуации);
- производительность (работоспособность системы под разными нагрузками);
- удобство (исследование удобства работы с приложением с точки зрения пользователя);
- масштабируемость (требования к горизонтальному или вертикальному масштабированию приложения);
- безопасность (защищенность пользовательских данных).

В контексте настоящей статьи будет производиться нефункциональное тестирование производительности системы. В качестве основного тестирования было выбрано нагрузочное тестирование.

Нагрузочное тестирование микроконтроллера – тестирование, нацеленное на проверку работы системы в условиях больших нагрузок (нахождения максимального предела запросов, при которых система «сохраняет работоспособность»).

4. Сценарий тестирования

В качестве сценария для нагрузочного тестирования будет рассматриваться ситуация, когда микроконтроллер является неким «агрегирующим датчиком». Под этой фразой в контексте такого тестирования будет предполагаться функциональная нагрузка микроконтроллера различными датчиками с последующим считыванием измеряемых данных, запаковкой в одну структуру и отправкой на сервер после его запроса.

По результатам пункта 1 ясно, что ESP8266 поддерживает подключение до 11 устройств. Таким образом необходимо предусмотреть тестирование для 11 случаев, каждый из которых обозначает пересылку между микроконтроллером и сервером пакетов, содержащих массивы из соответствующего количества данных.

Для большей точности в качестве измеряемой величины используются такты (англ. Ticks) процессора, т.к. задержки даже в порядках миллисекунд оказались крайне незначительны.

Для проведения тестирования была написана некая программная система, производящая 1000 измерений задержек между запросами к серверу и ответами от микроконтроллера. Аналогично была написана прошивка для микроконтроллера, сегмент которой представлен в листинге ниже.

Листинг

```
int packetSize = Udp.parsePacket();
if (packetSize)
{
    int len = Udp.read(incomingPacket, 255);
    if (len > 0)
        incomingPacket[len] = 0;

    int ii = 11;
    int arr[] = {random(1, 100), random(1, 100), random(1,
100), random(1, 100), random(1, 100), random(1, 100)
, random(1, 100), random(1, 100), random(1, 100),
random(1, 100), random(1, 100)};
    for(int i = 0; i < ii; i++)
    {
        replyPacket[i] = (byte)arr[i];
    }
    replyPacket[ii] = 0;
    sendMsg(replyPacket);
}
```

Конкретно, в листинге представлен пример кода тестирования для случая с 11 подключенными датчиками.

Алгоритм выглядит следующим образом:

1. Если итераций меньше 1000, то пункт 2, иначе пункт 10.
2. Сервер запускает таймер.
3. Сервер отправляет запрос на микроконтроллер.
4. Микроконтроллер получает запрос.
5. Микроконтроллер создает массив со случайными данными (моделирует считывание данных).
6. Микроконтроллер отправляет данные обратно на сервер.
7. Сервер получает ответ на запрос.
8. Сервер останавливает таймер и записывает данные в csv-файл.
9. Вернуться на 1 пункт.

И таким образом производятся все 11 моделирующих тестов.

5. Результаты тестирования

По результатам всех 11 тестирований собирается таблица в Excel, фрагмент которой показан на рис. 2

На первый взгляд, в этой таблице присутствуют некоторые выбросы, но для упрощения восприятия и анализа эти данные

необходимо визуализировать. Но необходимо выбрать наиболее подходящую диаграмму.

	A	B	C	D	E	F	G	H	I	J	K
1	1	2	3	4	5	6	7	8	9	10	11
2	70432	51641	71441	116171	77643	138306	78456	69386	52631	50336	66659
3	45675	64046	48376	70334	62760	134619	46561	59220	74151	46325	69056
4	56707	46773	46494	46186	47694	51477	47348	46573	50231	52082	49926
5	54174	50566	60326	45859	46623	49798	57689	45428	46349	46840	63840
6	40225	30432	30060	28925	30088	48464	32554	30020	34611	31203	57300
7	40367	31939	39996	38364	29918	36865	39310	34685	33059	28482	32628
8	35323	34830	32721	44153	32666	35719	33014	29202	33172	30548	33098
9	47517	36963	38352	31070	121296	32379	118050	35314	33620	32085	32573
10	29435	40609	111297	31257	35569	32631	39674	44172	31266	31796	33501
11	56943	28307	32023	38509	32172	35148	33744	46322	32117	29115	44882
12	42391	27308	35930	39599	38091	31290	32960	34865	32569	30752	33034
13	39345	28032	32912	35229	35548	32833	31527	93480	52414	32187	32160
14	38621	29585	40983	31041	36533	39885	32308	30909	32409	31626	34938
15	38485	36152	38588	37139	33901	33497	31754	43066	40187	30483	52524
16	39872	32852	32209	30198	42893	33853	39865	30753	32331	36258	82279
17	111808	28981	46541	36045	33194	32043	32781	32117	29584	47711	32092
18	45639	37628	42308	38739	36834	30144	32577	35211	31639	41350	45811
19	32070	28505	39686	31264	75684	39814	30260	32012	31456	31434	31418
20	40487	107027	37952	36777	33668	30494	31537	31744	47371	30151	30993
21	34925	30438	33528	32608	40074	32146	32169	34491	33743	29258	30942
22	40584	27924	29183	56788	31169	32358	34951	33771	29175	29197	31284

Рис. 2. Фрагмент таблицы с результатами тестирования

Для визуализации было принято решение использовать диаграмму размаха. Она позволяет сравнивать медианы, квартили, минимумы и максимумы по различным выборкам. Диаграммы-графики не подходят для визуализации такого объема данных, т.к. присутствуют те самые ранее замеченные выбросы. На рис. 3 показана итоговая диаграмма размаха.

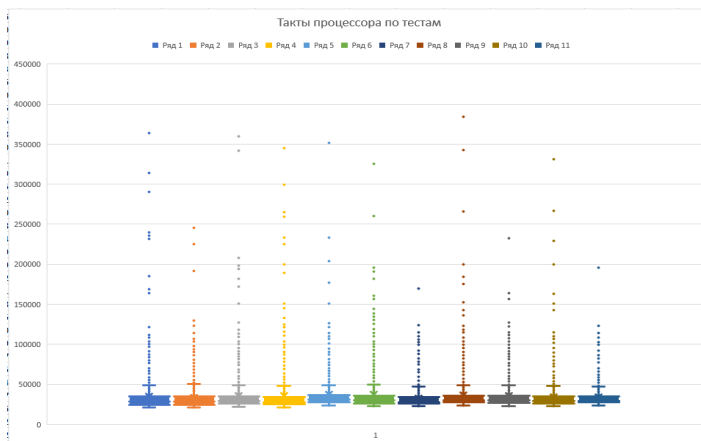


Рис. 3. Диаграмма размаха по результатам тестирования

Из вида данной диаграммы очевиден вывод, что зависимость между объемом пересылаемых данных между микроконтроллером и сервером полностью отсутствует.

Заключение

Было проведено нагрузочное тестирование микроконтроллера ESP8266. Моделировалось 11 тестов с возрастающим объемом пересылаемых данных. По результатам тестирования можно утверждать, что один микроконтроллер при подобных нагрузках не испытывает проблем при коммуникации с сервером.

Однако следует иметь в виду, что протестированный микроконтроллер не предполагается использовать в таких же «лабораторных условиях», а потому на задержку между командой и результатом могут влиять внешние факторы. К примеру, толщина стен, температурный режим, зашумлённость радиоканала и т. д.

Список литературы

1. Довгаль, В. А. Интернет Вещей: концепция, приложения и задачи / В. А. Довгаль, Д. В. Довгаль // Вестник Адыг. гос. ун-та. Сер. Ест.-мат. и техн. науки. – 2018. – С. 128-135.
2. Солодовников, В. В. Теория автоматического управления техническими системами : учебное пособие / В. В. Солодовников, В. Н. Плотников, А. В. Яковлев. – М.: Изд-во МГТУ, 1993. – 492 с. : ил.
3. ESP8266EX Datasheet (Version 6.6 Espressif Systems): [Электронный ресурс]. – Режим доступа: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf
4. Алджанов, В. ИТ-архитектура от А до Я: Теоретические основы. Первое издание / В. Алджанов. – М.: Litres, 2019. – 1272 с.

Разработка универсальной системы управления микроконтроллерами

М. Б. Леонтьев

Студент магистр

П. С. Лысачев

Старший преподаватель

С. А. Зуев

Доцент

Введение

Микроконтроллер – это специальная микросхема, предназначенная для управления различными электронными устройствами. Разработчики микроконтроллеров объединили процессор, память, ПЗУ и периферию внутри одного корпуса, внешне похожего на обычную микросхему. Микроконтроллер – это в каком-то смысле уже не процессор, но ещё и не компьютер. По определению должно производиться управление электронными устройствами. Для этого есть т.н. GPIO. На рис. 1 показана типовая схема с информацией о портах микроконтроллера.

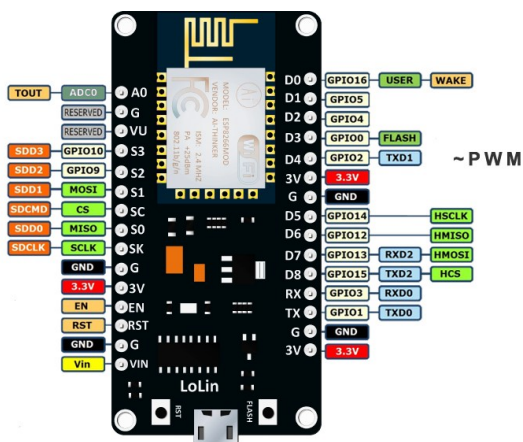


Рис. 1. Типовая схема портов микроконтроллера

GPIO (от англ. General Purpose Input/Output) – универсальные порты ввода-вывода. Это порты микроконтроллера, которые могут выступать в качестве источников контролируемых сигналов и управляться на уровне прошивки микроконтроллера. Из схемы на рис. 1 можно видеть, что на представленном микроконтроллере присутствует 13 GPIO. Это означает, что в общем случае с их помощью можно управлять 13 устройствами.

Существует такое понятие, как интернет вещей. Интернет вещей у многих ассоциируется с «умным» домом. Благодаря технологиям и устройствам, разработанным многими известными компаниями, пользователи могут совершать онлайн-покупки, регулировать температуру в комнате, включать свет и музыку, отдавая голосовые команды виртуальным помощникам [1]. В самом общем случае интернет вещей это набор микроконтроллеров, подключенный к серверу управления, коим в большинстве случаев выступают именно эти виртуальные помощники-станции. Однако, пользовательский сектор – лишь частный случай систем интернета вещей.

На самом деле, интернет вещей – понятие более сложное и абстрактное. Интернет вещей обычно предполагает наличие сети из «умных» устройств, взаимодействующих между собой, с возможностью создания сценариев для автоматизации в определенном смысле [2].

Задачи, поставленные перед системами интернета вещей, можно условно разделить на 4 большие класса, располагаемые в порядке увеличения сложности:

- мониторинг;
- управление;
- оптимизация;
- автономия.

Мониторинг – самый простой класс задач, представляющий собой просто сбор информации о каком-либо объекте.

Под задачами управления понимаются задачи, связанные с непосредственными воздействиями на окружающую среду или устройства.

Задачи оптимизации в большей степени применяются именно на промышленных производствах – к примеру, подключение станков к сети интернета вещей позволяет в режиме реального времени измерять и анализировать поступающие с них данные о производственном процессе.

Задачи автономии – самый сложный класс задач, предназначенный в первую очередь для «автономных» систем – «умных» роботов.

2. Анализ предметной области

На современном рынке представлены многие микроконтроллеры – от дешевых (до 5\$) и до промышленных (свыше 100\$). Интерес представляют дешевые микроконтроллеры, которые, в основной своей массе, уже имеют относительно серьезные вычислительные мощности. Из таких микроконтроллеров можно выделить микроконтроллеры от компаний Arduino, STMicroelectronics, Espressif. Из модельных рядов этих компаний интерес представляют только микроконтроллеры от Espressif, а именно ESP8266. Данные микроконтроллеры содержат в себе модуль IEEE 802.11, что позволяет создавать из них беспроводную сеть.

Существует множество протоколов для коммуникации в интернет-вещей – начиная от MQTT(SN) и заканчивая проприетарными. Так как разрабатываемая система предполагается интегрируемой, был выбран самый простой протокол – UDP по модели TCP/IP. Такой выбор обусловлен наименьшей задержкой.

На рынке присутствует множество решений «умных» домов на основе станций-контроллеров: Яндекс, Mail.ru, Google, Amazon, либо на основе открытых стандартов: Xiaomi, ASUS, Aqara, TP-Link. Большинство из этих систем позволяют кроссплатформенное «общение» с помощью API. Поэтому решено выбрать модель коммуникации системы по REST API.

3. Постановка задачи

Основная цель – разработка универсальной интегрируемой системы интернета вещей, в составе которой должны быть конкретные микроконтроллеры и серверное приложение для управления ими.

Необходимо под выбранный микроконтроллер (ESP8266) написать прошивку, разработать серверную систему управления и сформулировать механизм коммуникации между ними.

4. Стек технологий

Технологиями, используемыми при разработке серверной части, были выбраны:

- Платформа .NET Framework [3];
- язык программирования C# [3];
- СУБД SQLite.

5. Архитектура системы

Топология системы представлена на рис. 2. Как можно видеть, ESP8266 подключены к локальной сети по 802.11. Сервер подключен к этой же локальной сети либо по 802.11, либо по 802.3, в зависимости от

его сетевой карты. Клиенты (приложения) производят коммуникацию с системой через REST API с помощью JSON-запросов.

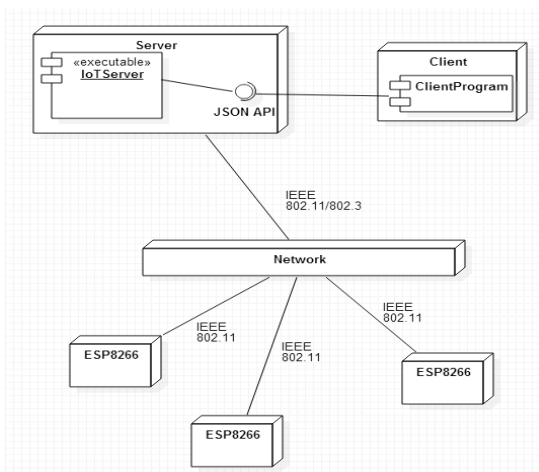


Рис. 2. Топология системы

На рис. 3 показана архитектура системы. Система состоит из 4 модулей.

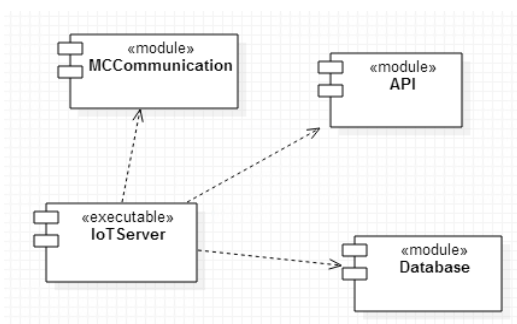


Рис. 3. Архитектура системы

Главный модуль – исполняемое приложение IoTServer, модуль MCCommunication отвечает за коммуникацию сервера с микроконтроллерами. Модуль Database – за работу с базой данных (сохранение, редактирование, выдача данных о микроконтроллерах). Модуль API предназначен для коммуникации клиентов-приложений с системой.

6. Варианты использования системы

На рис. 4 показана диаграмма вариантов использования для данной системы.

Из нее можно видеть, что в системе присутствуют 3 актора. Микроконтроллер пояснений не требует. Но его вариант использования «Напомнить о себе» подразумевает, что система следит за состоянием микроконтроллеров – включены ли они в систему и когда последний раз они «напоминали» о своем активном состоянии.

Актор Пользователь отличается от Оператора тем, что Оператор – это конкретно человек, имеющий непосредственный физический доступ к системе. Пользователь, в свою очередь, может быть как человеком, так и программой – «общение» с сервером производится через REST API, а потому для человека, знающего этот API, нет никаких препятствий перед его непосредственным использованием.

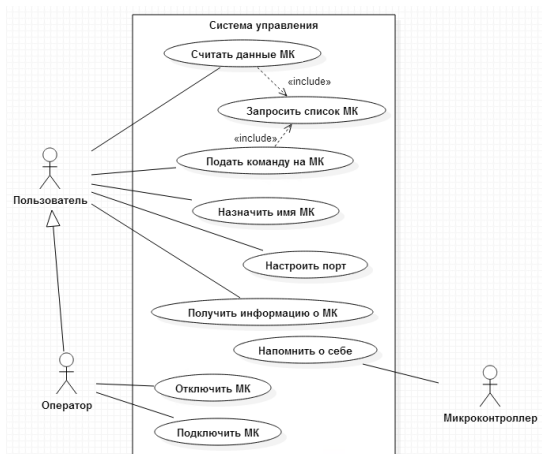


Рис. 4. Диаграмма прецедентов для системы

7. Результат работы системы

В данной разделе показано несколько примеров коммуникации с сервером с помощью REST API. Список микроконтроллеров в системе получается с помощью GET-запроса по адресу /getlist/, как показано на рис. 5.

```
localhost/mc/getlist
{"data":[{"id":"1","mid":"MC-12453"}, {"id":"2","mid":"MC-87655"}, {"id":"3","mid":"MC-96437"}]}
```

Рис. 5. Результат запроса списка микроконтроллеров

На рис. 6 представлен результат вывода краткой информации о микроконтроллере по его ID с помощью GET-запроса. Следует уточнить, что бесполезное на первый взгляд поле `lastactive` используется в варианте использования системы «Микроконтроллер» – «Напомнить о себе».

```
localhost/mc/get/1
{"name":"Lamp","lastactive":"09.05.2021 14.58"}
```

Рис. 6. Результат запроса краткой информации о микроконтроллере

GET-запрос по адресу `/getinfo/{id}` позволяет получить полную информацию о микроконтроллере, включая информацию обо всех портах. К примеру, у ESP8266 эффективно используемых GPIO всего 11 из 13 потенциальных, поэтому выводится массив из 11 элементов, что изображено на рис. 7.

```
localhost/mc/getinfo/1
{"ports":[{"num":"1","name":"Lamp1","type":"1","commentary":"Управление лампой на GPIO 1"}, {"num":"2","name":"Lamp2","type":"1","commentary":"Управление лампой на GPIO 2"}, {"num":"3","name":"Lamp3","type":"1","commentary":"Управление лампой на GPIO 3"}, {"num":"4","name":"","type":"0","commentary":""}, {"num":"5","name":"","type":"0","commentary":""}, {"num":"6","name":"","type":"0","commentary":""}, {"num":"7","name":"","type":"0","commentary":""}, {"num":"8","name":"","type":"0","commentary":""}, {"num":"9","name":"","type":"0","commentary":""}, {"num":"10","name":"","type":"0","commentary":""}, {"num":"11","name":"","type":"0","commentary":""}]}
```

Рис. 7. Результат запроса полной информации о микроконтроллере

Остальные запросы для управления системой выполняются по аналогии с представленными.

Заключение

Была разработана система, выполняющая поставленные задачи. Система позволяет подключать заранее прошитые специальной прошивкой микроконтроллеры и осуществлять управление и сбор данных с них с помощью запросов к API сервера. REST API позволяет

интегрировать эту систему в любые системы интернета вещей, поддерживающее коммуникацию со сторонними системами.

В данной системе не была реализована функциональность создания сценариев, например, включение одного микроконтроллера при выполнении какого-либо условия и т. д. Такое решение было принято на основе заключения, что сценарная часть системы управления зависит от области применения системы. А потому наилучшим вариантом является делегирование интерфейса управления другим системам – другие системы с помощью API могут реализовывать как сценарии, так и мониторинг в реальном времени.

Список литературы

2. Лочкарева, Т. Г. Интернет Вещей / Т. Г. Лочкарева // Инновационная наука. – 2016. – № 12-2. – С. 83-84.

3. Довгаль, В. А. Интернет Вещей: концепция, приложения и задачи / В. А. Довгаль, Д. В. Довгаль // Вестник Адыг. гос. ун-та. Сер. Ест.-мат. и техн. науки. – 2018. – С. 128-135.

4. Рихтер, Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. Мастер-класс. / Дж. Рихтер ; Пер. с англ. – 2-е изд., исправ. – СПб. : Питер, 2008. – 656 с. : ил.

Автоматизация тестирования RESTful API на основе BDD-подхода

А. В. Матвеев

Студент магистр

Н. К. Самойлов

Старший преподаватель

Введение

Применение современных методик позволяет командам разработки раньше принимать верные решения, обращаясь с требованиями как с тестами – то есть создавая сценарии использования новых возможностей. Чем раньше у разработчиков появляется возможность тестировать варианты использования нового функционала, тем раньше команда сможет получать отзывы о его работоспособности [1].

Обнаружение дефектов на самом раннем этапе позволяет получить рабочий функционал гораздо быстрее. От момента разработки, до момента получения нового функционала командой тестирования проходит много времени, а скорость, как и качество, имеет решающее значение.

Естественный язык часто используется для написания спецификаций системных требований, а также требований пользователей. Большинство спецификаций требований к программному обеспечению написано на естественном языке [2]. Однако спецификации естественного языка могут быть трудными для понимания. Когда требования написаны на естественном языке, могут возникнуть различные проблемы, так как, например, читатели и писатели могут использовать одно и то же слово для разных понятий или выразить одно и то же понятие совершенно по-разному.

В данной работе рассматривается задача разработки системы для написания исполняемых тестовых сценариев используя естественный язык.

1. BDD-подход

В программной инженерии разработка, управляемая поведением – это гибкий процесс разработки программного обеспечения, который поощряет сотрудничество между разработчиками, QA и нетехническими

или бизнес-участниками проекта программного обеспечения. Это побуждает команды использовать беседу и конкретные примеры для формализации общего понимания того, как должно вести себя приложение.

Важная составляющая разработки, основанной на поведении, является концепция использования специфичного предметно-ориентированный языка – естественного языка, понятного всем участникам процесса и позволяющего объединить постановку задачи, тесты и документацию воедино.

Спецификации поведения удобно и эффективно выполняют несколько задач. Единственный код, который должен быть написан разработчиком, – это код, удовлетворяющий спецификациям поведения. Нет необходимости реализовать дополнительную функциональность, если нет случаев, когда она будет использоваться.

При реализации спецификации поведения в виде автоматизированных тестов они представляют собой исполняемую документацию, которая становится частью набора тестовых ресурсов.

Спецификации поведения не являются частью официального документа с требованиями или юридического контракта – они представляют собой готовую документацию и средства тестирования.

2. Требования к системе

При разработке рассматриваемой системы учитывались следующие требования:

- реализация в виде библиотеки;
- простота интеграции в проект;
- максимальная простота и свобода написания сценария с возможностью проверки результата;
- простая интеграция с CI/CD платформами;
- формирование отчета по выполненным тестовым сценариям.

3. Реализация генерации тестового сценария

В этом разделе описана реализация для генерации исполняемого файла из требований написанном на естественном языке. При разработке модуля использовался язык программирования Java и его возможность исследования данных о программе во время её выполнения. Рефлексия позволяет исследовать информацию о полях, методах и конструкторах классов.

Процесс обработки тестовых случаев описывается диаграммой, представленной на рис. 1.

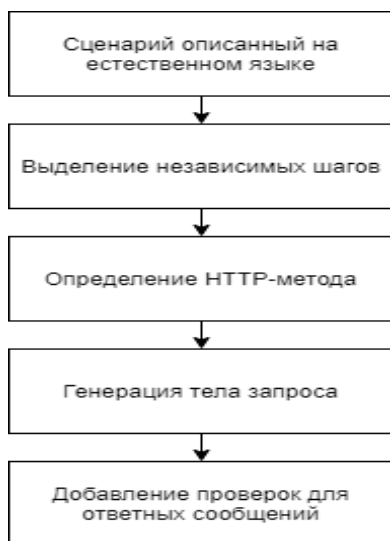


Рис. 1. Блок-схема алгоритма генерации сценария

Формат сценария опирается на известную спецификацию поведения Gherkin [3]. Сценарий тестирования может представлять из себя неограниченное количество шагов, каждый из которых является отдельным HTTP-запросом. Изначально из сценария выделяются независимые шаги на основе ключевых слов «Given» и «When». Тип запроса определяется на основе следующих ключевых слов: Create, Read, Replace, Modify и Delete. Введенные ключевые слова соответствуют HTTP – методам: POST, GET, PUT, PATCH и DELETE соответственно [4]. Тело запроса генерируется на основе существующих в проекте сущностей, а значения полей для них переопределяется указанными в сценарии. Для каждого запроса добавляется утверждение, которое необходимо проверить после получения ответа на отправленный запрос.

4. Реализация системы исполнения

Для исполнения сценариев была реализована соответствующая библиотека, позволяющая выполнять тестовые сценарии на работающем окружении.

Действия необходимые для внедрения данного процесс могут быть описаны диаграммой, представленной на рис. 2.



Рис. 2. Блок-схема алгоритма исполнения сценария

Для корректного запуска сценария необходимо указать подробности о системе, которую пытаемся проверить и для которой будет исполняться тест. Как минимум в конфигурационном файле должны быть указаны такие подробности, как имя пользователя для доступа к внешнему интерфейсу, пароль для вышеуказанного пользователя, хост и порт на котором доступна система.

По результату выполнения будет сформирован отчет по каждому выполненному сценарию в html формате с подробностями по каждому из шагов и с результатами выполненных проверок.

5. Результаты разработки

Разработанная система позволяет написать сценарий на естественном языке, который будет являться одним из способов использования системы реальным пользователем. В листинге приведен пример использования разработанной библиотеки.

Листинг

Сценарий тестирования

Feature: Publish a book to the shop

Background:

I`m an administrator of an online book store and I want to be able to add a new book

Scenario: Adding a new book

Окончание листинга

```
Given Create Book with Genre='Science Fiction'  
and Title='Star Wars'  
Then I receive successful response and BODY.ID is non-empty.  
When Read Book with Title='Star Wars'  
Then I receive successful response  
and BODY.ID is non-empty  
and BODY.Genre='Science Fiction'
```

Данный сценарий после сборки проекта становится исполняемым. Он проверяет на работоспособность интерфейсы отвечающие за создание книги, а также получения ее по имеющему идентификатору, проверяя каждый из ответов тестируемой системы на корректность.

Заключение

По результатам выполненной работы была разработана система, позволяющая покрывать разрабатываемый API, различными сценариями тестирования, автоматически при этом формируя документацию, доступную для понимания любому читателю. Разработанная библиотека обладает возможностью интеграции с CI/CD платформами, позволяя внедрить новое решение тестирования в уже настроенный процесс. Реализованная система является универсальным инструментом и может быть внедрена в проекты любого масштаба.

Список литературы

1. Almentero, E. Towards Software Modularization from Requirements / E. Almentero, J.C.S.P. Leite, C. Lucena // SAC 2014 : Symposium on Applied Computing (Gyeongju Republic of Korea, March 2014). – New York: ACM, 2014. – P. 1007-1012.
2. El-Far, I. K. Model-Based Software Testing / I. K. El-Far, J. A. Whittaker // Encyclopedia Of Software Engineering. – Wiley, 2001. – 10 p.
3. Gherkin Reference [Электронный ресурс] : Cucumber Open Docs. – Режим доступа : <https://cucumber.io/docs/gherkin>
4. Using HTTP Methods for RESTful Services [Электронный ресурс] : Learn REST : A RESTful Tutorial. – Режим доступа : <https://www.restapitutorial.com/lessons/httpmethods.html>

Расширение для поддержки мультиарендности в работе приложения

Н. В. Мидов

Студент магистр

А. А. Вахтин

Доцент

Введение

Применение быстрого решения и создания приложения для автоматизации процессов клиента является неотъемлемой частью разработки. Большой процент существующих проблем является типовым и для их решения существуют фреймворки. В данной работе будет рассматриваться анализ и разработка расширения для CUBA Platform.

Cuba — Высокоуровневая Java-платформа с открытым кодом для создания корпоративных информационных систем, которая уже обладает большим рядом возможностей. Основные из них:

- декларативное создание UI: компоновка экранов в XML, инициализация и обработка событий в классах Java;
- библиотека data-aware визуальных компонентов. Есть все стандартные, плюс специфические, например, универсальный фильтр данных, поля выбора связанных сущностей с разнообразными возможностями, таблица с группировками;
- экраны работают в веб (AJAX) и в десктоп (Swing) клиентах, исходники общие;
- метаданные — расширенная информация о модели данных, проектирование модели данных «от сущностей к таблицам»;
- мягкое удаление записей в БД;
- управление правами доступа на уровне операций с сущностями, их атрибутов и отдельных экземпляров, экранов и компонентов UI;
- подключаемая при необходимости функциональность: генератор отчетов, модуль бизнес-процессов с визуальным редактором, полнотекстовый поиск, работа с кредитными картами;
- поддерживаются PostgreSQL, MS SQL Server, Oracle, HSQL;
- стандартный деплоймент на любом веб-контейнере;

- горизонтальное масштабирование на кластере серверов, можно отдельно средний слой, отдельно веб-серверы.

Как и любая высокоуровневая Java-платформа Cuba имеет возможность расширения своего функционала готовыми решениями, в данном случае для этого используются аддоны. В частности мы рассмотрим аддон для поддержания мультиарендности в приложении.

1. Понятие мультиарендности

Мультиарендность — элемент архитектуры программного обеспечения, где единый экземпляр приложения, запущенного на сервере, обслуживает множество организаций-клиентов («арендаторов») [1]. Мультиарендность сопоставляется с архитектурой из множественных экземпляров, где для каждой организации-клиента создаются отдельные программные экземпляры. В мультиарендной архитектуре программные приложения работают одновременно с несколькими конфигурациями и наборами данных нескольких организаций, а каждая организация-клиент работает со своим экземпляром виртуального приложения, видя только свою конфигурацию и свой набор данных [2]. Основная идея и отличия от классической архитектуры приложения показаны далее (рис. 1).

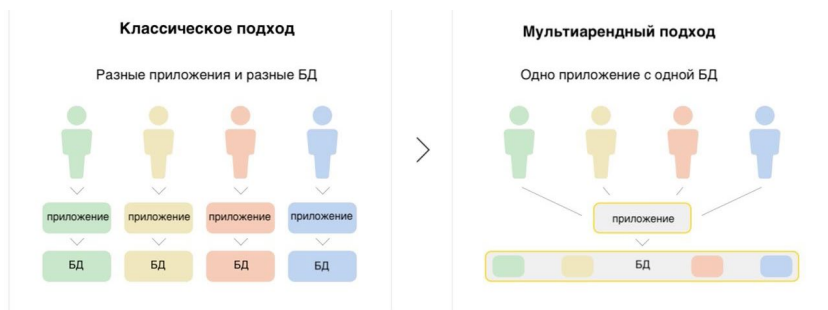


Рис. 1. Классическая и мультиарендная архитектура приложения

2. Требования к системе

При разработке текущей работы обращалось внимание на следующие пункты:

- реализация в виде расширения;
- минимальные трудозатраты при интеграции с существующим проектом;
- минимальная простота дальнейшего использования продукта.

3. Анализ способов реализации

В процессе выполнения работы были проанализированы несколько видов создания мультиарендности.

Одним из самых популярных подходов является использование нескольких БД одним приложением. При таком подходе код разделяется между клиентами (используются общие файлы UI и бизнес логики), а данные логически (а возможно и физически) разделяются между собой.

Преимущества:

- простая расширяемость (чтобы добавить клиента достаточно создать новую базу данных и настроить конфигурационный файл);
- простое масштабирование – можно разнести базы данных разных клиентов на отдельные сервера;
- индивидуальность – можно производить индивидуальные настройки для некоторых клиентов (даже размещать базу на другом СУБД);
- простое резервное копирование данных – если что-то сломалось у одного клиента, во время отката БД, других, это никак не заденет.

Недостатки:

- стоимость: количество поддерживаемых баз данных одним сервером ограничено, а значит, для такого решения потребуется больше аппаратных ресурсов, в отличие от метода, где всё хранится в одной базе, больше железа – больше администраторов, площадь серверных и стоимость электроэнергии;
- быстродействие: когда на одном сервере располагается несколько баз данных, суммарный объём которых больше оперативной памяти будет происходить скидывание данных в файл подкачки, а, следовательно, обращение к жёсткому диску, что очень медленно.

Следующим вариантом является использование одной БД, но разных схем. Для реализации этого способа нам помогут схемы. По сути, схемы это «пространства имён», которые содержат определённые ресурсы, например таблицы, и на которые можно давать определённые разрешения.

Преимущества:

- благодаря схемам, разделение доступа происходит на уровне СУБД и от нас не требуется дополнительной разработки;
- меньше баз данных – меньше аппаратных ресурсов;
- неплохая расширяемость – при добавлении клиента, создаётся новая схема на основе стандартной и настраивается доступ. Не смотря на то, что все схемы созданы на основе стандартной, они могут претерпевать некоторые изменения, так как изоляция сохраняется, а следовательно есть возможность редактировать колонки, таблицы.

Недостатки:

- проблема с резервным копированием и восстановлением, ведь база данных одна, если слетели таблицы у одного клиента, простой откат базы вернёт в прошлое данные всех клиентов, а это неприемлемо. Тут потребуются выборочный откат и слияние старых и новых данных, процедура немного сложнее, чем просто откатить всю базу целиком.

И последним из популярных решений является использование общей базы данных, общей схемы. Для реализации такого варианта необходимым условием будет введение дополнительного поля TenantID для разделения информации между заказчиками.

Преимущества:

- стоимость: в одном комплекте таблиц у нас будет храниться всё;
- добавление клиента: просто создание новой записи в таблице клиентов.

Недостатки:

- никакой гибкости: все клиенты используют один набор таблиц и колонок. Появление клиента-индивидуалиста – примета к костылям и заплаткам;

- потраченное время на разработку на разработку собственной системы разделения прав;

- с резервным копированием и восстановлением существуют проблемы, если в предыдущем подходе есть возможность удалить таблицу целиком и восстановить её, то тут придётся выискивать в одной таблице нужные записи, и перезаписывать их, что обойдётся дороже в производительности.

После анализа было принято решение использовать последний метод для первоначальной разработки.

4. Пример приложения

Данный пример показывает работу системы, которая обслуживает крупные магазины. В нашем случае это будут Costco и Walmart. Приложение содержит в себе сущности пользователей и магазинов. В процессе работы расширение для мультиарендности логически разделит данные между двумя магазинами и ограничит видимость чужих данных.

На рис. 2 изображена таблица со всеми пользователи. На данном этапе мы используем учётную запись главного администратора, поэтому ограничения видимости не происходит и мы видим всю метайнформацию.

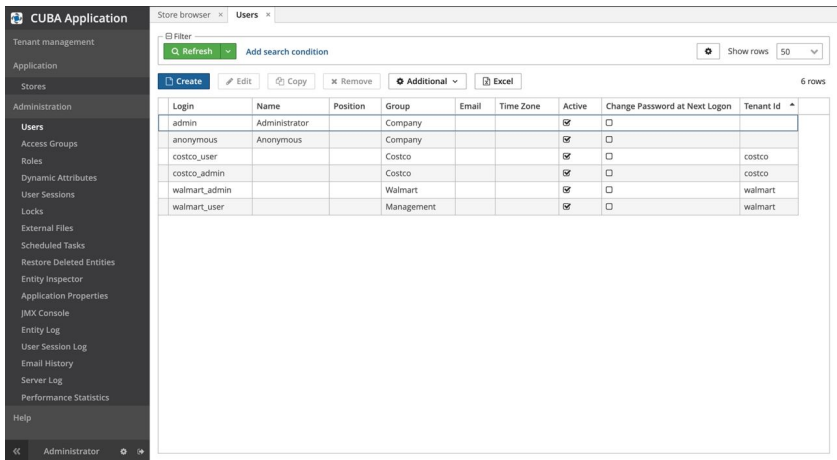


Рис. 2. Экран пользователей

На рис. 3 изображена таблица со всеми магазинами в системе.

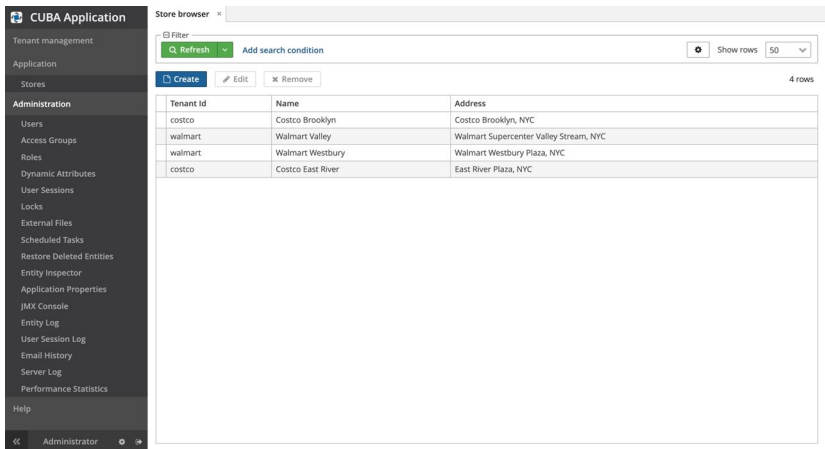


Рис. 3. Экран магазинов

Следующие два примера на рис. 4 и рис. 5 будут содержать информацию исключительно о магазинах, к которым относится текущий пользователь, так как будет использоваться уже не учётная запись главного администратора.

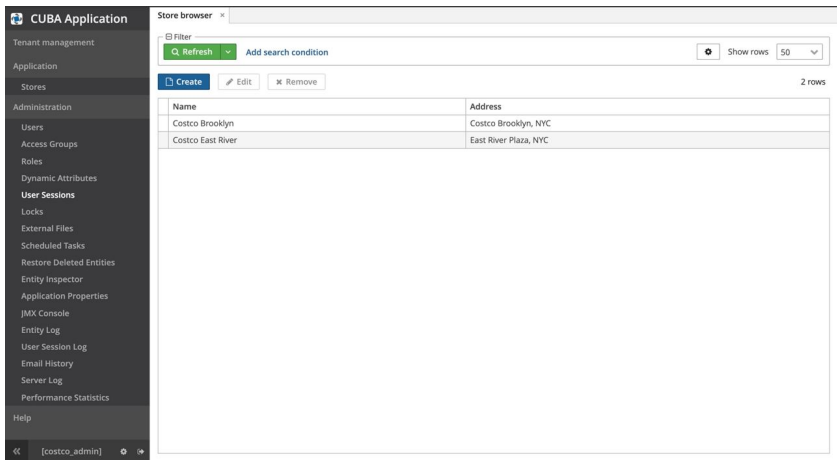


Рис. 4. Экран магазинов

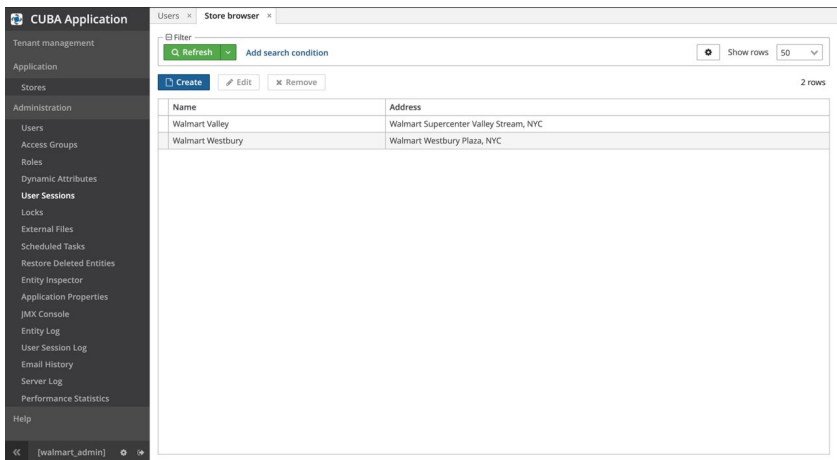


Рис. 5. Экран магазинов

Заключение

В результате данной работы было проанализировано множество подходов к решению проблемы мультиарендности, были разработаны расширения для крупной Java-платформы. Данное расширение позволяет использовать одну систему для обслуживания нескольких клиентов, работая в обычной конфигурации.

Список литературы

1. Java Platform API Specification [Электронный ресурс]. – Режим доступа: <https://docs.oracle.com/javase/7/docs/api/>
2. Пайлон, Д. UML 2 для программистов / Д. Пайлон, Н. Питмен; пер. с англ. Е. Матвеев. – СПб.: Питер, 2012. – 240с.

Фреймворк для обучения и тестирования детекторов лиц на изображении

Д. С. Митин

Студент магистрант

А. В. Акимов

Старший преподаватель

Введение

Задача распознавания объектов включает в себя ряд задач компьютерного зрения, которые в итоге позволяют идентифицировать объект.

Существует три задачи компьютерного зрения.

Классификация изображений – определение для изображения его класса из заданного конечного набора классов.

Обнаружение и локализация объекта – определение наличия объектов на изображении и указание на их местоположение с помощью ограничительной рамки.

Формирование множества обучающих данных имеет принципиально важное значение для успешного решения задач машинного обучения. Зачастую задачи машинного обучения сводятся именно к правильному формированию обучающего множества. Также результаты в обученной модели сильно зависят от соотношения числа объектов разных типов в выборке. В общем случае для получения наиболее удовлетворительного алгоритма обучения необходимо использовать обучающее множество достаточно большого размера.

Проблема несбалансированности обучающей выборки в задачах алгоритмов обучающей классификации также напрямую связана с рассматриваемой проблемой. Эта проблема возникает в ситуациях неравномерного представления разных классов, когда одни классы изображений представлены данными значительно меньшего размера, чем другие. Следовательно, существует необходимость в разработке эффективных методов и алгоритмов, улучшающих качество машинного обучения и, прежде всего, классификации изображений в условиях недостаточного количества обучающих данных.

Суть данной работы заключается в исследовании и сравнении подходов к размножению обучающих выборок, однако для этого нужна среда, в которой алгоритмы детектирования можно было бы обучать на размноженных выборках и затем тестировать. Основой для этого используются встроенные в Matlab инструменты работы с каскадным детектором.

Основные методы распознавания лиц

При распознавании лиц возникает ряд трудностей. Самые основные из них: старение, перекрытие лица, повороты и наклоны головы, вариативность углов освещения. Каждый из этих факторов приводит к некоторым неточностям при распознавании – детектор не может опознать все черты человека и поэтому либо выделит лицо частично, либо не там, где нужно или же вообще не обнаружит.

Изначально построение алгоритмов обучения не было таким, как сейчас: всё сводилось к полуавтоматическому алгоритму, в котором человек сам расставлял контрольные точки на изображении и алгоритм использовал набор геометрических тел, затем были введены модели, которые содержали уже целый ряд признаков для распознавания. Первой из таких моделей, которая дала начало развитию автоматическим, стала Eigenface (собственные грани) [1].

Она имела ряд ограничений в сравнении с современными, однако послужила стартом для их развития.

Построение этой модели начинается с подготовки обучающей выборки с изображениями лиц. Затем вычисляется среднее изображение (Рисунок 1а) и вычитается из исходных (Рисунок 1б). На основе полученных результатов вычисляются собственные вектора и значения ковариационной матрицы. Целью является отбор основных компонент посредством сортировки собственных значений по убыванию, чтобы ранжировать соответствующие им собственные векторы.

Метод Виолы-Джонса. Позволяет распознавать лица в полностью автоматическом режиме. Ввиду низкой алгоритмической сложности можно использовать для обнаружения объектов в режиме реального времени. Метод основан на применении целого ряда взаимосвязанных подходов и технологий, в основе которых лежит идея построения сильного классификатора путем объединения слабых классификаторов.

Метод включает в себя использование признаков Хаара в процессе анализа изображений. Слабый классификатор, используемый на низком уровне в качестве алгоритма принятия решения о классификации, строится на основе значения, полученного в результате применения признака Хаара [2].



а – среднее изображение, б – получение уникальных черт изображения путем вычитания среднего изображения

Рис. 1. Изображение

Так же используется технология бустинга или усиление слабых классификаторов: алгоритм AdaBoost является мета-алгоритмом машинного обучения, который в процессе обучения выстраивает набор базовых алгоритмов обучения для повышения их эффективности. Это алгоритм адаптивного повышения в том смысле, что каждый следующий классификатор построен на объектах, которые плохо классифицируются предыдущими классификаторами [3].

AdaBoost в цикле вызывает слабый классификатор и при каждом вызове обновляется распределение весов, которое соответствует требованиям каждого из объектов в обучающем наборе для классификации. В итоге новый классификатор сосредотачивается на этих объектах.

Метод имеет ряд преимуществ. В их число входит возможность обнаружения большого количества объектов на исходном изображении, возможность обучения классификатора на нахождение любого объекта, наличие большого количества различных открытых библиотек (к примеру OpenCV с расширенным количеством признаков Хаара) и небольшое количество ложных срабатываний при поиске объекта. Метод требует достаточно большого объема обучающей выборки.

2. Основные подходы к размножению обучающих выборок

Существует два хорошо известных подхода к формированию синтетических обучающих выборок, которые можно использовать для обучения алгоритмов машинного обучения в задачах, где получение достаточного количества данных затруднено или требует слишком много времени для разметки, и в то же время обучение на синтетических данных позволяет достичь качества, сопоставимого с обучением на реальных данных или превосходящего его. Поэтому сейчас на практике

используются два типа синтетических данных: дополнительные обучающие примеры, полученные из реальных с помощью размножения, и синтетические данные, полученные с помощью трехмерного моделирования.

Первый метод – распространение данных – широко используется при обучении свёрточных нейронных сетей. Нейронные сети имеют большое количество параметров, и даже самых больших выборок недостаточно для их обучения. Поэтому к изображениям применяются различные преобразования: зеркальные отражения, вращения, перемещения, масштабирование, искажающие деформации и т. д. С помощью таких преобразований можно получить только синтетические изображения, близкие к обучающей выборке.

Второй способ создания синтетических данных – это 3D-моделирование. Он определяет параметризованную 3D-модель. В качестве параметров используются случайные переменные. Параметры модели выбираются, и с фиксированными параметрами 3D-модель преобразуется в изображение с помощью графического движка. Этот метод имеет ограниченное применение, поскольку для каждой задачи требуется собственная 3D-модель с фотореалистичными текстурами.

3. Средства тренировки и запуска каскадных детекторов в Matlab

В Matlab существует ряд методов, которые позволяют обучить с необходимыми параметрами или взять уже обученный каскадный детектор и применить его.

Метод `trainCascadeObjectDetector` [4] может принимать в себя следующие параметры для обучения (представлены основные):

`positiveInstances` – положительные образы (с нужным объектом на изображении), заданные, например, как таблица с двумя столбцами. Первый столбец таблицы содержит имена файлов изображений, заданные как символьные векторы или строковые скаляры. Каждый ограничивающий прямоугольник имеет формат $[x \ y \ width \ height]$ и определяет местоположение объекта на соответствующем ему изображении.

`negativeImages` – отрицательные образы (соответственно, без объекта на изображении, один из способов задания: указывается путь к директории, их содержащей).

`ObjectTrainingSize` – Минимальный размер объекта поиска. Задаётся как два значения $[w, h]$.

`NumCascadeStages` – максимальное количество стадий для обучения и одновременно слоев будущего каскада. С увеличением

количества стадий увеличивается качество, время обучения и требуется больше обучающих данных.

FeatureType – свойство-значение. Вместо значений можно подставить один из вариантов: сущности хаара (Haar), локальные бинарные шаблоны (LBP) или гистограмму ориентированных градиентов (HOG).

Обученный каскадный детектор сохраняется в файл по предварительно заданному пути в формате *.xml и в дальнейшем его можно использовать для распознавания.

Практические результаты

В результате был написан фреймворк для обучения и распознавания объектов. Применив **trainCascadeObjectDetector** с определёнными параметрами, получен обученный каскадный детектор. Далее этот файл указывается при вызове самого метода детектирования **vision.CascadeObjectDetector**.

Тестовая выборка CMU+MIT отлично подходит под исследование возможностей детектирования ввиду наличия в ней разнообразных изображений с разной контрастностью, освещённостью и положением объектов обнаружения. Указав её для детектора были получены следующие результаты (Рисунок 2). Из 340 лиц на изображениях верно и ошибочно обнаружено 318 и 17 соответственно, что составляет 93,53 процента правильно найденных лиц от их общего числа. Ошибки в результатах распознавания в основном связаны со слабой яркостью самих изображений и наклоном головы, а также тем, что объект сливается со сценой. На данный момент ведётся поиск наиболее оптимальных параметров для обучения каскадного детектора и тем самым улучшением результатов его работы.



Рис. 2. Примеры результатов детектирования

Заключение

На данный момент создан фреймворк для обучения и распознавания фронтально расположенных лиц. Обучение детектора производилось на оригинальной выборке Виолы-Джонса.

Проведены замеры производительности работы детектора, которые на данный момент пока не достигают показателей встроенного. Предметом дальнейшего исследования является подбор оптимальных параметров для обучения каскадного детектора.

Список литературы

2. Eigenfaces [Электронный ресурс]: Computer Science Department, Department of Electrical & Computer Engineering. / scholarpedia – Электрон. журн. – Режим доступа : <http://www.scholarpedia.org/article/Eigenfaces>.

3. Viola, P. Robust Real-Time Face Detection / P. Viola, M. Jones // International Journal of Computer Vision. – Netherlands: Kluwer Academic Publishers. – 2004. – № 57(2). – P. 137-154

4. Мурыгин, К. В. Особенности реализации алгоритма AdaBoost для обнаружения объектов на изображениях / К. В. Мурыгин // Штучний інтелект. – 2009. – № 3. – С. 573-581.

5. trainCascadeObjectDetector [Электронный ресурс] : IT компания. / MathWorks – Электрон. журн. – Режим доступа : <https://www.mathworks.com/help/vision/ref/traincascadeobjectdetector.html>

Разработка ПО контроллера системы отопления с адаптивными функциями управления

И. Е. Никонов

Студент магистр

А. Ю. Савинков

Профессор

Введение

Рассмотрим систему, предназначенную для поддержания температуры теплоносителя на одном уровне. Такой системой может быть котельная, поддерживающая температуру воды в трубах, система отопления, поддерживающая температуру в доме или зерновая сушилка, поддерживающая температуру горячего воздуха, который сушит зерно. Проблема возникает в момент начального нагрева, потому как нагревательный элемент имеет некоторую инертность, то есть продолжает нагревать теплоноситель после выключения нагрева за счет собственной внутренней энергии. Кроме того, имеется некоторая задержка из-за задержки между снятиями показания термодатчиком. Перегрев может вызвать серьезные проблемы, для каждой системы свои. В связи с этим стоит задача «умного» управления нагревом.

Как правило, для этой задачи используются пропорционально-интегрально-дифференцирующий (ПИД) регуляторы [1]. Однако такие устройства имеют свои недостатки, в их числе высокая сложность эксплуатации, в силу того, что ПИД регуляторы требуют тонкой настройки для каждой конкретной системы.

Данная работа посвящена разработке программного обеспечения микроконтроллера, управляющего нагревательной системой, не требующей сложной настройки.

1. Аппаратные средства реализации

В качестве одного был выбран микроконтроллер STM32F103C8T6 на базе 32 битного процессора ARM Cortex-M3. Данный контроллер работает на частоте 72 МГц, что является достаточным для данной задачи [2]. Также из преимуществ можно выделить компактность и низкую стоимость (рис. 1).

Данный термодатчик способен измерять температуры от -55 до 125 градусов Цельсия, при этом в диапазоне от -10 до 85 градусов Цельсия погрешность составляет не более 0.5 градусов. Обмен информацией и командами осуществляется по протоколу 1-Wire. А благодаря возможности использовать «паразитное» питание, термодатчик работает как с использованием трех, так и с использованием двух контактов [3].

Несмотря на все достоинства протокола 1-Wire, разработка программного обеспечения для работы с ним является весьма непростой задачей. Для упрощения взаимодействия с термодатчиками, используется последовательный драйвер линии 1-Wire DS2480B (рис. 3). Данный драйвер позволяет производить сообщение микроконтроллера с устройствами, работающими по протоколу 1-Wire, используя протокол UART, что значительно упрощает поставленную задачу.

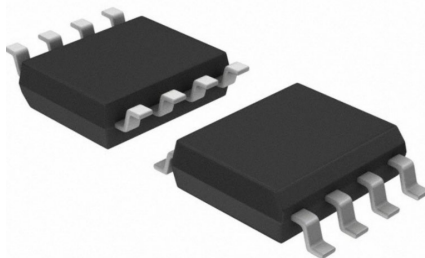


Рис. 3. Драйвер линии DS2480B

Для управления нагревательным элементом используется электромагнитное реле, позволяющее микроконтроллеру управлять током большой мощности (рис. 4).



Рис. 4. Электромагнитное реле

2. Макет

На основе описанных выше устройств был собран макет. На рисунке ниже изображен микроконтроллер с подключенными к нему драйвером линии и термодатчиком.

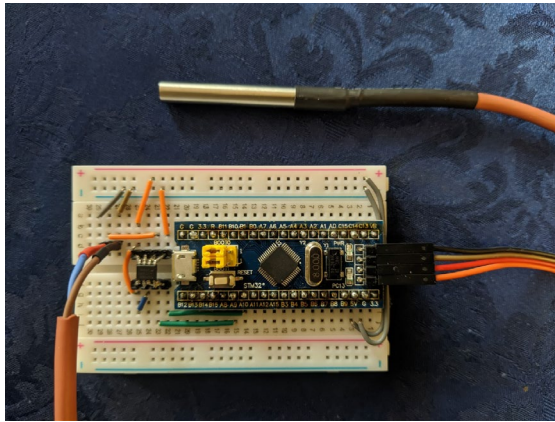
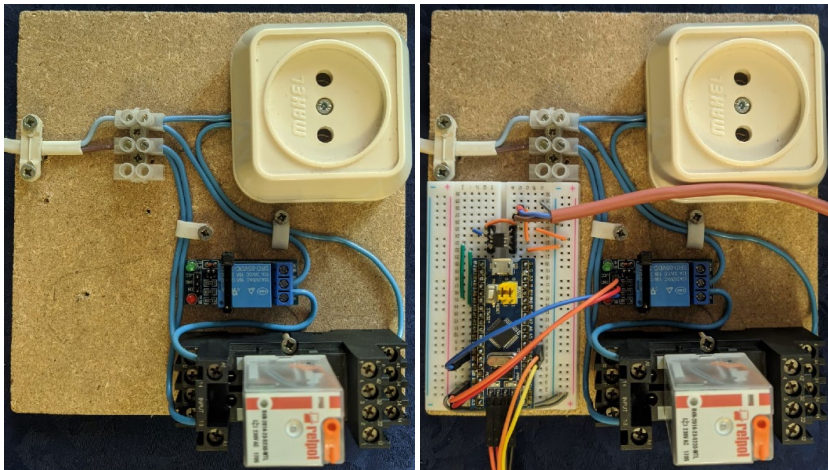


Рис. 5. Микроконтроллер с подключенными к нему драйвером линии и термодатчиком



а

б

а –отдельно, б – с подключенным микроконтроллером

Рис. 6. Релейный модуль

Для управления нагревательным элементом был создан релейный модуль, состоящих из двух реле, маломощной, управляемой микроконтроллером, и реле, непосредственно управляющая розеткой (рис. 6). В совою очередь, к розетке можно подключить какой угодно нагревательный элемент.

В качестве системы нагрева была взята настольная лампа, с закрепленным в ней термодатчиком. В качестве нагревательного элемента выступает лампа накаливания, а в качестве теплоносителя – нагретый воздух (рис. 7).



а

б

а – вид спереди, б – вид сбоку

Рис. 7. Макет системы нагрева

3. Испытание макета

Для испытания макета был создан простой алгоритм поддержания температуры. Этот алгоритм отключал нагрев при достижении поддерживаемого уровня и включал, когда температура падала ниже заданного на два градуса.

На рис. 8 представлен график, изменения температуры, при ее поддержании в пятьдесят градусов. Крестиками отмечены моменты выключения, а точками – моменты отключения.

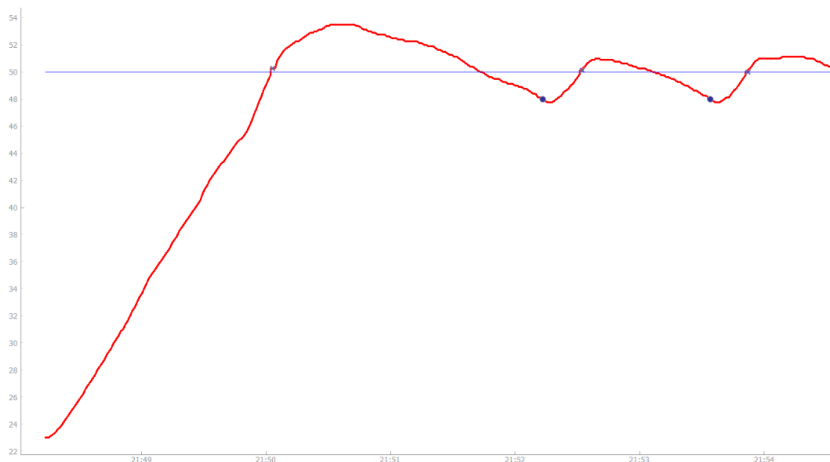


Рис. 8. График изменения температуры

Из графика видно, что после первого отключения температура продолжает расти и превышает заданный порог почти на четыре градуса. В то время как при последующих циклах включения и выключения, температура превышает его примерно на один градус.

Заключение

Проблема перегрева, возникающего в результате запоздалого выключения нагревателя особо остро стоит в областях, где такой перегрев может привести к непоправимому ущербу. Например, при сушке зерна сильное превышение температуры сужо воздуха может испортить зерно, что является серьезной проблемой. Кроме того, отопительные котлы обладают крайне высокой тепловой инерцией, температура после выключения горелки может возрасти еще на двадцать и более градусов и привести к необратимым нарушениям в системе, например, потеря герметичности трубопроводов.

Наличие проблемы было наглядно продемонстрировано испытанием созданного макета. Так же это испытание показала, что макет релевантен и может быть использован для тестирования адаптивных алгоритмов управления.

В дальнейшем планируется разработать один или несколько адаптивных алгоритмов. Рассматривается возможность замены дискретного управления по средствам реле на аналоговое с помощью диммеров. Также планируется создание новых макетов систем нагревания.

Список литературы

1. ПИД-регуляторы [Электронный ресурс]: электронный науч. журн. – Режим доступа : https://www.bookasutp.ru/Chapter5_1.aspx
2. Официальная страница STMicroelectronics - STM32F103C8 [Электронный ресурс]: база данных. – Режим доступа : <https://www.st.com/en/microcontrollers-microprocessors/stm32f103c8.html>
3. Официальная документация термодатчика DS18B20 [Электронный ресурс]: документация. – Режим доступа : <https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf>
4. Официальная документация драйвера линии One-wire DS2480B [Электронный ресурс]: документация. – Режим доступа : <https://datasheets.maximintegrated.com/en/ds/DS2480B.pdf>

Оценка масштабируемости протоколов маршрутизации на имитационной модели беспроводной ячеистой сети в среде симулятора NS3

А. Е. Онищук

Студент магистр

А. А. Епифанцев

АО «Концерн «Созвездие»

Введение

В настоящее время системы связи играют важную роль. Такие характеристики, как качество, скорость соединения между абонентами сети и доступность во многом определяют уровень развития общества. Беспроводные сети являются одним из самых перспективных направлений развития современных телекоммуникационных систем связи. Перспективы их использования связаны, во-первых, с заменой кабельной инфраструктуры на радиоэфир; во-вторых, с новыми возможностями коммуникаций между различными устройствами. При этом наряду с построением централизованных сетей, интерес представляет использование элементов децентрализации, которые присутствуют в ячеистых (mesh) сетях.

Mesh-сеть (WMN – Wireless Mesh Network) представляет собой ячеистую топологию, в которой различные устройства объединяются между собой в единую сеть [1, 2]. В этой сети отсутствует единый центральный сервер. WMN обеспечивает лучшую мобильность, более низкую стоимость развертывания, простое расширение сети, а также надежные соединения. Именно эта структура обеспечивает одно из основных преимуществ ячеистых сетей – адаптивную топологию, которая способна перестраиваться, если один из узлов сети недоступен или перегружен.

Информационные сети, организованные по топологии mesh [3], получили за последние полтора-два года большое признание. Масштабы проектов выросли до тысяч точек доступа и десятков тысяч пользователей. Ячеистые сети представляют наиболее интересные решения, интегрирующие различные сетевые и радиотехнологии, и потому в полной мере отвечают все более растущим требованиям

абонентов (мобильность, качество обслуживания, безопасность). Данная работа посвящена сравнению работы протоколов маршрутизации AODV и HWMP в беспроводной ячеистой сети с различным количеством узлов [4]. Также оценивается масштабируемость протоколов с учетом используемой ими метрики, поэтому помимо обычных протоколов AODV и HWMP для оценки также берется протокол HWMP с метрикой числа ретрансляций.

1. Среда моделирования и выбор протоколов маршрутизации

В сетевом симуляторе NS3 была разработана программная реализация имитационной модели беспроводной ячеистой сети. В качестве протоколов маршрутизации были выбраны протоколы HWMP, который является стандартным протоколом маршрутизации по умолчанию для стандарта 802.11s, и AODV. Цель работы - сравнить и оценить их производительность с точки зрения масштабируемости (при изменении числа узлов и соединений).

2. Оценка метрики маршрутизации

Метрика – это число переходов до места назначения. Она используется для определения наилучшего маршрута. Важным аспектом, который необходимо учитывать, является то, что HWMP использует метрику радиосоединения (метрика эфирного времени ALM – Airtime Link Metric), в то время как AODV использует метрику числа ретрансляций. Следует отметить, что используя в качестве метрики число ретрансляций, не учитывается текущая пропускная способность отдельных соединений в сети, поэтому при выборе маршрута для передачи информации нагрузка в сети не учитывается.

Проведем тестирование для сравнения AODV и HWMP с использованием двух разных постоянных канальных скоростей передачи (при скорости 6 Мбит/с и 12 Мбит/с) на модели беспроводной ячеистой сети со следующим основными характеристиками:

- длительность моделирования = 180 с;
- топология: квадратно-гнездовое распределение в области 600×600 м;
- количество узлов = 25;
- количество соединений = 5-20;
- длительность каждого соединения = экспоненциальная (средняя = 25 с);
- скорость передачи данных для каждого соединения = 200 Кбит/с;
- размер пакета = 1024 байта.

На рис. 1 и 2 показаны результаты тестирования. По оси X установлено количество соединений, а по оси Y – пропускная способность и доля доставленных пакетов соответственно. Пропускная способность при использовании протокола AODV значительно меньше, чем при использовании протокола HWMP. В среднем для протокола AODV со скоростью 12 Мбит/с доля доставленных пакетов меньше на 35% и на 38% меньше пропускная способность по сравнению с его пропускной способностью на скорости 6 Мбит/с, тогда как для HWMP данные показатели меньше только на 10% и 15% соответственно.

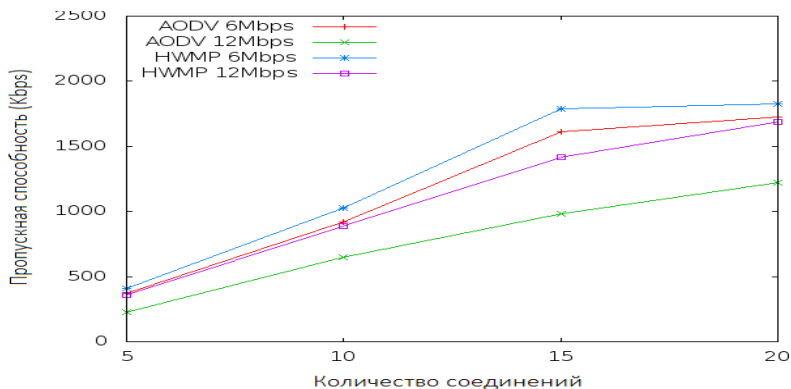


Рис. 1. Пропускная способность

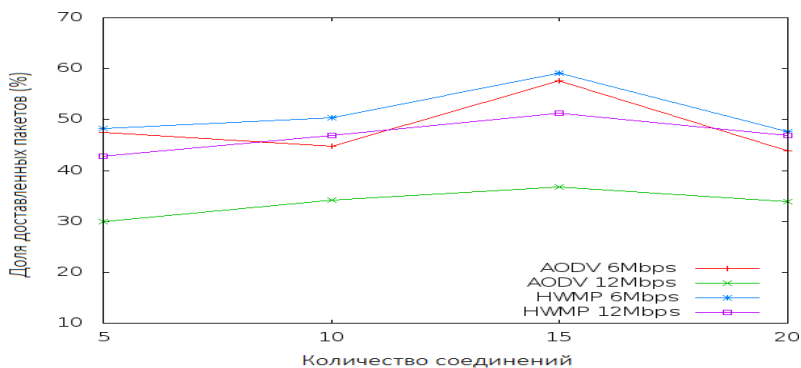


Рис. 2. Доля доставленных пакетов

3. Анализ масштабируемости протоколов

В следующих тестированиях анализируется масштабируемость протоколов HWMP, AODV и HWMP с метрикой числа ретрансляций. Будем использовать топологию, изображенную на рис.3.

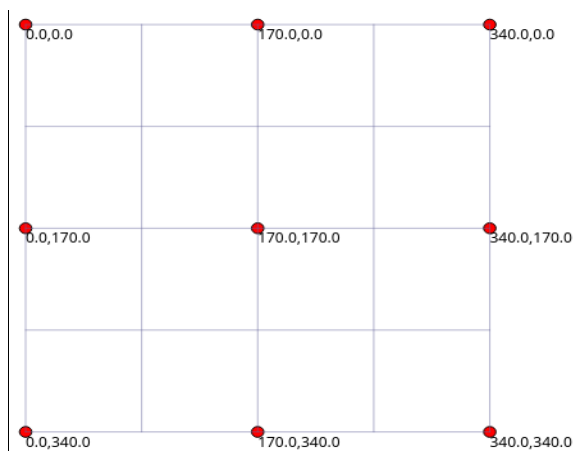


Рис. 3. Пример топологии сети 3x3 (9 узлов)

Данная топология более надежна для сравнения результатов, чем при случайном распределении узлов в определенном месте. Используя случайное распределение, мы имеем дополнительную переменную, случайное положение узлов, что затрудняет сравнение результатов.

Рассмотренные топологии сети: 3x3 (9 узлов), 4x4 (16 узлов), 5x5 (25 узлов) и 6x6 (36 узлов) с расстоянием между узлами в 170 метров.

Количество случайных соединений, установленных между узлами, совпадает с количеством узлов.

Размер пакета – 1024 байта, скорости изменяются с 50 Кбит/с до 400 Кбит/с, поэтому в определенный момент сеть будет загружена достаточно, чтобы увидеть, как каждый протокол справляется с перегрузкой отдельных соединений.

Причиной существенных различий между протоколами может быть метрика, а также время на построение маршрутов в сети. Чтобы узнать, что определяет разницу между производительностью HWMP и AODV, был протестирован третий протокол: HWMP с использованием метрики числа ретрансляций вместо метрики эфирного времени.

В результате тестирования сценария были построены графики. По оси X была установлена скорость передачи пакетов данных, а по оси Y – пропускная способность и доля доставленных пакетов данных. После

проверки полученных результатов для топологий с различным количеством узлов было установлено, что в небольших сетях доля доставленных пакетов и пропускная способность лучше с использованием протокола HWMP, в то время как в больших сетях протокол AODV работает лучше. Так, например, в сети 3x3 пропускная способность протоколов AODV и HWMP с метрикой числа ретрансляций в среднем на 7% и на 3% меньше соответственно, чем у обычного HWMP. Тогда как в сети 6x6 пропускная способность HWMP с метрикой числа ретрансляций и обычного HWMP на 6% и 7% меньше соответственно, чем у AODV. Это связано с тем, что метрика эфирного времени использует менее эффективный маршрут, что обусловлено дополнительной нагрузкой на канал служебными сообщениями протокола маршрутизации. Дополнительная нагрузка на канал служебными сообщениями приводит к необходимости перестроения маршрута с целью обеспечения требуемого качества обслуживания, которое может ухудшаться при недостаточной пропускной способности соединений и коллизиях. Это также объясняет, что при высокой скорости передачи данных в небольшой сети HWMP менее эффективен, чем протокол AODV. При высокой нагрузке в сети частота потерь пакетов увеличивается, а метрика ALM не позволяет строить оптимальные маршруты и, как следствие, обеспечить максимально возможную пропускную способность в сети.

Кроме того, во всех случаях производительность HWMP с метрикой числа ретрансляций находится между AODV и HWMP, что помогает нам доказать, что одной из основных причин разницы в пропускной способности и доле доставленных пакетов при увеличении узлов и соединений является метрика маршрута.

В среднем HWMP превосходит AODV в сети 3x3 и 4x4, тогда как в сети 6x6 протокол AODV лучше (в сети 5x5 они показывают аналогичную пропускную способность и доли доставленных пакетов). Таким образом, AODV работает лучше, чем HWMP с точки зрения доли доставленных пакетов и пропускной способности, когда число узлов и соединений выше. Однако это различие в худшем случае составляет 5% при сравнении долей доставленных пакетов. Графики пропускной способности и доли доставленных пакетов для сетей 3x3 и 6x6 представлены на рис. 4 – рис. 7.

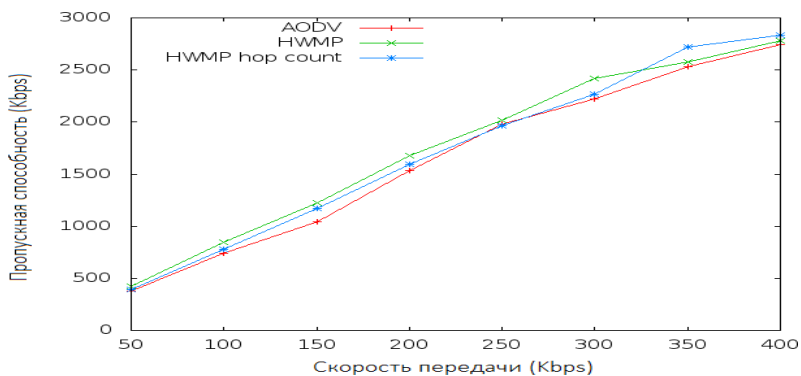


Рис. 4. Пропускная способность в сети 3x3

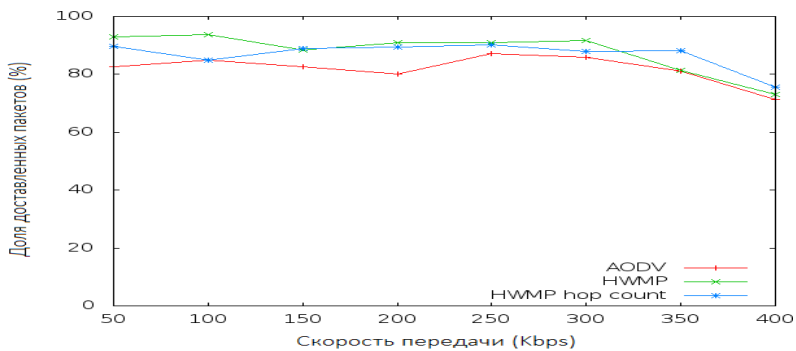


Рис. 5. Доля доставленных пакетов в сети 3x3

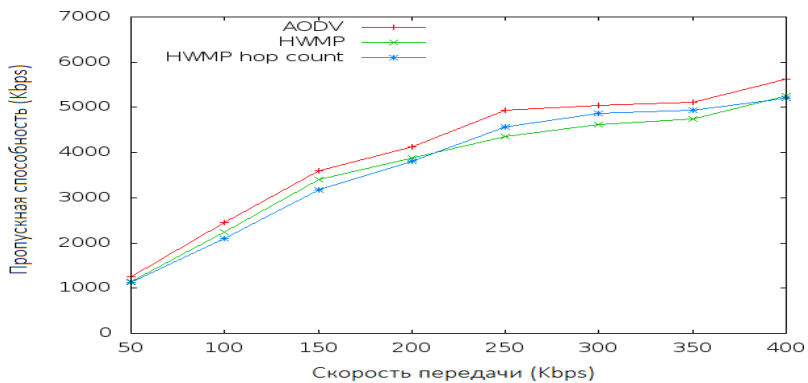


Рис. 6. Пропускная способность в сети 6x6

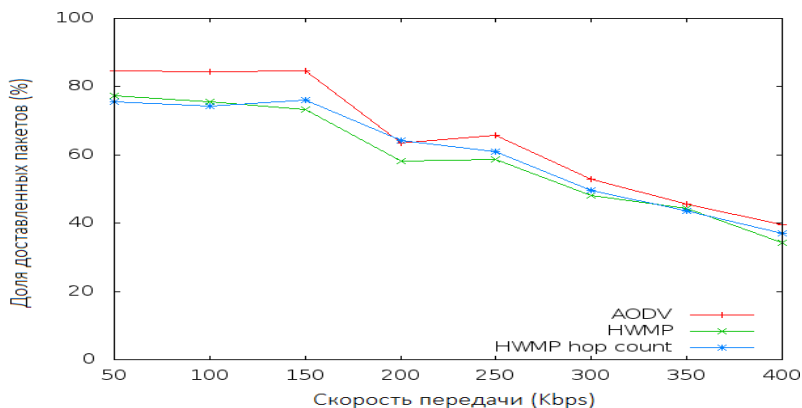


Рис. 7. Доля доставленных пакетов в сети 6х6

Заключение

Данная статья посвящена сравнению работы протоколов маршрутизации в сетях с различным количеством узлов. Описаны модели сетей с протоколами маршрутизации HWMP, AODV и HWMP с метрикой числа ретрансляций, построены графики, позволяющие оценить работу протоколов в сети. Было установлено, что в небольших сетях, к примеру, в топологии 3х3 с 9 узлами, доля доставленных пакетов данных и пропускная способность лучше с использованием протокола HWMP, в то время как в больших сетях, в топологии 6х6 с 36 узлами, данные показатели лучше у протокола AODV. Аналогичные характеристики для HWMP с метрикой числа ретрансляций принимают средние значения между AODV и HWMP с метрикой ALM. Также было установлено, что при увеличении канальной скорости передачи с 6 Мбит/с до 12 Мбит/с при использовании в сети AODV доля доставленных пакетов и пропускная способность существенно уменьшаются, что свидетельствует о необходимости построения маршрутов с учетом текущего значения пропускной способности соединений, реализованного в протоколе HWMP.

Список литературы

1. Абрамов, А. А. Уязвимости протокола маршрутизации в mesh-сети стандарта 802.11s / А. А. Абрамов, М. Е. Бурлаков // Вестник ПНИПУ, Самарский национальный исследовательский университет им. С.П. Королева. – 2017. – С. 59-72.

2. Осипов, И. Е. Mesh-сети: технологии, приложения, оборудование / И.Е. Осипов // Технологии и средства связи. – 2008. – № 4. – С. 38-45.

3. Mesh-сети стандарта IEEE 802.11s: протоколы маршрутизации / В. Вишневский, Д. Лаконцев, [и др.] // Первая миля. — 2008. – № 2-3. – С. 26-31.

4. Пролетарский, А. В. Беспроводные сети Wi-Fi / А. В. Пролетарский, И. В. Баскаков, Д. Н. Чирков // М. : БИНОМ; Лаборатория знаний, 2007. – 178 с.

Использование методов компьютерного зрения и машинного обучения для анализа изображений спутниковой и аэрофотосъемки в интересах мониторинга изменений на местности

Р. Р. Отырба

Студент магистрант

М. А. Дрюченко

Доцент

Введение

Обнаружение изменений – это аналитический процесс, целью которого является обнаружения новых или исчезнувших объектов на изображениях, зарегистрированных в разные моменты времени для одной и той же целевой области. Обнаружение изменений полезно и важно особенно для автоматического обновления городских карт, анализа после катастрофических событий, обнаружения незаконных построек и незаконной вырубki, анализа и мониторинга развития городских, пригородных территорий, акваторий и изменений ландшафта [1].

Классический ручной поиск таких изменений выполняется с использованием инструментальных методов и с непосредственным участием оператора-интерпретатора изображений. При частых изменениях обнаружение таких изменений вручную будет трудоемкой и неточной задачей.

Научных методологий, касающуюся обнаружения изменений на местности, существует не великое множество, особенно связанной с городской местностью. В основном, эти методы имеют значительные ограничения, они слишком сильно полагаются на информацию, предоставляемую с изображениями, и поэтому их нельзя применить к любому изображению, которое не обладает какой-либо информацией, такой как геопространственная информация. Более того, они также ограничены геометрическими аспектами, такими как: одинаковой ориентацией изображения, масштабом, высотой, координатами и одинаковой калибровкой камеры. Дополнительно, исследования, направленные на обнаружение изменений в городской и пригородной

среде ограничены конкретными топографическими объектами, чаще всего зданиями и не учитывают другие значимые объекты для мониторинга.

Поскольку на практике, чаще всего снимки не ограничены этими аспектами, по этим причинам обнаружение изменений является довольно важной темой исследования и является привлекательной как для частных, так и государственных учреждений.

Целью данного исследования является разработка системы для мониторинга изменений на местности, используя методы машинного обучения и компьютерного зрения.

Научная новизна работы состоит в том, что она посвящена теме фактически не рассматриваемой ранее комплексно отечественными исследователями и предлагает расширенные возможности решения рассматриваемой задачи по отношению к предложенным зарубежным подходам.

1. Общий алгоритм разработанной системы

1. На вход подаётся свежий снимок, с возможными изменениями.
2. Выполняется нарезка исходного изображения на блоки.
3. Производится циклическая сегментация нарезанных блоков обученной нейронной сетью.
4. Результирующее сегментированное изображение подвергается пост-обработке (очистка от шума, упрощение геометрии и т.п.).
5. Подбирается давний снимок и его соответствующая бинарная маска с целевой областью из предварительно готового набора данных, используя GPS метаданные. Далее снимок и маска сшивается с ближайшими по координатам снимками.
6. Выполняется коррекция перспективы нового снимка по отношению к полученному старому снимку, посредством вычисления и использования матрицы гомографии.
7. Выполняется фильтрация от шумов и вычисление разницы двух масок и отображение изменений на изображении.

2. Подготовка набора данных для обучения нейронных сетей

Для реализации поставленной цели было осуществлено создание собственного набора данных, посредством ручной разметки снимков высокого разрешения (6792x4800) Google Earth Pro в онлайн сервисе CVAT.

Для обучения сети была произведена нарезка снимков на фрагменты размером 512x512 с перекрытием на 100 пикселей для учёта контекста.

Итоговый состав набора данных следующий:

- здания и дороги: 10 снимков (~2000 размером 512x512);
 - деревья и вода: 5 снимков (~1000 размером 512x512).
- Примеры таких размеченных данных представлены на рис.1.

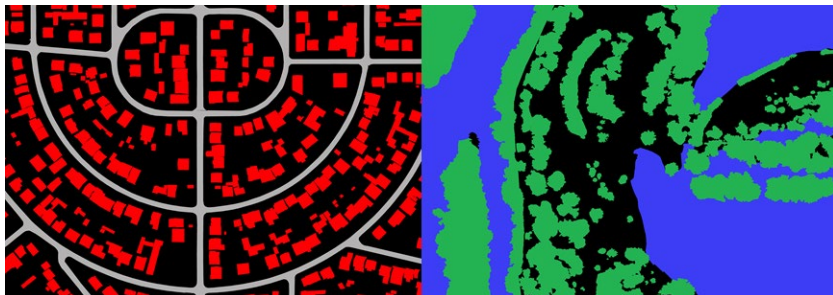


Рис. 1. Примеры размеченных снимков

Для лучшего качества обучения, была применена аугментация, которая происходит на лету во время обучения:

- горизонтальные и вертикальные повороты;
- случайные повороты на 90 градусов;
- масштабирование с случайными поворотами;
- уменьшение контрастности (от 5% до 20%);
- изменение цветового тона и насыщенности (до 20 %).

Так как снимки могут быть сняты в разную погоду и время суток, было использовано уменьшение контрастности и изменение цветового тона и насыщенности.

К размеченным данным было применено One-Hot кодирование, и все данные были разделены на обучающие и проверочные выборки (90% и 10% данных соответственно).

3. Обучение нейронных сетей

В качестве моделей нейронных сетей, были рассмотрены наиболее современные модели DeepLabV3+ и DANet с различными кодирующими сетями среднего и сильно глубокого размера DenseNet201, ResNet101v2, ResNet152v2, EfficientNetb5, EfficientNetb7.

В качестве оптимизатора был выбран Момент Нестерова, накапливающий импульс и с идей заглядывания вперед по вектору обновления параметров сети. В отличие от адаптивных оптимизаторов, Момент Нестерова не так сильно переобучается, хорош в обобщающих способностях и поиске плоских локальных минимумов высокого качества, однако требуют тщательной настройки скорости обучения [2].

Дополнительно, чтобы избежать увязания в процессе обучения, была реализована схема отжига скорости обучения (формула 1), график которой представлен на рис. 2, где максимальная скорость равна 0.2, а минимальная 0.

$$\eta_t = \eta_{\min}^i + \frac{1}{2}(\eta_{\max}^i - \eta_{\min}^i) \cdot (1 + \cos(\frac{T_{cur}}{T_i} \pi)); \quad (1)$$

где η_{\min} – минимальная скорость обучения, η_{\max} – максимальная скорость обучения, T_{cur} – количество выполненных эпох.

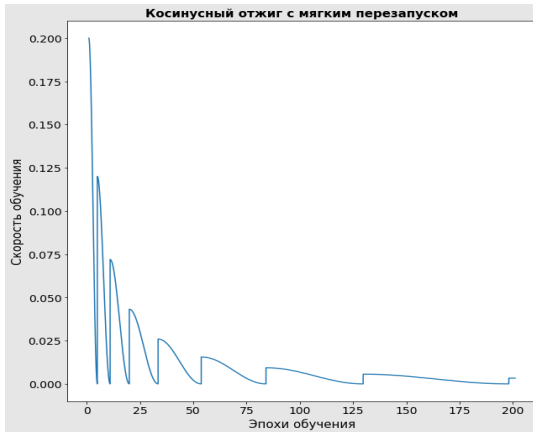


Рис. 2. Косинусный отжиг с мягким перезапуском

В качестве метрики используется средняя взвешенная степень пересечения изображений:

$$IoU = \frac{1}{N} \cdot \sum_{c=0}^C \omega(c) \cdot \frac{I(c)}{U(c)} \quad (2)$$

где $w(c)$ – весовой коэффициент класса, $I(c)$ – область пересечения, $U(c)$ – область объединения.

В качестве функции ошибки была использована комбинация взвешенной функции Дайса и Фокальной ошибки (формула 3), где функция Дайса – степень соответствия между предсказанием и истиной (формула 4), а фокальная ошибка – это модификация кросс-энтропии, занижающий вклад преобладающего класса и фокусируется на более сложных примерах (формула 5).

$$L = L_{Dice} + \lambda L_{Focal} \quad (3)$$

$$L_{Dice} = 1 - \frac{1}{N} \cdot \sum_{c=0}^C \omega(c) \frac{(1 + \beta) + TP_p(c)}{(1 + \beta) \cdot TP_p(c) + \beta^2 \cdot FN_p(c) + FP_p(c)}, \quad (4)$$

$$L_{Focal} = -\frac{1}{N} \sum_{c=0}^C \sum_{n=1}^N g_n(c) \cdot (1 - p_n(c))^2 \cdot \log(p_n(c)), \quad (5)$$

где λ – компромисс между функциями (0.5), TP – положительный класс распознанный как положительный, FN – отрицательный класс распознанный как отрицательный, FP – отрицательный класс распознанный как положительный.

Результаты обучений нейронных сетей на проверочных данных после 200 эпох обучения представлены в табл. 1. В итоге была выбрана модель DeepLabV3+ с кодирующей сетью EfficientNetb7.

Таблица 1

Результаты обучения нейронных сетей

<i>Здания и дороги</i>			
<i>Модель</i>	<i>Кодирующая сеть</i>	<i>Loss</i>	<i>IoU</i>
DeepLabV3+	DenseNet201	0.28356	0.7832
DeepLabV3+	ResNet101v2	0.33176	0.7006
DeepLabV3+	ResNet152v2	0.31824	0.7260
DeepLabV3+	EfficientNetb5	0.29112	0.7779
DeepLabV3+	EfficientNetb7	0.27049	0.7919
DANet	DenseNet201	0.29981	0.7291
DANet	ResNet101v2	0.35021	0.6339
DANet	ResNet152v2	0.33669	0.6593
DANet	EfficientNetb5	0.29809	0.7188
DANet	EfficientNetb7	0.29248	0.7328
<i>Деревья и вода</i>			
DeepLabV3+	EfficientNetb7	0.5537	0.7371

4. Пост-обработка

Результаты сегментирующих сетей необходимо обработать, так как они содержат шум и нечёткость. Для этого были рассмотрены следующие алгоритмы рафинирования результирующей сегментации:

- плотные условные случайные поля (Dense CRF);
- сегментационная модель рафинирования CascadePSP [3].

Поскольку алгоритмы рафинирования несмотря на хорошее уточнение сегментации, могут оставлять небольшой шум была дополнительно реализована следующая ручная обработка:

1. Метод пороговой обработки (Метод Оцу).
2. Медианная фильтрация.
3. Поиск контуров и очистка от лишних объектов.
4. Морфологические операции (реконструкция, открытие, закрытие).
5. Модифицированный алгоритм упрощения геометрии Дугласа-Пекера, сохраняющий топологию объекта.
6. Медианная фильтрация в цикле.

Сравнение результатов до и после обработки представлены в табл. 2, а визуализация на рис. 3. Как видно, пост-обработка с CascadePSP даёт наилучший прирост к общему качеству сегментации.

Таблица 2

Сравнение результатов до обработки и после

<i>Модель</i>	<i>Loss</i>	<i>IoU</i>	<i>Время</i>
DeepLabV3+	0.2701	0.7846	21 сек.
DeepLabV3+ & CRF	0.2596	0.8147	97 сек.
DeepLabV3+ & CRF с обработкой	0.2505	0.8215	113 сек.
DeepLabV3+ & CascadePSP	0.2411	0.8307	221 сек.
DeepLabV3+ & CascadePSP с обработкой	0.2395	0.8332	230 сек.

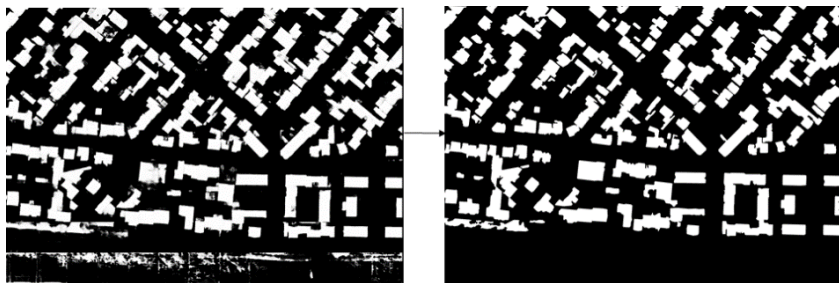


Рис. 3. Результаты до и после пост-обработки

5. Получение и обработка давнего снимка целевой области

Для получения ранее отснятого снимка целевой области используются GPS метаданные снимка, с помощью которых находится

наиболее кратчайшее расстояние до координаты целевой области, посредством вычисления геодезического расстояния, в которой используется эллипсоидальная модель Земли.

Когда ближайший давний снимок найден, происходит сшивка вокруг его области, в том числе сшивка соответствующих масок. Это необходимо, так как свежий снимок может быть снят по-иному. На рис. 4 квадратом представлены границы возможного свежего снимка, где к нему нашёлся ближайший давний снимок. Далее происходит циклический процесс сшивки с помощью алгоритма масштабнo-инвариантной трансформации признаков AKAZE, о котором речь пойдёт далее.

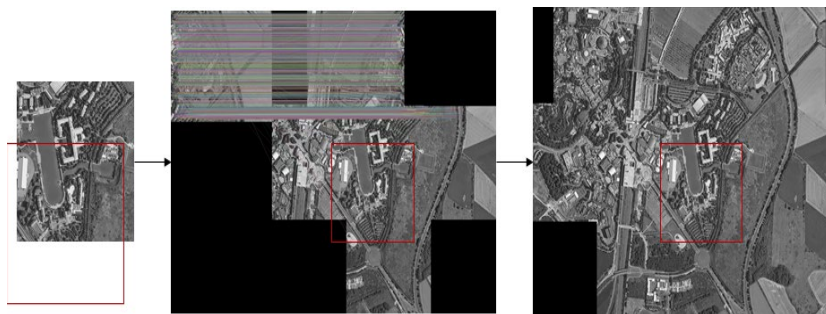


Рис. 4. Процесс сшивки давнего снимка

6. Коррекция перспективы

Для коррекции перспективы производится расчёт матрицы гомографии, которая в процессе расчёта использует лучшие совпадения по ключевым признакам. Для их нахождения рассматривались:

1. Алгоритмы выявления локальных признаков изображения: SIFT, KAZE, AKAZE, ORB.

2. Методы сопоставления признаков. Рассматривались два варианта: Brute-Force с перекрёстной проверкой и KNN Brute-Force с тестом соотношения Дэвида Лоу [4].

3. Методы вычисления расстояния между дескрипторами: расстояние Хэмминга или L2 норма.

4. Методы вычисления гомографии: RANSAC, LMEDS, RHO.

Результаты сравнения этих алгоритмов представлены в табл. 3.

Сравнение методов выявления локальных признаков с коррекцией перспективы

Метод	Метод оценки гомографии	Время работы	IoU		
			Масштаб (x1.4)	Угол 150	Масштаб + угол
AKAZE	LMEDS	7.2 сек.	0.9261	0.9355	0.9215
ORB	RANSAC	0.86 мс.	0.4130	0.9340	0.0262
KAZE	RANSAC	55.8 сек.	0.9262	0.9354	0.9214
SIFT	RHO	15.4 сек.	0.9256	0.9354	0.9316

Далее на рис. 5 и 6 представлено тестирование такой коррекции и выявление изменений между якобы давним и новым снимком.



Рис. 5. Обработанная маска ранее отснятого и нового снимка



Рис. 6. Коррекция перспективы и обнаружение изменений

Заключение

Данная статья посвящена разработке системы для мониторинга изменений на местности. В результате был подготовлен собственный набор данных, подготовлены модели нейронных сетей, проведена пост-обработка, организовано получение и обработка ранее отснятого снимка и процесс коррекции перспективы с итоговым обнаружением изменений.

Список литературы

1. Zhang, C. Detecting large-scale urban land cover changes from very high resolution remote sensing images using CNN-based classification / C. Zhang, S. Wei, S. Ji // ISPRS International Journal of Geo-Information – 2019. – Т. 8. – № 4. – С. 189.
2. Wilson, A. The marginal value of adaptive gradient methods in machine learning / A. Wilson, R. Roelofs, M. Stern, N Srebro, B Recht // In Advances in Neural Information Processing Systems – 2017. – С. 4148-4158.
3. Ho, C. CascadePSP: Toward Class-Agnostic and Very High-Resolution Segmentation via Global and Local Refinement / C., Ho, C. Jihoon, T. Yu-Wing, T. Chi-Keung // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition – 2020. – С. 8890-8899.
4. Lowe, D. Distinctive Image Features from Scale-Invariant Keypoints / D. Lowe // International Journal of Computer Vision. – 2004. – Т. 60. – № 2. – С. 91–110.

Реализация и исследование нейросетевого алгоритма переноса стиля изображения

А. А. Панькова

Студент магистрант

Е. Ю. Митрофанова

Доцент

Введение

Использование глубоких сверточных нейронных сетей (ГСН) в задачах компьютерного зрения дало старт для многочисленных исследований и экспериментов, связанных с обработкой изображений нейронными сетями. Одним из самых популярных приложений ГСН стала классификация объектов на изображениях. В процессе распознавания нейросеть на каждом уровне (слое) строит карты признаков для элемента изображения. Данная особенность, а именно извлечение различных признаков объекта, не зависящих от фонового шума и других искажений, позволила применить ГСН для ряда других задач.

Представляет интерес задача переноса стиля изображения, где заранее обученная нейронная сеть используется в качестве универсального комплексного экстрактора признаков объекта на изображении [1]. Взяв стиль с одного изображения, контент с другого изображения и скомбинировав их, мы получим совершенно новое, нигде до этого не существующее, изображение. К сожалению, получившееся изображение скорее всего будет иметь достаточно много шума. В этой работе рассматривается один из способов устранения шума на сгенерированном в процессе переноса стиля изображении.

1. Алгоритм переноса стиля

При обработке изображения ГНС на выходе каждого слоя сети получается некоторое уникальное представление об объектах и их расположении на входном изображении. Проведя реконструкцию из любого слоя нейросети, можно увидеть, какая именно информация о изображении получена на текущем слое. На нижних слоях ГНС содержится информация о текстуре и цвете изображения, а на верхних –

об объектах и их расположении. Соответственно, выходы нижних слоев сети будем называть «стилем», а выходы верхних слоев – «контентом».

Поскольку контент изображения и его стиль не могут быть полностью разделены, при создании изображения, которое сочетает в себе контент одного изображения со стилем другого, практически никогда не получается изображение, идеально соответствующее обоим ограничениям одновременно. Можно воспользоваться функцией потерь, которую нужно минимизировать при генерации изображения. Данная функция применяется как для контента, так и для стиля. Используя функцию потерь, получаем возможность плавно регулировать акцент на восстановлении контента или же стиля.

Сильный акцент на стиле приведет к тому, что новое изображение будет отлично отображать текстурную версию изображения-источника стиля, но едва ли показывать содержание изображения-источника контента. Выделяя в большей мере контент, можно четко увидеть изображенные объекты, но текстура и цвета изображения-источника стиля будут практически потеряны.

Пусть \bar{p} – оригинальное изображение и \bar{x} – сгенерированное изображение, P^l и F^l – их карты признаков на слое l , тогда функция потери контента будет определена так:

$$L_{\text{content}}(\bar{p}, \bar{x}, l) = \frac{1}{2} \sum_{i,j} (F_{ij}^l - P_{ij}^l)^2. \quad (1)$$

Для вычисления функции потери стиля, необходимо сначала вычислить корреляционную матрицу Грамма. Определим N_l – количество карт признаков размера M_l на слое l , G_{ij}^l – корреляция между разными цветовыми каналами карт признаков i и j одного слоя l :

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l. \quad (2)$$

Пусть \bar{a} – оригинальное изображение и \bar{x} – сгенерированное изображение, A^l и G^l – матрицы Грамма карт признаков на слое l , тогда вклад этого слоя в функцию потерь стиля будет представлять собой:

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2. \quad (3)$$

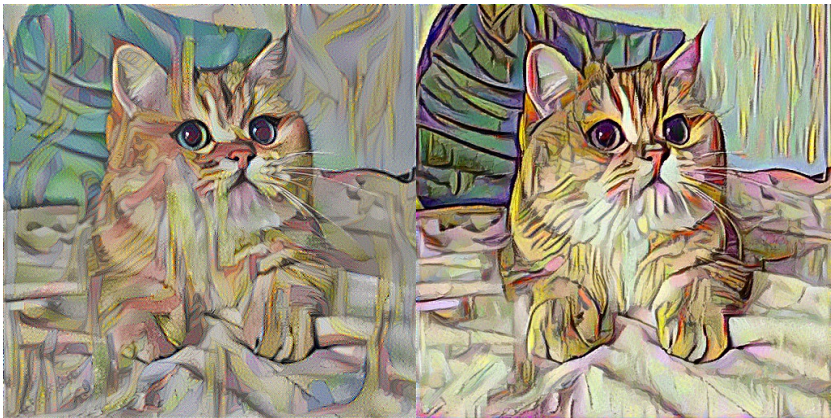
Кроме того, для каждого слоя зададим свой вес w_l , тогда функция потерь стиля будет определена следующим образом:

$$L_{\text{style}}(\bar{a}, \bar{x}) = \sum_{l=0}^L w_l E_l. \quad (4)$$

Для получения изображения, совмещающего в себе объекты контента и текстуру стиля, необходимо минимизировать различия между изображением белого шума и представлением контента с одного из верхних слоев ГНС и представлением стиля группы нижних слоев ГНС. Общая функция потерь представлена на формуле ниже. В формуле \bar{p} – изображение контента, \bar{a} – изображение стиля, \bar{x} – инициализирующее изображение (обычно – белый шум), α, β – коэффициенты влияния контента и стиля соответственно.

$$L_{\text{total}}(\bar{p}, \bar{a}, \bar{x}) = \alpha L_{\text{content}}(\bar{p}, \bar{x}) + \beta L_{\text{style}}(\bar{a}, \bar{x}). \quad (5)$$

Ниже представлены изображения, полученные с применением этих формул в алгоритме переноса стиля изображения. В первом случае в качестве инициализирующего изображения используется белый шум, во втором – копия контента.



а

б

а – инициализация белым шумом, б – инициализация контентом

Рис. 1. Результат генерации изображения алгоритмом переноса стиля

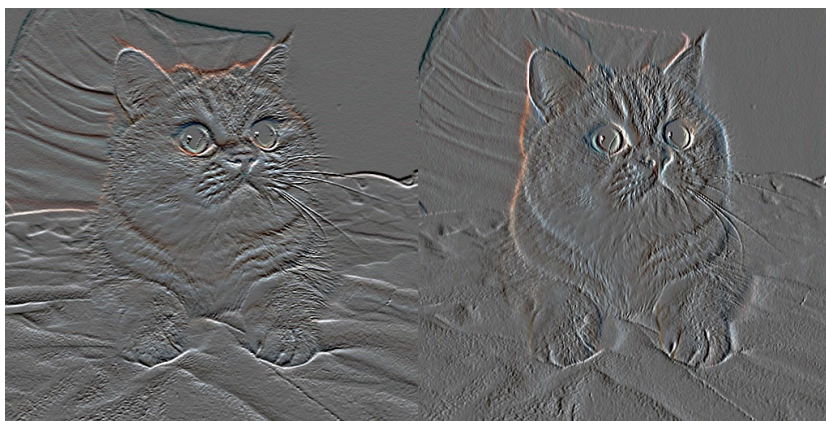
Как можно видеть, оба изображения действительно комбинируют в себе стиль одного изображения с контентом другого, однако оба получили много высокочастотных артефактов, что является существенным недостатком базового алгоритма переноса стиля.

2. Устранение шума

Оба изображения, полученные с помощью алгоритма переноса стиля, являются достаточно зашумленными. Однако, если сравнивать изображения, то очевидно, что изображение на рис. 1 (б) обладает гораздо меньшим уровнем шума. Тем не менее, хотелось бы улучшить оба изображения. В этом может помочь добавление к общей функции потерь (5) такого параметра, как «полная вариация».

Концепция устранения шума методом минимизации полной вариации была разработана в 1992 году, её авторами являются Рудин, Ошер, Фатеми [2]. Идея алгоритма состоит в том, что изображение с большим количеством шума обладает высоким уровнем вариации, тогда как в изображении без шума, вариация должна быть минимальной. В модели ROF (Rudin-Osher-Fatemi) уменьшение общей вариации сигнала (при условии, что он близок к исходному сигналу) удаляет нежелательные детали, сохраняя при этом важные геометрические элементы, такие как края.

Рассмотрим оригинальное и сгенерированные изображения, выделив все горизонтальные и вертикальные высокочастотные составляющие [3-4].

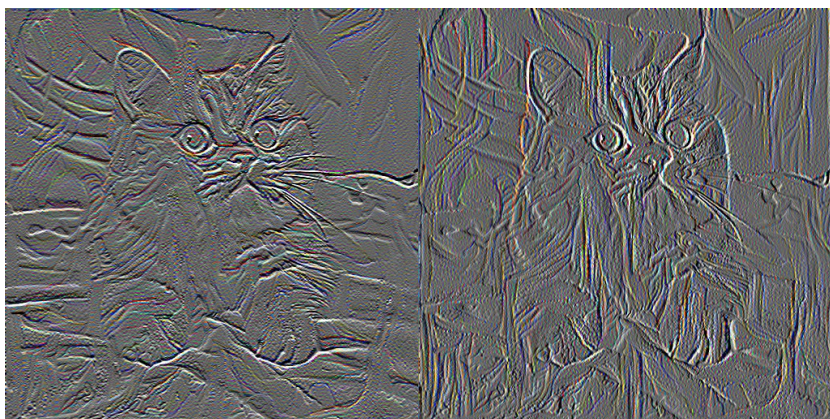


а

б

а – горизонтальные части, б – вертикальные части

Рис. 2. Оригинальное изображение с выделенными оператором Собеля границами

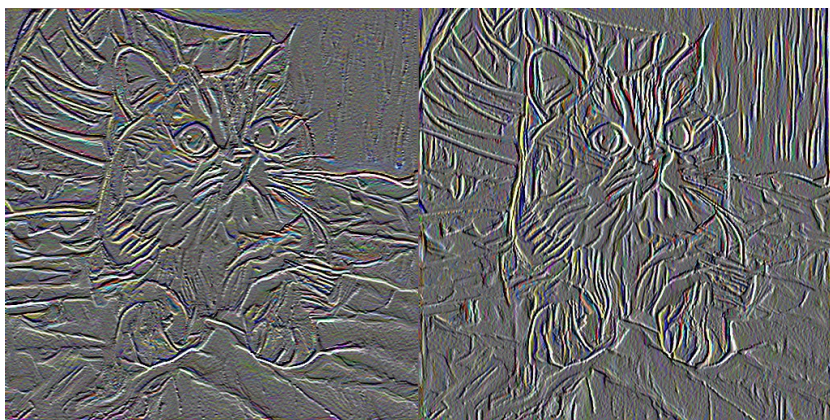


а

б

а – горизонтальные части, б – вертикальные части

Рис. 3. Сгенерированное (инициализация белым шумом) изображение с выделенными оператором Собеля границами



а

б

а – горизонтальные части, б – вертикальные части

Рис. 4. Сгенерированное (инициализация контентом) изображение с выделенными оператором Собеля границами

Таким образом, на сгенерированных изображениях в большом количестве присутствуют высокочастотные артефакты. Для их

устранения могут использоваться фильтры детектора края, известные как фильтры высоких частот, которые сглаживают изображение, пропуская только значения выше определенного порога. Уменьшим высокочастотные артефакты и применим явную регуляризацию к компонентам с высокими частотными значениями. Вычислим полную вариацию, а затем включим в оптимизационную функцию потерь данный параметр, задав также коэффициент потери вариации. Полная формула представлена ниже (в формуле (6) $y_{i,j}$ – пиксель изображения):

$$L_{\text{totalVariation}}(y) = \sum_{i,j} |y_{i+1,j} - y_{i,j}| + |y_{i,j+1} - y_{i,j}|; \quad (6)$$

$$L_{\text{total}}(\bar{p}, \bar{a}, \bar{x}) = \alpha L_{\text{content}}(\bar{p}, \bar{x}) + \beta L_{\text{style}}(\bar{a}, \bar{x}) + \gamma L_{\text{totalVariation}}. \quad (7)$$

Сравним получившиеся после добавления еще одного параметра изображения. Очевидно, что шума стало гораздо меньше, при этом контуры объектов достаточно сохранены.



а

б

а – инициализация белым шумом, б – инициализация контентом

Рис. 5. Полученные после устранения шума изображения

Выводы

Применение ГНС в обработке изображений до сих пор остается огромным пространством для исследований. Алгоритмы совершенствуются и дорабатываются, позволяя получать всё лучшие результаты. Рассмотренный классический способ устранения шумов на изображении, сгенерированном с помощью базового алгоритма переноса стиля изображения показал, что результаты, полученные в

процессе работы нейросети вполне могут быть улучшены стандартными математическими способами. Добавление в общую функцию потерь параметра, отвечающего за полную вариацию изображения, позволило значительно уменьшить высокочастотные артефакты на полученных изображениях, существенно улучшив качество последних.

Список источников

1. Gatys, L. A. A Neural Algorithm of Artistic Style / L. A. Gatys, S. E. Alexander, B. Matthias. – 2015. – С. 17.
2. Rudin, L. I. Nonlinear total variation based noise removal algorithms / L. I. Rudin, S. Osher, E. Fatemi // *Physica D: Nonlinear Phenomena*. – 1992. – №60. – С. 259-268.
3. Sobel I. A 3x3 Isotropic Gradient Operator for Image Processing / I. Sobel, G Feldman // *The Stanford Artificial Intelligence Laboratory*. – 1968.
4. Дуда Р. Распознавание образов и анализ сцен // Р. Дуда, П. Харт. – М. : Мир, 1976. – С. 511.

Разработка робота для построения карты местности

А. А. Папина

Студент магистр

М. Г. Матвеев

Профессор

Введение

Чтобы определить положение робота в окружающем его пространстве, необходимо решить две основные задачи: первая – построение карты местности (отображение), вторая – нахождение (локализация) робота в пространстве.

В рамках первой задачи необходимо определить точное положение мобильного робота, чтобы производить построение карты, основываясь на данных получаемых от датчиков. Тогда получается, что сначала необходимо определить положение робота, то есть решить задачу локализации.

Как же определить положение робота? Можно использовать одометры, но точность такого метода невелика, можно использовать GPS, но, к сожалению, такой метод не является возможным в некоторых ситуациях. Поэтому необходимо использовать карты для определения положения робота. Но, так как окружающее робота пространство неизвестно, то необходимо разработать алгоритм, с помощью которого робот будет одновременно определять своё местоположения, траекторию движения, и по мере передвижения строить карту окружающего пространства.

1. Метод SLAM

Для решения поставленных задач используется область знаний под названием SLAM (одновременная локализация и построение карты). В настоящее время есть много реализаций метода SLAM, отличающихся сложностью и различными возможностями платформ.

Для отображения состояние окружающей среды современные системы используют вероятностные методы и ориентиры. Во время локализации робота, его состояние x_t , например, положение робота, должно оцениваться по всем предыдущим показаниям датчика $z(1, \dots, t-1)$ и предыдущим действиям робота $u(1, \dots, t-1)$. Локализация

роботов часто решается с использованием свойства Маркова, то есть предполагается, что мир статичен, шум не зависим и нет ошибок аппроксимации во время моделирования. Таким образом, благодаря свойству Маркова, задача локализации сводится к сети Байеса, представленной на рис. 1.

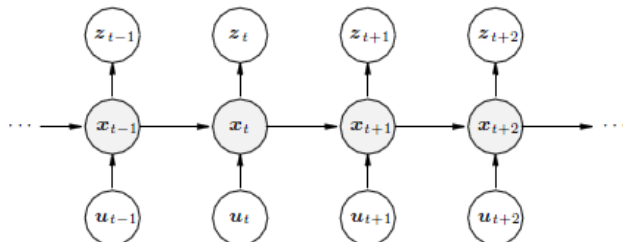


Рис. 1. Байесовская сеть для локализации робота.

В дополнение к состоянию робота, должна быть оценена карта m , как показано на рис. 2. Обычно карта состоит из n ориентиров, поэтому положение этих ориентиров должно быть оценено для построения карты.

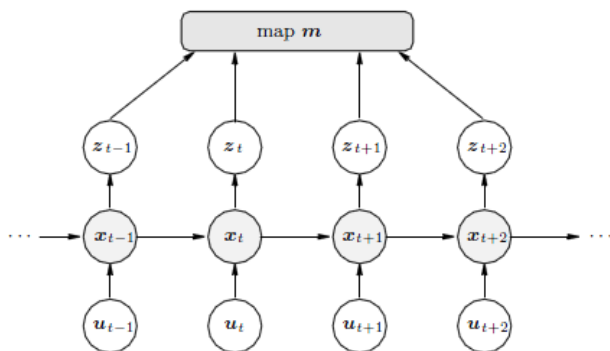


Рис. 2. Байесовская сеть для построения карты роботом.

Когда робот движется в известной ему местности, то есть ему дается карта, неопределенность положения робота сводится к минимуму. Однако, если робот движется в неизвестной ему местности, неопределенности положений робота могут быть большими, так как ошибки одометрии суммируются. Поэтому необходимо наблюдать за ориентирами, чтобы точно оценить положение робота с точностью до неопределенности положения ориентира и наблюдения. Датчики,

работающие в глобальной системе координат, такие как GPS/ГЛОНАСС, являются удобным решением использование одометрии, но они не всегда доступны, например, внутри зданий.

Замкнутые циклы играют важную роль в алгоритмах SLAM. Если робот определяет положение, в котором он находился ранее и правильно сопоставляет ориентиры, то накопление ошибки ограничивается [1]. Вероятностный метод, которые использовался в данной работе – это фильтр Калмана.

2. Фильтр Калмана

Фильтр Калмана – последовательный рекурсивный алгоритм, оценивающий вектор состояния динамической системы, используя ряд неполных и зашумленных измерений [2].

Алгоритм состоит из двух повторяющихся фаз: предсказание и корректировка. На первом этапе рассчитывается предсказание состояния в следующий момент времени (с учетом неточности их измерения). На втором, новая информация с датчика корректирует предсказанное значение (также с учетом неточности и зашумленности этой информации):

Уравнения на рис. 3 представлены в матричной форме. В случае с одной переменной матрицы вырождаются в скалярные значения.

Подстрочный индекс обозначает момент времени: k – текущий, $(k-1)$ – предыдущий, знак «минус» в верхнем индексе обозначает, что это предсказанное промежуточное значение.



Рис. 3. Алгоритм работы фильтра Калмана

3. Построение карты местности

Для иллюстрации работы алгоритма робота используем следующую карту местности, показанную на рис. 4.

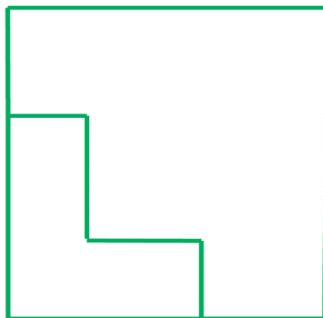
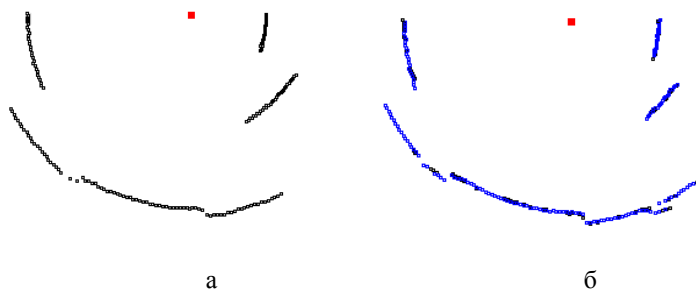


Рис. 4. Карта местности

Для начала рассмотрим построение карты местности на единичном шаге, на основе «сырых» данных, полученных с ультразвукового датчика. При прохождении по часовой стрелке получим следующее изображение (рис. 5а). Наложим на полученную карту данные с ультразвукового датчика при движении против часовой стрелки (рис. 5б).



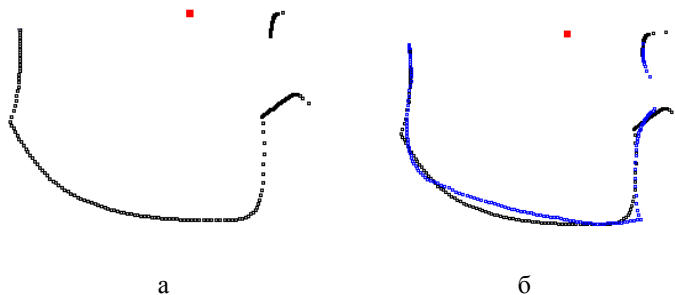
а – по часовой, б – по часовой и против часовой

Рис. 5. Карта местности, основанная на «сырых» данных

Таким образом, из этих рисунков можно сделать вывод, что «сырые» данные не дают реального представления о местности.

Теперь рассмотрим данные, которые приходят с ультразвуковых датчиков и обработанные фильтром Калмана. На рис. 6а изображены

полученные данные при вращении датчика по часовой стрелке. Наложим на полученную карту отфильтрованные данные с ультразвукового датчика при движении против часовой стрелки (рис. 6б).



а – по часовой, б – по часовой и против часовой

Рис. 6. Карта местности, основанная на отфильтрованных данных

Полученная карта также не будет полностью являться действительной. Поэтому на следующем шаге необходимо убрать неточности в построении, а именно, требуется оставить точки, которые пересекаются или находятся на близком расстоянии друг от друга, при вращениях датчика по часовой стрелке и против часовой стрелки. Полученный результат изображен на рис. 7.

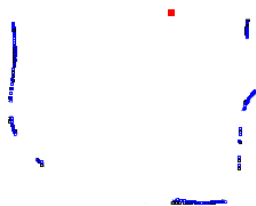


Рис. 7. Карта местности, построенная на реальных данных

Данные действия повторяются на каждом шаге робота (шаг робота в работе 10 см), в результате чего получается приближенная к реальной карта местности. Полученная карта изображена на рис. 8.

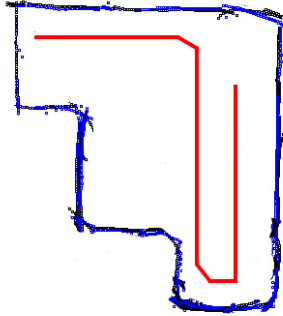


Рис. 8. Карта местности

Как видно из рис. 8, в целом, построенная карта соответствует реальным положениям препятствий, что подтверждает правильность работы робота и алгоритма построения карты. Наличие незначительных шумов в полученном результате, обуславливается погрешностями в измерениях ультразвуковых датчиков.

Заключение

Данная статья посвящена разработке и реализации алгоритма для робота с целью построения карты местности. Описаны метод одновременной локализации и построения карты, а также фильтр Калмана. Показаны результаты построения роботом карты. Данный алгоритм можно использовать для построения карты двухколесным мобильным роботом.

Список литературы

1. Zunino, G Simultaneous Localization and Mapping for Navigation in Realistic Environments : учеб. пособие / G Zunino – М. : КТН, 2002. – 187 с.
2. Балакришнан А. Теория фильтрации Калмана / А. Балакришнан – Пер. с англ.— М. : Мир, 1988. – 168 с.

Методы распознавания сарказма в тексте

Я. С. Полук

Студент магистр

Д. И. Соломатин

Ст. преподаватель

Введение

Анализ тональности текста – это область компьютерной лингвистики, занимающаяся проблемой автоматизированного определения эмоциональной окраски текстов в целом и по отношению к определенным объектам. Она используется при анализе мнений о продуктах или событиях, высказываемых на различных онлайн-площадках – сайтах интернет-магазинов, новостных площадках, в социальных сетях и т.д.

Одной из основных проблем, с которыми сталкиваются исследователи в области анализа тональности текста, является распознавание сарказма – язвительной насмешки, основанной на усиленном контрасте подразумеваемого и выражаемого и немедленном намеренном обнажении подразумеваемого. Зачастую для проведения такого анализа требуется знание каких-либо фактов или понимание контекста.

В данной работе будут рассмотрены некоторые методы распознавания сарказма на основе алгоритмов машинного обучения и их применение на реальном наборе данных.

1. Обзор методов распознавания сарказма

Рассмотрим некоторые методы, применяющиеся для анализа текста и в частности, для распознавания сарказма.

Первый метод - это сверточные нейронные сети [1]. Изначально они применялись для анализа изображений в силу их двумерной структуры, но позже стали использоваться и для анализа текста в силу своей способности распознавать паттерны в тексте.

Работа сверточных нейронных сетей основана на использовании свертки – операции над парой матриц A (размера $n_x \times n_y$) и B (размера $m_x \times m_y$), результатом которой является матрица $C = A \times B$ размера

$(n_x - m_x + 1) \times (n_y - m_y + 1)$. Каждый элемент результата $C_{i,j}$ вычисляется по следующей формуле:

$$C_{i,j} = \sum_{u=0}^{m_x-1} \sum_{v=0}^{m_y-1} A_{i+u,j+v} B_{u,v} \quad (1)$$

Эта операция позволяет с помощью ядра выделить небольшие, но значимые признаки.

Типичная сверточная нейронная сеть состоит из одного или нескольких наборов двух слоев: сверточного слоя и слоя субдискретизации (пулинга), заменяющего выход слоя в некоторой точке сводной статистикой близлежащих.

LSTM, или долгая краткосрочная память [2] - это разновидность архитектуры рекуррентных нейронных сетей, способная обучаться долговременным зависимостям, в отличие от стандартных рекуррентных сетей, и решающая проблему исчезающего/взрывного градиента. Благодаря этому сети с такой архитектурой могут «запоминать» контекст, в котором были использованы те или иные слова/словосочетания.

Основой ячейки LSTM (рис. 1) являются состояние ячейки и 3 вентиля (фильтра): забывания, входной и выходной, которые контролируют состояние ячейки.

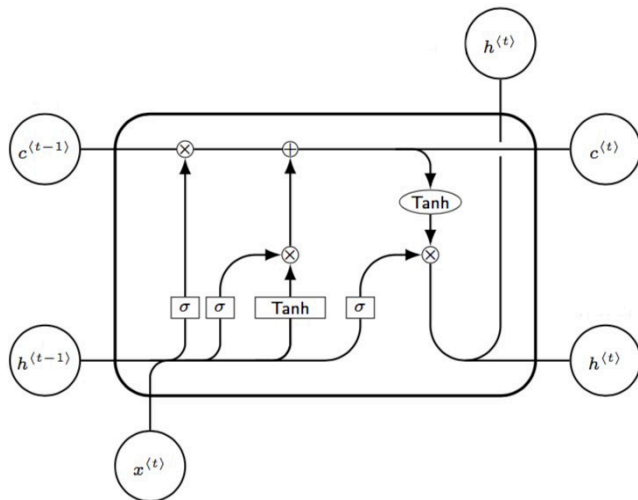


Рис. 1. Ячейка LSTM

Фильтр забывания определяет меру сохранения значения.

Входной фильтр определяет меру вхождения нового значения в память.

Выходной фильтр определяет, в какой степени значение, находящееся в памяти, влияет на значение выходной функции активации.

Математически работа фильтров выглядит следующим образом:

– фильтр забывания:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2)$$

где x_t - текущий входной вектор, h_{t-1} - вектор предыдущего скрытого слоя, содержащий выходы всех LSTM-ячеек, b_f , W_f - смещения и рекуррентные веса для вентилей забывания соответственно.

– входной фильтр:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (3)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (4)$$

Здесь сигмоидный слой, который и является входным вентиляем, определяет, какие значения необходимо обновить, а слой гиперболического тангенса создает вектор потенциальных новых значений.

– выходной фильтр:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \quad (5)$$

$$h_t = o_t \cdot \tanh(C_t), \quad (6)$$

o_t - вектор, содержащий сведения о том, какая информация из состояния сети будет выводиться, C_t - состояние текущей ячейки, h_t - результирующий вектор выводимой информации.

2. Сравнение методов распознавания сарказма

Для проведения сравнения был взят набор данных, составленный из комментариев пользователей, оставленных на ресурсе Reddit на английском языке [3]. Особенностью этого набора является то, что комментарии отмечались как содержащие сарказм непосредственно их авторами.

В наборе содержится одинаковое количество саркастических и не саркастических комментариев (по 505413 в каждой категории). Данные были разделены следующим образом: 80% комментариев составило обучающую выборку, 10% - валидационную, и 10% - итоговую тестовую выборку.

Данные были предварительно очищены: убраны все ссылки, html-теги, пунктуация, наборы подряд идущих символов, содержащих в себе как буквы, так и цифры, эмодзи, пробелы и переносы строк. После этого все тексты были приведены к нижнему регистру и выполнена их токенизация.

Для программной реализации использовался язык Python и библиотеки sklearn и Keras.

В качестве источника весов для Embedding-слоя были использованы готовые векторы, натренированные с помощью модели GloVe на текстах из Википедии и GigaWord.

Результаты работы моделей на тестовом наборе данных представлены в таблице.

Таблица

Результаты работы CNN и LSTM сетей на наборе данных Reddit

	CNN	LSTM
F1-мера	0.7184	0.7383
Точность	0.7184	0.7383

Из таблицы видно, что LSTM-сеть (рис. 2) ненамного превосходит сверточную сеть (рис. 3) по качеству работы на выбранном наборе данных. Дальнейшие результаты могут быть улучшены путем использования другого способа векторного представления слов, такого как word2vec, а также изменением структуры сетей, что, однако, следует делать с осторожностью, чтобы избежать переобучения.

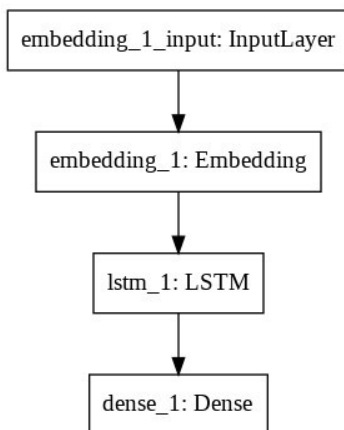


Рис. 2. Архитектура LSTM-сети

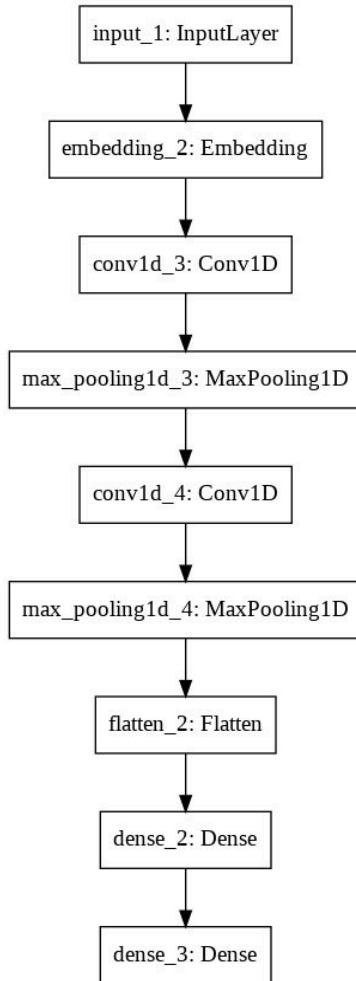


Рис. 3. Архитектура сверточной нейронной сети

Заключение

В данной работе были рассмотрены методы распознавания сарказма средствами машинного обучения и нейронных сетей и проведено сравнение результатов их работы на конкретном наборе данных. Также были предложены способы улучшения результатов работы моделей.

Список литературы

1. Гудфеллоу, Я. Глубокое обучение / Я. Гудфеллоу, И. Бенджио, А. Курвилль; пер. с англ. А. А. Слинкина. – 2-е изд, испр. – М. : ДМК Пресс, 2018. – 652 с.
2. Hochreiter, S. Long short-term memory / S. Hochreiter, J. Schmidhuber // Neural Computation. – 1997. – Vol. 9. – no. 8. – P. 1735-1780.
3. Khodak, M. A Large Self-Annotated Corpus for Sarcasm / M. Khodak, N. Saunshi, K. Vohrahalli // Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). – Miyazaki, Japan, 2018. – P. 641-646.

Преимущества предварительного выравнивания трёхмерных моделей при поиске разреженных соответствий

Л. А. Прохорченко

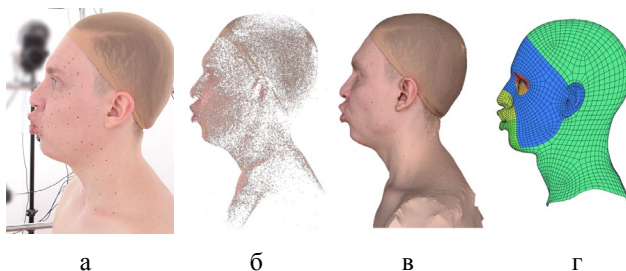
Студент магистр

Д. И. Соломатин

Старший преподаватель

Введение

Для современных кинофильмов и видеоигр всё чаще создаются цифровые дублёры реальных актёров. Так называемых *digital humans* производят при помощи различных технологий, ведущей из которых является фотограмметрия. Комбинируя набор фотографий лица актёра с различных ракурсов, можно сформировать облако точек в трёхмерном пространстве. Из них получают трёхмерную модель, также называемую *сканом*. Для последующего использования скана его необходимо обработать. В частности, необходимо снизить количество вершин модели до приемлемых величин. Для этого изменяют топологию скана – правило, по которому заданы полигоны. Её соотносят с какой-либо известной базовой моделью. Этапы процесса показаны на рис. 1.



а – фотография актёра, б – облако точек, в – скан,
г – модель в базовой топологии

Рис. 1. Поэтапное создание цифрового дублёра

Для того чтобы сохранить подобие во время преобразования, требуется набор *плотных соответствий* между сканом и этой базовой моделью. Это пары всех семантически равнозначных точек (кончик

носа – кончик носа, середина правой брови – середина правой брови). Нахождение подобных соответствий представляется нетривиальной задачей, так как любая семантика вершин теряется ещё на этапе преобразования фотографий в облако точек. Алгоритмы, позволяющие сократить время и повысить точность поиска плотных соответствий между двумя моделями, являются предметом актуальных исследований. В рамках данной работы будет проведён обзор существующих решений, а также показаны преимущества метода обнаружения соответствий с предварительным выравниванием трёхмерных моделей.

2. Постановка задачи

Для точной постановки задачи потребуется ввести ряд сквозных определений и подробнее раскрыть уже использованные термины.

Трёхмерная модель – это отображение реального объекта в трёхмерном пространстве, состоящее из набора вершин, соединяющих их рёбер и образуемых рёбрами полигонов. Модели, полученные при помощи трёхмерного сканирования, называют *сканами*.

Топология модели – это набор полигонов модели. Реальный объект может быть отображён бесконечным числом различных сканов, близких к нему. Изменение топологии модели с сохранением отображаемого объекта называют *ретопологией*. В рамках данной работы набор вершин и заданная на них топология будут в совокупности названы *геометрией*.

Семантически однозначные вершины – это вершины на различных сканах, определяющих одну и ту же точку реального объекта. В случае скана лица актёра, это соответствия по анатомическому признаку.

Плотное семантическое соответствие модели A к модели B – это множество семантически однозначных вершин модели B для каждой вершины модели A . *Частичным* или *разреженным семантическим соответствием* называют подмножество плотного соответствия.

В отличие от поиска плотного соответствия между двумя моделями, поиск разреженных соответствий является посильной задачей для человека, так как число характерных анатомических черт находится в пределах нескольких десятков, в то время как число вершин модели обычно находится в пределах от нескольких тысяч до нескольких миллионов. Существует ряд методов, например описанные в работах [1, 2], позволяющих аппроксимировать плотные соответствия моделей по частичным соответствиям. Для упрощения поиска частичных соответствий используют нанесённые на лицо маркеры.

Маркер – это сравнительно небольшое пятно, нанесённое на кожу актёра контрастным цветом. Расстановку маркеров на лице актёра, а также последующий процесс расстановки их на трёхмерной модели называют *разметкой*.

Поиск частичных соответствий между двумя моделями может быть упрощён до определения разметки модели по известному образцу. Образцовую разметку называют *базовой* или *нейтральной*. Пример используемой разметки приведён на рис. 2. Скан, для которого необходимо найти разметку называют *целевым*. Ручная разметка в известном смысле неконсистентна. В рамках данной работы проводится анализ методов автоматического обнаружения разметки, то есть поиска разреженных соответствий.

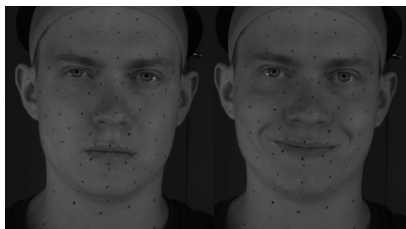


Рис. 2. Разметка на лице актёра при различных выражениях

3. Анализ задачи

Обнаружение маркеров может быть проведено по изображению. Это может быть набор фотографий или рендеры полученных моделей. После обнаружения соответствия, полученные положения маркеров можно спроецировать на трёхмерную модель. Нахождение схожих маркеров по изображениям часто сводится к *трекингу*, то есть обнаружению маркеров на последовательности схожих фотографий. Такое решение используется, например, в приложении [3] для отслеживания объектов в видеоряде.

Подход основан на предположении, что вид и положение маркеров на близких изображениях схожи, а потому их легко соотнести. Из-за того, что маркеры похожи друг на друга, имеет смысл добавить ограничение по области поиска маркера, чтобы уменьшить вероятность некорректного обнаружения. Эту область называют *патчем* (англ. *patch*). Для соотнесения точек двух областей можно использовать известные алгоритмы [4]. Метод позволяет достичь высокой консистентности разметки, а также ускоряет обнаружение для больших наборов данных, например видеозаписей с высокой частотой кадров.

Однако использование метода для неблизких в известном смысле выражений затруднительно. Повороты головы могут привести к пропаданию маркера с определённых ракурсов. Это вынуждает искать решения с несколькими проекциями и совмещением независимых результатов. Напрашивается внесение трёхмерного представления.

Задача соотнесения вершин топологий в трёхмерном пространстве рассмотрена в работах [54. 6, 6], однако подобные общие методы также не могут гарантировать соответствия при сильных деформациях лица. Однако для таких случаев применима идея близости. Общая схема предлагаемого подхода показана на рис. 3.

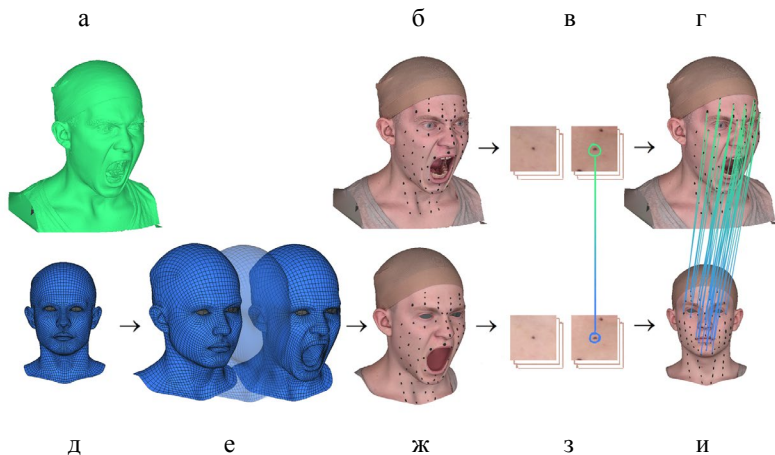


Рис. 3. Схема предлагаемого подхода к автоматическому поиску разреженных соответствий

Предполагается, что если базовая геометрия (д) и целевой скан (а) близки, то отображения маркеров на них сжуются. Предполагая некоторое выравнивание базовой геометрии (е) к скану (б), можно создать два набора патчей. Патчи базовой геометрии (з) будут содержать образцовую разметку, а патчи целевой геометрии (в) – искомые соответствия. Соотнесение маркеров по ним должно быть не хуже, чем в двухмерной задаче. Проекция найденных соответствий патчей (в-з) позволяет найти соответствия между трёхмерными моделями (г-и).

Вообще говоря, топологии геометрий различны. Поэтому их близость можно определить поверхностно, используя формулу (1).

$$L = \sum_{i=0}^N \|v_i - v'_j\|_2^2 \quad (2)$$

где v_i – i -ая вершина базовой геометрии, v'_j – ближайшая к ней вершина целевой геометрии. При устремлении $L \rightarrow 0$ геометрии поверхностно близки. Вообще говоря, поиск ближайшей целевой вершины для каждой вершины базовой геометрии, следует проводить

заново по мере поверхностного сближения моделей. Для оптимизации выравнивания имеет смысл использовать метод итеративных ближайших точек (ICP). Выравнивание может быть достигнуто некоторой параметрической трансформацией базовой геометрии, общий вид которой показан формулой 2.

$$V^{\text{transformed}} = D(V, \text{params}) \quad (3)$$

где $V^{\text{transformed}}$ – деформированные вершины базовой геометрии, D – некоторая модель деформации, params – её параметры. Задача выравнивания состоит в поиске параметров params из (2) для нахождения минимума (1), что приведёт к поверхностной близости геометрий (б-ж), как показано на рис. 4.



Рис. 4. Поверхностная близость базовой и целевой геометрии

Самым простым является выравнивание ригидного типа. Базовая модель деформируется как плотный целостный объект: переносом, поворотом и масштабам по формуле (3).

$$V^{\text{transformed}} = V \cdot R \cdot T \cdot S, \quad (4)$$

где R – матрица поворота, определяемая тремя углами Эйлера, T – матрица переноса, определяющая смещение каждой вершины, S – матрица юниформного масштабирования.

Ригидная трансформация не может обеспечить совпадения черт лица, лишь приблизительно соотнести облака вершин геометрий. Изменения черт, такие как закрытие глаз, открытие рта и подобное, возможно осуществить дополнив модель деформации D . Метод линейных сочетаний моделей может быть примером такой модели деформации. Его суть состоит в добавлении к базовой геометрии геометрий других выражений в той же топологии и семантики вершин. Такие геометрии называют *блендшейпами*, они обычно отображают предельные выражения. При этом подходе координаты i -ой вершины после трансформации описываются формулой 4, а общая деформация модели, с учётом добавления ригидной трансформации, описывается формулой 5.

$$\mathbf{v}_{\text{transformed}}^{(i)} = \sum_{k=0}^{N_B} \alpha_k \mathbf{v}_k^{(i)} \quad (5)$$

$$V^{\text{transformed}} = \left[\sum_{k=0}^{N_B} \alpha_k V_k \right] \cdot R \cdot T \cdot S \quad (6)$$

где $\mathbf{v}_k^{(i)}$ – i -ая вершина k -го блендшейпа, α_k – его вес, N_B – число блендшейпов.

Подбор параметров для минимизации (1) является задачей наименьших квадратов, а потому может быть решён при помощи оптимизации параметров. По аналогии с задачей, рассмотренной в работе [Ошибка! Источник ссылки не найден.], для реализации предпочтительно использование метода Левенберга-Марквардта. Параметрами деформации (2) будут переменные ригидной деформации и веса блендшейпов.

4. Реализация задачи

В листинге 1 представлен псевдокод алгоритма поиска соответствий. Исходными данными являются базовая геометрия, блендшейпы, целевая геометрия, образцовая разметка (*аннотация*).

Листинг 2

Псевдокод метода поиска разреженных соответствий

```

findMarkers
Geom base, Geom[] blends, Geom target, Point[] baseMarkers
-> Point[] targetMarkers

Number[] params = initParams()
for icpIter = 0..nIcpIters
  Geom current = deform(base, blends, params)
  Index basePointIndices = range(0..base.nVs)
  Index[] targetClosestPointIndices =
    findClosestPointIndices(current.vs, target.vs)
  do
    . Number loss = L(
    .   current, target,
    .   basePointIndices,
    .   targetClosestPointIndices)
    .
    . params = adjustParams(params, loss)
    . current = deform(base, blends, params)
    .
  while lossIsOk(loss)

Geom fitted = deform(base, blends, params)
Point[] cameraPoses = computeCameraPositions(fitted)

```

Окончание листинга 1

```
Image[] patchesBase = patches2d(cameraPoses, fitted)
Point2d[] baseMarkersPoses =
    projectMarkersToPatch(patchesBase, baseMarkers)

Image[] targetPatches = patches2d(cameraPoses, target)
Point2d[] targetMarkerPoses =
    match(patchesBase, targetPatches, baseMarkersPoses)

return projectPointsOnGeom(
    targetMarkerPoses,
    cameraPoses,
    target)
```

Приведённый код разбит на компоненты: инициализация параметров деформации, применение деформации, поиск ближайших вершин, расчёт поверхностной близости, изменение параметров методом ЛМ, расчёт положений виртуальных камер, рендеринг патчей, поиск соответствий между маркерами, проекция точек на поверхности модели на изображение и обратно.

Для реализации алгоритма был выбран язык программирования C++, для бизнес-логики – фреймворк *Qt*, для матричных операций – библиотека *Eigen*, для рендеринга геометрии моделей – движок *OpenGL*, для соотнесения маркеров в двумерной задаче – библиотека *OpenCV*. Структурное представление трёхмерных моделей является решённой задачей и предоставляется в открытых системах, таких как *MayaAPI*, *Blender*. Маркеры на поверхности могут быть заданы при помощи барицентрических координат на определённом полигоне геометрии.

Реализация деформации базовой геометрии показана в листинге 2.

Листинг 3

Реализация метода деформации базовой модели (детальный синтаксис опущен)

```
template <typename Scalar>
Geom<Scalar> Deform(
    Geom<Scalar> base,
    Matrix<Scalar> R, T, S,
    Geom<Scalar>[] blends,
    Scalar[] blendWeights)
{
    Matrix<Scalar> V = Vertices(base);
    for (ind = 0; ind < blends.size(); ++ind) {
        V += blendWeights[ind] * Vertices(blendShapes[ind]);
    }
    return V * R * T * S;
}
```

Здесь переменные *params* из (2) разделены на матрицы трансформаций и веса блендшейпов. Подставив вместо типа *Scalar* определённый пользователем тип *Derivable* можно добиться автоматического расчёта первых производных по каждому параметру, что показано в листинге 3. Подробнее приём автоматического дифференцирования и его использование при решении задачи оптимизации методом ЛМ описан в работах [7, 8].

Листинг 4

Частичная декларация типа *Derivable*

```
template <typename Scalar>
class Derivable
{
    Scalar value = Scalar(0.0);
    Scalar derivative = Scalar(0.0);
public:
    Derived<Scalar> operator*(Derived<Scalar> other) const;
    Derived<Scalar> operator+(Derived<Scalar> other) const;
    ...
};
```

Примеры патчей после использования выравнивания приведены на рис. 5. Соответствия найдены на них методом *OpenCV CrossCorrelation*. Общие полученные соответствия показаны на рис. 5.



Рис. 5. Примеры патчей после выравнивания

Заключение

Анализ, проведённый в ходе работы, обосновывает преимущества применения выравнивания моделей предложенным способом в задаче поиска разреженных соответствий между трёхмерными моделями. Был рассмотрена и реализована композиция ригидной деформации и смешивания блендшейпов для повышения эффективности выравнивания. Данное решение имеет несколько декомпозируемых этапов: определение поверхностной близости геометрии, выбор композиции моделей деформации, метод оптимизации, алгоритмы обнаружения соответствий на патчах. Каждый этап можно независимо подвергнуть последующему улучшению, что открывает возможности последующей доработки и улучшения алгоритма.

Список литературы

2. Blanz, V. Face recognition based on fitting a 3D morphable model / V. Blanz, T. Vetter // IEEE Transactions on pattern analysis and machine intelligence. – 2003. – Vol. 25. – Iss. 9. – P. 1063-1074.
3. Kraevoy, V. Cross-parameterization and compatible remeshing of 3D models / V. Kraevoy, A. Sheffer // ACM TOG – 2004. – Vol. 23. – Iss. 3. – P. 861-869.
4. PFTrack от The Pixel Farm Ltd [Электронный ресурс]. – Режим доступа : <https://www.thepixelfarm.co.uk/pftrack/>
5. Трекеры из библиотеки OpenCV [Электронный ресурс]. – Режим доступа : https://docs.opencv.org/3.4/d2/d0a/tutorial_introduction_to_tracker.html
6. Dense Semantic and Topological Correspondence of 3D Faces without Landmarks / Zhenfeng Fan [et al.] // 15th European Conference on Computer Vision (Munich, Germany, September 8-14, 2018). – Munich, 2018. – P. 541-558.
7. Землянухин, М. В. Эффективный поиск соответствий между трёхмерными объектами методом трансфера функций / М. В. Землянухин, Д. И. Соломатин // Сборник студенческих научных работ факультет компьютерных наук ВГУ. – Воронеж, 2018. – Т. 12. – С. 124-131.
8. Прохорченко, Л. А. Разработка приложения авторига для моделей с одинаковой топологией : ВКР Бакалаврская работа / Л. А. Прохорченко; ВГУ. – Воронеж, 2019. – С. 71-74.
9. Евтушенко, Ю. Оптимизация и быстрое автоматическое дифференцирование / Ю. Г. Евтушенко. – М: ВЦ РАН, 2013. – 144 с. – Режим доступа : <http://www.ccas.ru/personal/evtush/p/198.pdf>

Применение методологии Behavior-Driven Development в процессах разработки и автоматизированного тестирования ПО

С. Б. Рузин

Студент магистр

И. В. Илларионов

Доцент

Введение

В настоящее время, когда всё большее внимание уделяется качеству разрабатываемого программного обеспечения, невозможно представить себе процесс его разработки, который не включает в себя стадии его тестирования. Ведь без тщательной проверки на соответствие требованиям и стандартам, разрабатываемый продукт с высокой долей вероятности не сможет корректно выполнять требуемые от него функции. Именно поэтому стадии тестирования присутствуют в том или ином виде практически в любой из существующих методологий разработки ПО.

Вместе с тем команды, разрабатывающие ПО часто сталкиваются с проблемой разночтения требований к создаваемому продукту, что ведёт к дополнительным затратам на внесение изменений и переделку.

Из-за обширной функциональности современных программ и приложений, а также преобладания итеративных моделей разработки и связанной с этим необходимости непрерывной поставки продукта (Continuous delivery), существует потребность во внедрении постоянного автоматического тестирования основной функциональности продукта, чтобы сэкономить время ручных тестировщиков и при этом быть уверенным что внесенные изменения не привели к критическим сбоям в работе ПО.

Однако наиболее распространенные подходы к реализации автоматизированного тестирования имеют определенные недостатки, такие как читабельность программного кода и проблема синхронизации автоматизированных тестов с ручными.

Помочь с решением описанных выше проблем призваны методологии разработки ПО, основанные на тестировании, такие как BDD (Behavior-Driven Development, Разработка через поведение) [1].

Целью данной статьи является изучение применения методологии BDD в процессах разработки и тестирования программного обеспечения, разработка собственного инструментария для реализации задачи автоматизированного тестирования и последующее внедрение с анализом полученных результатов.

1. Подходы к автоматизированному тестированию ПО

Как правило, автоматизированные тесты делятся на тесты на уровне программного кода и тесты пользовательского интерфейса (User interface, UI). Тесты первого типа как правило создаются непосредственно разработчиками, в то время как вторыми занимаются специалисты по автоматизированному тестированию ПО. Однако наиболее распространенные подходы к автоматизации тестирования UI приводят к различным проблемам, таким как:

- Сложность синхронизации автоматизированных тестов с ручными;
- Сложность понимания операций, выполняемых в рамках теста, для людей, не имеющих специальной квалификации;
- Невозможность привлечения ручных тестировщиков к расширению тестового покрытия, так как от них требуется написание программного кода.

Одновременно с этим существует проблема того, что тестировщики, как и разработчики могут иметь свой взгляд на требования, предъявляемые к разрабатываемой системе, что ведет к потенциальным дополнительным затратам на внесение изменений, либо в сам продукт, либо в проверяющие его тесты.

2. Методология BDD

BDD – это методология разработки программного обеспечения, основной идеей которой является совмещение в процессе разработки чисто технических интересов и интересов бизнеса, позволяя тем самым управляющему персоналу и программистам говорить на одном языке [1].

В основе данной методологии лежит подход спецификации на примерах (Specification by example, SBE). Specification by example представляет собой совместный подход к определению требований и бизнес-ориентированных функциональных тестов для программных продуктов, основанный на создании требований с использованием практических примеров вместо абстрактных утверждений [2].

Таким образом в рамках методологии BDD ожидаемое поведение продукта должно определяться в рамках обсуждения между разработчиками, тестировщиками и владельцами продукта, используя для этого примеры, написанные на языке, доступным для понимания простому человеку. Позже шаги, определяющие ожидаемое поведение системы, привязываются к коду тестовой автоматизации, что дает возможность создания автоматизированных тестов, без написания программного кода.

При этом при использовании BDD должны сократиться некоторые расточительные действия такие как:

- Доработка, вызванная неверно понятыми или расплывчатыми требованиями;
- Техническая задолженность из-за нежелания проводить рефакторинг кода;
- Медленные циклы обратной связи из-за разрозненности и передачи обслуживания [3].

Языком, используемым для написания примеров, является Gherkin. Gherkin использует набор специальных ключевых слов для придания структуры и значения исполняемым спецификациям [4]. Пример подобного сценария представлен ниже.

Листинг 1

```
Scenario Outline: 014-02 Check FAQs and About us pages content
  Given user is signed in
  When user opens relative URL <url>
  Then "Page" title should become visible
  And the "Page" title text should be <titleText>
  And "Page" content should contain "Lorem ipsum dolor sit
  amet, consectetur adipiscing elit." text
Examples:
  | url           | titleText           |
  | "/faqs"      | "Frequently Asked Questions" |
  | "/about-us"  | "About US"         |
```

Данный сценарий описывает открытие пользователем одной из двух страниц и проверяет то, что после открытия они содержат определенные заголовки и текст.

Однако применение данного подхода требует внесения изменений в процессы разработки и тестирования ПО, а также разработки инструментария для создания автоматизированных тестов.

3. Выбор программных средств для разработки тестового фреймворка

Ввиду того, что основная часть программного кода тестируемого продукта была написана на языке C#, то рациональным выглядит подход

применения этого языка и для тестирования этого продукта. В качестве дополнительных библиотек были выбраны следующие:

- NUnit для запуска и выполнения тестов;
- Selenium Webdriver для взаимодействия с браузером;
- Specflow для работы с языком Gherkin, написания тестовых сценариев, а также их привязки к методам в программном коде [5];
- FluentAssertions для реализации различных проверок ожидаемого результата.

Упрощенная схема работа фреймворка представлена на рис. 1.

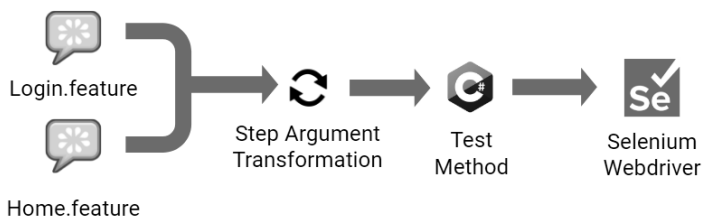


Рис. 1. Схема работы BDD фреймворка

На рисунке можно увидеть, что тестовые сценарии передают находящиеся в них аргументы в функцию реализующую механизм трансформации аргумента шага. Он отвечает за то, чтобы преобразовать текстовый аргумент из шага сценария к тому типу, который принимает тестовый метод, после этого метод обращается к Selenium, для выполнения действий в браузере. Базовый пример, выполняющий конверсию из string в int, представлен на рис. 2.

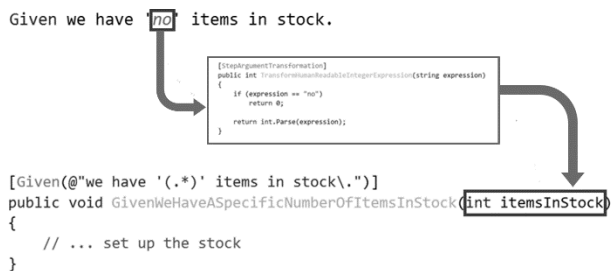


Рис. 2. Механизм Step argument transformation

При этом для работы с различными типами элементов, такими как таблицы, выпадающие списки и другие потребовалась реализация

дополнительной функциональности, однако это не является обязательным во всех возможных случаях.

4. Результаты внедрения BDD в процессы разработки и тестирования

В рамках реализации тестового фреймворка были удовлетворены следующие требования:

- Работа с наиболее распространенными типами элементов в web приложениях;
- Простота поддержки полученных автоматизированных тестов;
- Читабельность тестов даже для людей, не имеющих соответствующей квалификации;
- Расширяемость и возможность внедрения в разные продукты и типы ПО;
- Обширное и понятное логирование работы тестов.

Пример тестового лога представлен на рисунке 3. В него входит перечисление выполненных шагов, а также сравнение полученного результата с ожидаемым. Подобный формат лога позволяет использовать его для создания отчетов о найденных дефектах.

```
11:45:47 Error Message:
11:45:47 When user opens relative URL "/employee"
11:45:47 Then user should be redirected to People page
11:45:47
11:45:47 Expected:
11:45:47 "Search" input become enabled
11:45:47 Actual:
11:45:47 "Search" input is not enabled
11:45:47 Stack Trace:
11:45:47 at Behavioral.Automation.FluentAssertions
```

Рис. 3. Тестовый лог

Были введены следующие изменения в процессах разработки и тестирования:

- Сценарии, описывающие ожидаемое поведение системы, стали создаваться до реализации самой функциональности в рамках обсуждений между тестировщиками, разработчиками и стейкхолдерами;
- Специалисты по ручному тестированию были привлечены к созданию автоматизированных тестов;
- Написанные сценарии стали использоваться в качестве документации;
- Разработанный фреймворк был внедрен в системы непрерывной интеграции и поставки, что обеспечило постоянную проверку вносимых изменений.

Что касается статистики, из написанных в рамках пилотного проекта 470 сценариев было автоматизировано 469. Среднее время их выполнения в однопоточном режиме составило один час. При этом, из 469 автоматизированных сценариев, 200 были созданы ручными тестировщиками без написания кода. Также логи, полученные в результате выполнения тестов успешно использовались для создания отчетов о найденных дефектах. Разработанный фреймворк был рекомендован для использования и успешно применяется на других проектах.

Заключение

В данной статье были рассмотрены особенности применения методологии Behavior-driven development в процессах разработки и тестирования программного обеспечения. В рамках данной работы был также разработан инструментарий для создания автоматизированных тестов, который после успешного применения был внедрен в ряд разрабатываемых проектов. Грамотное применение этой методологии показало её преимущество при разработке требований, а также при расширении покрытия функциональности приложения автоматизированными тестами.

Список литературы

1. Carlos Solis, Xiaofeng Wang A Study of the Characteristics of Behaviour Driven Development. – [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/224265781_A_Study_of_the_Characteristics_of_Behaviour_Driven_Development
2. Gojko Adzic Specification by example: How successful teams deliver the right software/ Gojko Adzic// – Manning Publications Co, 2011. – 3 с.
3. Emily Maxie The Pros and Cons of Behavior-Driven Development. – [Электронный ресурс]. – Режим доступа : <https://www.verypossible.com/insights/the-pros-and-cons-of-behavior-driven-development>
4. Gherkin Reference. – [Электронный ресурс]. – Режим доступа : <https://cucumber.io/docs/gherkin/reference/>
5. Specflow Documentation. – [Электронный ресурс]. – Режим доступа : <https://docs.specflow.org/en/latest/>

Алгоритмы ранжирования товаров для электронной коммерции

В. Г. Сафонов

Студент магистр

М. Г. Матвеев

Профессор

Н. А. Алейникова

Доцент

Введение

При анализе и синтезе социально-экономических систем значительную роль играет обоснованный выбор активными элементами системы таких системных компонент и характеристик, которые формируют состояние системы, эффективное, с точки зрения активного элемента, принимающего решение [1]. В частности, в электронной коммерции (электронные торговые площадки, интернет-магазины) важную роль играют фильтры, задающие границы потребительских свойств товаров и обеспечивающие покупателю сужение полной совокупности товаров до существенно меньшей группы альтернативных товаров [2]. Фильтр представляет собой компьютерную программу, которая, обращаясь к базе данных располагаемой совокупности товаров, представленной в виде модели EAV, на основе логических операций проверяет - какие сущности удовлетворяют условиям фильтра. Модель «сущность-атрибут-значение» (EAV), это модель данных для кодирования пространственно-эффективным способом сущностей, где число атрибутов (свойств, параметров), которые могут быть использованы для их описания, потенциально велико, но число, которое фактически будет применяться к данной сущности, относительно небольшое.

Однако представленный классический подход к формированию фильтров уже не может в полной мере удовлетворять запросам покупателей [3], поскольку обладает рядом недостатков, из которых следует отметить следующие. Прежде всего фильтрация на основе бинарной логики не в состоянии обеспечить ранжирование множества

альтернатив с противоречивыми векторными характеристиками; требуется решение нетривиальной задачи векторной оптимизации. Кроме того, используемая бинарная логика не обладает достаточной выразительностью при описании желаемых покупательских потребностей [4, 5]. Подходом, устраняющим указанные недостатки, может служить применение сравнительного покомпонентного анализа соответствия (матчинга) покупательского спроса и предлагаемых товаров на основе бесконечнозначной (нечеткой) логики и использование операторов агрегирования полученного покомпонентного соответствия [6]. В этом случае бесконечнозначная логика обеспечивает требуемую выразительность покупательского спроса и гарантирует расположение представленных товаров на ранговой шкале. Статья посвящена описанию основных алгоритмов применения логики нечетких высказываний и предикатов для ранжирования типов однородного товара на рынке электронной торговли.

1. Алгоритмы оценки соответствия товаров потребительскому спросу

Разработка моделей покупательского спроса требует введения на рынке электронной коммерции единого товарного каталога. Такое требование не является уникальным. Унификация товарного каталога сегодня рассматривается как национальная задача, [7]. Мы предполагаем, что для каждого однородного товара в рамках единого каталога существует свой вид модели EAV, в векторной форме.

Пусть имеется некоторый однородный товар, представленный совокупностью своих взаимозаменяемых типов, различающихся значениями характеристических параметров (атрибутов) данного товара. Описание каждого j -го образца (типа) товара, $j = 1; 2; \dots; J$ будем представлять значениями характеристических параметров, описывающих коммерческие, технические и другие возможные свойства товара. Эти параметры удобно представлять соответствующим вектором:

$$q_j = (q_j^1; \dots; q_j^n; \dots; q_j^N) \quad (1)$$

каждая n -я компонента которого принимает значения x либо на количественной, либо на качественной шкале.

Будем считать, что каждый k -й покупатель, $k = 1; 2; \dots; K$ хочет приобрести однородный товар с желаемыми значениями характеристических параметров. Вполне естественно, что желания покупателя будут носить расплывчатый характер. В этом случае покупатель имеет возможность выбора, который обуславливается

взаимозаменяемостью типов однородного товара. Для этого покупателю необходимо формализовать свои желания и возможности в виде вектора спроса с «обезличенными», нечеткими характеристиками однородного товара:

$$\tilde{g}_k = (\tilde{g}_k^1, \dots, \tilde{g}_k^n, \dots, \tilde{g}_k^N) \quad (2)$$

Векторные компоненты здесь - суть лингвистические переменные с именами, совпадающими с именами соответствующих характеристических параметров описания типа однородного товара. Каждая переменная имеет кусочно-линейные функции принадлежности $f_g(x)$, носители которых - $x_{\min} \leq x \leq x_{\max}$ отражают возможности выбора потребителя, а значения функции – уровень его предпочтения (желания). Для дискретных значений носителя функция принадлежности будет иметь табличный вид. Нечеткие характеристические параметры формируют модель индивидуального покупательского спроса.

Пусть каждый i -й продавец, $i = 1; 2; \dots; I$, зашедший на ЭТП, формирует свои предложения по j -му типу однородного товара в виде вектора:

$$q_{ij} = (q_{ij}^1, \dots, q_{ij}^n, \dots, q_{ij}^N) \quad (3)$$

с четкими значениями характеристических параметров, структурно идентичного вектору спроса. В общем случае предложение представлено некоторой совокупностью векторов q_{ij} , $j = 1, 2, \dots, J_i$.

Оценка локальных (покомпонентных) соответствий товаров индивидуальному покупательскому спросу выполняется подстановкой значений компонент вектора (3) в функции принадлежности компонент вектора (2). Полученное значение функции принадлежности рассматривается как локальное соответствие предложение и спроса по указанной компоненте.

Для получения обобщенного соответствия необходимо агрегировать локальные соответствия при помощи того или иного оператора агрегирования [8]. Под агрегированием будем понимать переход от векторной оценки размерности n к скалярной величине

$$agr : \bigcup_{n=1}^N [0; 1]^n \rightarrow [0; 1] \quad (4)$$

Наиболее общий подход к агрегированию информации заключается в аксиоматическом определении оператора агрегирования. При этом

можно выделить три основные стратегии [9]: конъюнктивный (обобщенная оценка не может быть лучше самой плохой из локальных оценок); дизъюнктивной (обобщенная оценка определяется лучшей из локальных оценок) и компромиссная (обобщенная оценка занимает промежуточное положение между значениями локальных оценок, участвующих в агрегировании).

Будем рассматривать класс средних, удовлетворяющих условию [10]

$$agr(x_1; x_2; \dots; x_n) \in [\min(x_1; x_2; \dots; x_n); \max(x_1; x_2; \dots; x_n)] \quad (5)$$

Особенностью средних является наличие весов $w = (w_1; w_2; \dots; w_n)$ с помощью которых учитывается «вклад» каждой частной оценки в обобщенную оценку, при этом вес отражает значимость соответствующего источника информации (критерия, показателя, атрибута). В случаях, когда первичной является важность значений частных оценок, применяются порядковые операторы взвешенного агрегирования [11] – OWA-операторы, которые агрегируют компоненты векторной оценки, упорядоченные определенным образом:

$$OWA(x_1; x_2; \dots; x_n) = \sum_{j=1}^n w_j x_{\sigma(j)}, \quad (6)$$

где σ – индекс упорядочивания по уровню элементов $x_{\sigma(1)} \leq x_{\sigma(2)} \leq \dots \leq x_{\sigma(n)}$.

Допустим, что все критерии имеют с точки зрения потребителя одинаковую важность. Тогда правило задания весов можно определить следующим выражением

$$w_j = Q\left(\frac{j}{n}\right) - Q\left(\frac{j-1}{n}\right) \quad (7)$$

где Q – заданный тип нечеткого квантификатора.

Нечеткий квантификатор представляет собой нечеткое высказывание о допустимой форме компромисса между оценками x_j , отражающее интуитивное представление лица принимающего решение (ЛПР) о предпочтительности решений [12]. Нечеткое высказывание задается функцией принадлежности $Q(r) - Q\left(\frac{j}{n}\right)$, где r – интерпретируется как доля полученных оценок, для которых выполняется свойство, описанное функцией принадлежности квантификатора. Например, если задан квантификатор $Q = r$, который интерпретируется как «для как можно большего числа», то веса для

четырёх альтернатив будут заданы в соответствии с (7) следующим образом:

$$w_1 = 0,25; w_2 = 0,25; w_3 = 0,25; w_4 = 0,25.$$

То есть, квантификатор $Q = r$ задает частный случай оператора OWA – среднее арифметическое.

2. Численная апробация алгоритмов оценки соответствия

Пусть покупатель хочет проранжировать два типа однородного товара, характеризующиеся следующими четкими значениями параметров, заданных векторами: $(q_1^1 = 2; q_2^1 = 25; q_3^1 = 0,8)$ и $(q_1^2 = 3; q_2^2 = 20; q_3^2 = 0,7)$. Соответствующие потребности покупателя отображаются нечеткими значениями с функциями принадлежности: $f_1(q) = 1,5 - 0,25q; f_2(q) = 3 - 0,1q; f_3(q) = q$.

Соответствие характеристических параметров типов товара потребностям покупателя вычисляется путем подстановки значений компонент векторов в соответствующие функции принадлежности. В результате векторное соответствие, S каждого типа товара потребностям можно представить двумя векторами: $S_1 = (1; 0,5; 0,8)$ и $S_2 = (0,75; 1; 0,7)$.

Для ранжирования товаров, то есть для выяснения, который из них лучше соответствует потребности покупателя, воспользуемся оператором агрегирования OWA с вычислением весов с помощью квантификатора «для большинства», задаваемого следующей функцией принадлежности:

$$Q(r) = \begin{cases} 0, & \text{при } r \leq 0,4; \\ 2,5r - 1, & \text{при } 0,4 < r < 0,8; \\ 1, & \text{при } r \in [0,8; 1]. \end{cases}$$

Упорядоченные по убыванию вектора соответствий примут вид:

$$S_1 = (1; 0,8; 0,5) \text{ и } S_2 = (1; 0,75; 0,7)$$

Заметим, что вес w_j связан не с конкретным элементом вектора соответствия, а с его порядком в упорядоченном векторе.

При заданном квантификаторе $Q(r)$ и в соответствие с (7) веса принимают значения:

$$w_1 = 0; w_2 = 0,65; w_3 = 0,33.$$

Тогда агрегированное соответствие типов товаров вычисляется следующим образом:

$$S_1 = QWA(x) = 0 \cdot 1 + 0,65 \cdot 0,8 + 0,33 \cdot 0,5 = 0,685;$$

$$S_2 = QWA(x) = 0 \cdot 1 + 0,65 \cdot 0,75 + 0,33 \cdot 0,7 = 0,7185.$$

Полученный результат говорит о том, что второй тип товара немного предпочтительней первого.

Заключение

Применяемые в настоящее время различные способы фильтрации при организации поиска товаров, удовлетворяющих потребностям покупателей, позволяют выделять из всей совокупности товаров, подмножество товаров с векторными характеристиками, удовлетворяющими условиям фильтрации. Полученное подмножество содержит товары с допустимыми характеристиками, но отсутствует возможность их сравнения. Предложенный подход существенно расширяет возможности поиска. Он позволяет уточнять предпочтения покупателя, формализуя их в виде лингвистических переменных, что обеспечивает числовое представление соответствий по отдельным характеристическим параметрам товаров. Дальнейшее агрегирование локальных соответствий в глобальные, позволяет ранжировать допустимое подмножество товаров, полученное в результате фильтрации, обеспечивая покупателю поддержку принятия решений при выборе товара.

Литература

1. Айзерман, М. А. Выбор вариантов: основы теории. / М. А. Айзерман, Ф. Т. Алескерев – М.: Наука, 1990. – 240 с.
2. E-Katalog - каталог товаров, сравнение цен в интернет-магазинах России [Электронный ресурс]: каталог товаров. – Режим доступа : <https://www.e-katalog.ru/>
3. Поиск по каталогу: какими должны быть фильтры и сортировка каталога товаров [Электронный ресурс]: информационная статья. – Режим доступа : <https://lps.ru/blog/dirs/2018/kakimi-dolznyi-byit-filtry-i-sortirovka-kataloga-tovarov/>
4. Будяков, А. Н. Решение задачи выбора ресурсов и их поставщиков в условиях противоречивости технических и коммерческих требований / А. Н. Будяков, К. Г. Гетманова, М. Г. Матвеев // Вестник Воронежского государственного университета. Сер. Системный анализ и информационные технологии. – 2017. – №. 2. – С. 66-71.

5. Matveev, M. Models of Centralized Equipment Procurement Based on Supplier-Consumer Matching / M. Matveev, S. Podvalny. // 2019 1st International Conference on Control Systems, Mathematical Modelling, Automation and Energy Efficiency (SUMMA). – IEEE.– 2019. – P. 151-154.
6. Matveev, M. Automated Service for Product Offer Creation on the E-Trading Platform with Marketplace Technology / M. Matveev, S. Podvalny, Y. Yadgarova // 2020 2nd International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA). - IEEE. – 2020. – P. 672-676.
7. Национальный Каталог | национальный-каталогрф [Электронный ресурс]: база данных – Режим доступа : национальный-каталогрф
8. Detyniecki, M. Mathematical aggregation operators and their application to video querying. / M. Detyniecki PhD dissertation. Docteur de l'Universite. // – Paris, – 2000. – 185 p.
9. Леденева, Т. М. Агрегирование информации в оценочных системах / Т. М. Леденева, С. Л. Подвальный // Вестник Воронежского государственного университета. Сер. Системный анализ и информационные технологии. – 2016. – №. 4. – С. 155-164.
10. Джини, К. Средние величины / К. Джини. – М.: Статистика, 1970. – 448 с.
11. Yager, R. R. Quantifier guided aggregation using OWA operators / R. R. Yager // International Journal of Intelligent Systems, 11 (1996), P. 49-73.
12. Аверченков, В. И. Представление и обработка нечеткой информации в многокритериальных моделях принятия решений. / В. И. Аверченков, А. В. Лагерева, А. Г. Подвесовский // Вестник Брянского технического университета, 2012. – № 2 (34). – С. 97-104.

Обучение классификатора для распознавания лиц по алгоритму Виолы-Джонса

А. А. Смирнова

Студент магистрант

А. В. Акимов

Старший преподаватель

Введение

Распознавание групповых объектов – одна из самых развивающихся отраслей в области информационных технологий. Групповые объекты представляют собой набор определенных объектов, связанных между друг другом. Для распознавания группового объекта требуется детектор, который будет распознавать отдельные составные части.

В этой работе мы рассмотрим вопросы обучения и тестирования одной такой составной части будущего алгоритма распознавания групповых объектов на примере классификатора распознавания лиц по методу Виолы-Джонса на базе встроенной реализации каскадного детектора в программной среде Matlab.

1. Обзор детекторов для распознавания объектов в составе группы

Для распознавания объекта на цифровом изображении существуют различные методы. Большинство методов обнаружения по способу отнесения участка изображения к области интересов можно разделить на два вида это обобщающие и различающие.

Обобщающие методы составляют абстрактное, идеализированное представление о структуре объекта на основе позитивных обучающих изображений [1]. Такое представление может описывать изображение как схему, математическую модель, задающую через алгоритмы определенную структуру и свойства объекта. При анализе нового изображения обобщающие методы проводят сравнение и оценку точности соответствия с построенной моделью. Для обучения обобщающих алгоритмов не требуются негативные примеры изображений.

Как пример обобщающих методов можно рассмотреть модель случайного поля.

В модели случайного поля изображение разбивают на несколько частей [2]. Каждой части присваивается метка, характеризующая его содержимое, например: «дом», «вода», «земля», «небо», «человек». Решение принимается на основе алгоритма разделения и слияния, разбивающий изображения до тех пор, пока нельзя назначить метку, а затем объединяет одни и те же части с одной и той же меткой. Также, учитывается вероятность размещения двух различных меток на одной части. Таким образом, случайно поле состоит из совокупности назначенных меток и их вероятностям соответственно.

Для распознавания объекта различающие методы используют специальную функцию-классификатор, которая устанавливает свои параметры в ходе обучения. Обучение заключается в нахождении различий между позитивными и негативными примерами изображений. После обучения такая функция-классификатор может разделять входные изображения на содержащие и не содержащие позитивный пример объекта.

В статье [3] для обнаружения составного объекта используется усовершенствованный алгоритм Далала-Триггса. Каждое изображение в обучающей выборке делится на ячейки 8×8 пикселей, представляющую собой вектор. Внутри ячейки для каждого пикселя вычисляется модуль и направление градиента. Далее методом скользящего окна на пересечении ячеек формируются блоки размером 16×16 пикселей. Для каждого блока вычисляется вектор соразмерности, состоящий из 4 векторов ячеек. В свою очередь все блоки образуют длинный вектор, описывающий изображение через HOG-признаки. Чтобы разделить эти векторы на два класса с помощью метода опорных векторов [4] строится решающая поверхность. Отметим, что метод опорных векторов ищет ближайшие точки к линии разделения. Затем высчитывается расстояние между этими точками и линией, алгоритм стремится к максимальному расстоянию между ними. Соответственно, обученный классификатор сможет принять решение о том, находится ли на изображении искомый объект.

Как пример разделяющих методов также рассмотрим алгоритм Виолы-Джонса. Для распознавания алгоритм использует примитивы Хаара (рис. 1).

2. Описание каскадного детектора на примере Виолы Джонса

Алгоритм Виолы-Джонса, реализованный для распознавания лиц, был создан в 2001 году [5, 6]. Он использует метод скользящего окна, это означает что рамка меньшего размера, чем исходное изображение,

перемещается по изображению, и применяет слабые классификаторы для определения, есть ли заданный для поиска объект на изображении.

Многие современные алгоритмы детектирования полагаются на идеи, предложенные Виолой и Джонсом [5, 6]. Можно сказать, что такие детекторы тем или иным способом сначала описывают локальные особенности изображения с помощью простых функций, затем используют алгоритмы для усиления пороговых решающих правил и после организуют каскадный детектор [7]. Рассмотрим подробнее каждый этап работы детектора в алгоритме Виолы-Джонса.

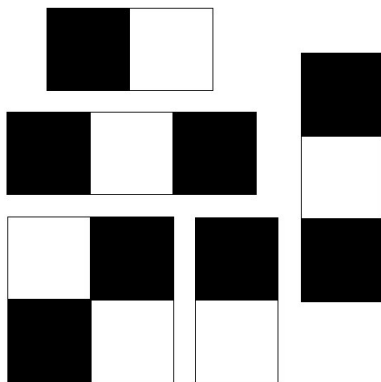


Рис. 1. Признаки Хаара

Локальные особенности изображения описываются с помощью признаков Хаара. Эти признаки дают точечное значение перепада яркости между черными и белыми областями.

В работе Виолы-Джонса при обучении признака выставляется граница, разделяющая тестируемые изображения на два класса: с объектом поиска и без объекта поиска. Эта граница должна быть поставлена таким образом, чтобы минимизировать ошибку классификации. В свою очередь слабый классификатор использует точечное значение, полученное от признаков Хаара, и сравнивает это с выставленной границей, что позволяет принять решение о том, принадлежит или не принадлежит изображение к искомому классу объектов.

$$h(x, f, p, \theta) = \begin{cases} 1, & \text{если } pf(x) < p\theta; \\ 0, & \text{иначе.} \end{cases} \quad (1)$$

Здесь $h(x, f, p, \theta)$ – слабый классификатор, $f(x)$ – признак, θ – граница, p – направление поиска, x – содержимое сканирующего окна.

В алгоритме Виолы-Джонса используется технология Adaboost или адаптивный бустинг. Процедура бустинга подразумевает комбинирование различных слабых и простых классификаторов в один сильный. Адаптивный бустинг, в отличие от «классического», характеризуется тем, что выдвигаемые требования к слабым классификаторам выделяют случаи, с которыми алгоритм не справился на предыдущем этапе.

Итоговый классификатор в алгоритме Виолы-Джонса представляет собой каскадный детектор. В ходе процедуры бустинга были получены комбинации слабых классификаторов, которые используются в виде слоев каскада. Каждый следующий слой отсеивает все больше негативных экземпляров обучающей выборки, но при этом верно определяют максимально возможное число позитивных примеров. При использовании такого детектора анализ сканирующего окна продолжается до тех пор, пока последний сильный классификатор не выдаст результат, что это окно соответствует искомому примеру, либо пока какой-нибудь из слоев не вернет отрицательный результат.

3. Описание средств Matlab для обучения детектора

Для обучения детектора лиц в среде Matlab по алгоритму Виолы-Джонса требуются обучающая выборка с положительными примерами и набор изображений без лиц в качестве негативных экземпляров.

Есть несколько способов подачи положительных экземпляров обучающей выборки при обучении каскадного детектора в среде Matlab [8]. Для объектов поиска первый способ заключается в представлении всех изображений в виде структуры данных таким образом, чтобы для каждой картинке были представлены путь к ней и известные позиции и размеры положительных результатов детектирования внутри нее. Второй способ, предоставление в Matlab заранее подготовленного Mat-файла, содержащего все необходимые положительные примеры.

Существует также несколько способов подачи негативных экземпляров на обучение. В этой работе был выбран способ с указанием пути к папке, где они хранятся.

С помощью функции `trainCascadeObjectDetector` устанавливаются требуемые параметры для обучения: задаем имя будущего XML-файла, подключаем позитивные и негативные примеры, настраиваем изменяемые параметры.

При тренировке детектора в среде Matlab возможно изменение следующих параметров.

- `ObjectTrainingSize` – задает размер обучающего объекта. Можно использовать режим «авто», где функция автоматически определит размер на основе среднего отношения ширины к высоте положительного примера. Либо задать параметр самому как двух элементный вектор [высота, ширина];

- `NegativeSamplesFactor` – это коэффициент для расчета количества негативных примеров, требуемых для тренировки детектора на каждом слое;

- `NumCascadeStages` – это максимальное (при наличии достаточного числа элементов обучающей выборки) количество слоев в каскаде (по умолчанию 20);

- `FalseAlarmRate` – это значение устанавливается в промежутке (0, 1] и обозначает допустимую долю негативных экземпляров, ошибочно рассматриваемых как положительные при тренировке каждого слоя каскада;

- `TruePositiveRate` – это значение устанавливается в промежутке (0, 1] и обозначает требуемую долю положительных примеров из обучающей выборки, рассматриваемых как действительно положительные при тренировке каждого слоя каскада;

- `FeatureType` – типы признаков для обучения: 'Haar' – признаки Хаара представляют собой черно-белые прямоугольники, вычисляющие разность между белыми и черными частями, используются в алгоритме Виолы-Джонса; 'HOG' – гистограмма ориентированных градиентов (их направлений в локальных областях изображения), используется в работе Далала-Тригса [3]; 'LPB' – локально бинарные шаблоны, представляют собой описания окрестностей пикселей в двоичной форме и используются для классификации текстур.

4. Тестирование детектора по алгоритму Виолы-Джонса

В листинге представлен код подготовки параметров и вызова алгоритма обучения каскадного детектора. Для тестирования построенного таким образом детектора была использована выборка CMU+MIT. На рис. 2 представлены примеры результатов его работы в сравнении с аналогичным встроенным в Matlab предобученным детектором.

Листинг

Обучение классификатора для распознавания лиц в среде Matlab

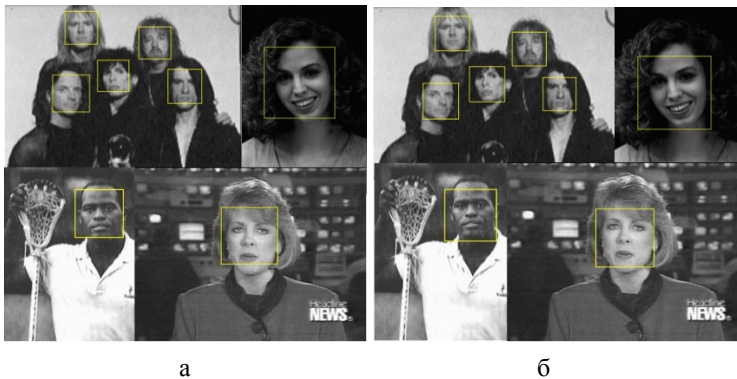
```
S = dir(fullfile(D, '*.*png'));  
for k = 1:numel(S)
```

```

F = fullfile(D,S(k).name);
I = imread(F);
bbox = [1 1 24 24];
positiveInstances {k,1}=F;
positiveInstances {k,2}=bbox;
V=cell2table(positiveInstances);
end
negativeFolder='D:\Nastya\image\AllNotFaces';
trainCascadeObjectDetector('Detector0995haar05.xml',
V, negativeFolder, 'ObjectTrainingSize', [24 24], 'FeatureType', 'Haar');

```

Результаты правильного распознавания составляют 148 из 168 лиц выборки при 11 ложных обнаружениях. При этом показатели работы встроенного в Matlab предобученного по алгоритму Виолы-Джонса детектора представляют собой 151 правильное распознавание из 168 лиц при 2 ошибочных обнаружениях.



*а – встроенный в Matlab предобученный детектор,
б – построенный в работе детектор*

Рис. 2. Сравнение результатов обнаружения лиц для алгоритма Виолы-Джонса

Работа над выбором оптимальных параметров все еще ведется, так как результаты пока что ниже оригинального алгоритма Виолы-Джонса.

Заключение

Рассмотрена модель работы различающего детектора на примере алгоритма Виолы-Джонса

Произведено сравнение результатов оригинального детектора Виолы-Джонса с новым детектором, обученным по этому алгоритму.

В итоге получен детектор, распознающий лица с результатами, приближенными к показателям алгоритма Виолы-Джонса. Работа над улучшением результатов детектора продолжается за счет подбора более оптимальных параметров. В дальнейшем планируется использовать полученный классификатор как часть детектора для распознавания групповых объектов.

Список литературы

1. Алфимцев А. Н. Метод обнаружения объекта в видеопотоке в реальном времени / А. Н. Алфимцев, И. И. Лычков // Вестник ТГТУ. – 2011. – Т. 17, № 1 – С. 44-54.
2. Murphy, K.P. Models for Generic Visual Object Detection / K.P. Murphy // Technical report, Department of Computer Science, University of British Columbia, Vancouver, Canada, May, 2005. – 8 p.
3. Dalal, N., and B. Triggs. “Histograms of Oriented Gradients for Human Detection.” IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Volume 1, 2005, pp. 886–893 Cortes C.,
4. Vapnik V. Support-Vector Networks// Machine Learning, 20. – 1995. – P. 273-297
5. Viola, P. Rapid Object Detection using a Boosted Cascade of Simple Features / P. Viola, M. Jones // Proceedings of the 2001 IEEE Conference On Computer Vision And Pattern Recognition 2001. – 2001. – Vol. 1. – P. 511-518.
6. Viola, P. Robust Real-Time Face Detection / P. Viola, M. Jones // International Journal of Computer Vision. – Netherlands: Kluwer Academic Publishers. – 2004. – № 57(2). – P. 137-154.
7. Калиновский, И.А. Обзор и тестирование детекторов фронтальных лиц / И.А. Калиновский, В.Г. Спицын // Компьютерная оптика. – 2016. – Т. 40, № 1. – С. 99-111
8. TrainCascadeObjectDetector. – <https://www.mathworks.com/help/vision/ref/traincascadeobjectdetector.html>

Обработка аудиосигнала на DSP-кластере СнК САЛЮТ-ЭЛ24Д

В. М. Солодунин

Студент магистр

Д. Н. Борисов

Доцент

Введение

Научно производственный центр электронные вычислительно-информационные системы (НПЦ «Элвис») является одним из немногих на Российском рынке, осуществляющий разработку и изготовление микропроцессорной техники, ряду изделий которого присвоен статус отечественных микросхем первого и второго уровня, что в условиях импортозамещения позволяет использовать их в оборонной промышленности РФ.

В работе рассматривается система-на-кристалле (или сокращенно СнК) на базе собственной платформы проектирования НПЦ «Элвис» – «Мультикор». Процессоры серии «Мультикор» – это однокристалльные программируемые многопроцессорные СнК, созданные на базе библиотеки IP [1] – ядер платформы Мультикор.

Процессоры данной серии позволяют сочетать в себе качества двух классов устройств: цифровых процессоров обработки сигналов и микроконтроллеров, что позволяет одновременно решать несколько задач: контроля и высокоточной обработки информации.

В данной работе рассматриваются особенности обработки звука с помощью DSP кластера семейства «Мультикор» – процессором 1892ВМ14Я [2] («Мультиком-02», МСom-02), при помощи ФНЧ фильтра Баттерворта 3-го порядка реализованного на СнК.

1. Система на кристалле и ее особенности

Микропроцессор «Мультиком-02» (МСom-02, 1892ВМ14Я) является 6-ядерным сигнальным микропроцессором с пониженным энергопотреблением созданным для мультимедийных, связанных, навигационных, а также встраиваемых мобильных приложений.

Микросхема изготовлена по КМОП-технологии с минимальными топологическими размерами элементов равным 40 нм.

Микроконтроллер имеет несколько основных модулей:

- стандартным управляющим процессором представленный двойным процессорным ядром – Dual CORTEX-A9 (CPU 0-1) с FPU-акселератором и NEON SIMD-акселератором (ARM);

- DSP-кластером [3] на основе двух DSP-ядер ELcore-30M [4] с полной программной совместимостью с микросхемами 1892BM10Я и 1892BM15АФ.

Кластер DSP представляет собой двухпроцессорную MIMD (multiple instruction, multiple data) систему. Каждое DSP ядро имеет в своем наличии собственную программную память и может работать независимо от других ядер. На верхнем уровне DSP-кластера имеется набор общих для всего кластера регистров управления и состояния.

2. Звуковые устройства

Для работы с аудио данными на СнК установлено несколько портов и внешних выводов, а именно:

- Два многофункциональных буферизированных последовательных порта (MFBSPP) в режиме I2S.

- Блок с 3 аналоговыми внешними выводами (LINE IN, LINE OUT, MIC IN), реализованный на базе аудиокодека SGTL5000.

Порт MFBSPP позволяет вести обмен параллельно-последовательным кодом с другими микросхемами по линковому интерфейсу (LPORT), либо обмениваться аудиоданными и управляющей информацией с внешними устройствами по последовательным интерфейсам в дуплексном режиме, с возможностью независимой настройки приёмника и передатчика.

Для работы с аудиоданными, данный порт должен быть настроен на режим I²S.

Порт используется как порт общего назначения и не является специализированным на передаче аудиоинформации.

Аудиокодек SGTL5000 – это стереокодек с низким энергопотреблением, разработанный для обеспечения комплексного аудиорешения для портативных устройств, которым требуются линейный вход, микрофонный вход, линейный выход, выход для наушников и цифровой ввод / вывод [5]. Структурная схема аудиокодека представлена на рис. 1.

Так как аудиокодек предоставляет нам в использование стандартные внешние выводы, то получение аудиосигнала реализуется через него.

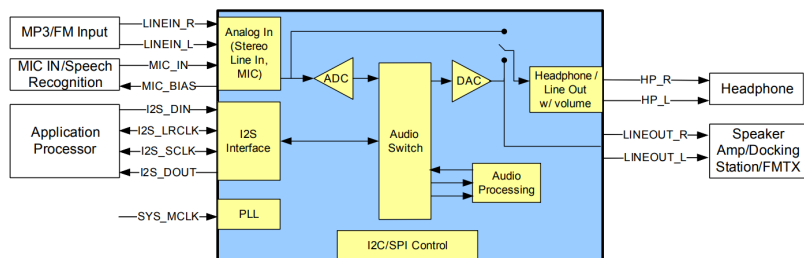


Рис. 1. Структурная схема аудиокодека

3. I²C и I²S протокол

Для возможности получения аудиоданных и настройки аудиокодека, на ШнК реализованы два протокола передачи данных: шина I²C и шина I²S. Для каждой шины на ШнК реализован свой контроллер.

Шина I²C – это двухпроводной последовательный интерфейс, состоящий из линии данных (SDA) и линии тактовой частоты синхронизации (SCL), с помощью которых происходит обмен данными между устройствами, подключенными к шине. На шине различают master- и slave-устройства. В каждом обмене участвует одно master-устройство и одно slave-устройство. Master-устройство инициализирует и контролирует передачу данных, генерирует тактовый сигнал. Каждое slave-устройство имеет уникальный адрес, по которому происходит обращение master-устройства. Протокол передачи выглядит следующим образом: данные передаются побайтово, число байтов в одной передаче не ограничено. После того, как master-устройство отправляет slave-устройству адрес с битом R/W или байт данных, slave-приемник должен выдать импульс подтверждения ACK в линию SDA. Если slave-приемник не отвечает импульсом ACK, master-устройство прерывает передачу и отправляет последовательность Stop.

Контроллер I²S предназначен для обмена аудио-данными в стерео-формате по последовательной шине I²S.

Шина I²S состоит из 4-х линий:

- SDO: линия данных передатчика;
- SDI: линия данных приемника;
- WS: линия выбора слова (частота дискретизации);
- SCLK: линия синхронизации.

На шине, как и в случае с I²C, различают master- и slave-устройства. Master-устройство генерирует сигналы SCLK и WS. Значение линии WS обозначает тип стерео данных, передаваемых в данный момент времени.

При $WS = 0$ передается «левое» слово, а при $WS=1$ – «правое». Передача данных всегда начинается с наиболее значимого бита (MBS).

Шина I²S будет использоваться для получения аудиоданных с внешнего вывода, которые будут проходить через аудиокодек.

4. Проектирование фильтра

Для обработки звука используется ФНЧ фильтр Баттерворта 3-го порядка. Основная задача фильтра заключается в уменьшении низкочастотного шума.

Проектирование ФНЧ фильтра Баттерворта 3-го порядка и генерация коэффициентов проводилась с использованием средства Filter Design and Analysis Toolbox (проектирования и анализа фильтров) MATLAB. Использовалась частота дискретизации 48 кГц и частота среза 5 кГц [6]. На рис. 2 представлены АЧХ и ФЧХ разрабатываемого фильтра.

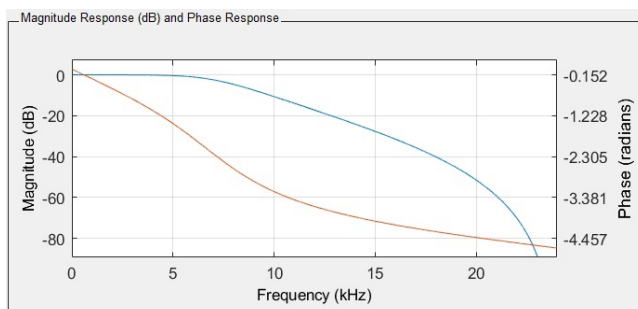


Рис. 2. Графики АЧХ (синий) и ФЧХ (оранжевый) фильтра

Рассчитанные коэффициенты фильтра представлены на рис. 3.

```
Numerator:  
0.020074623057528728081733149224419321399  
0.060223869172586180775752495719643775374  
0.060223869172586180775752495719643775374  
0.020074623057528728081733149224419321399  
Denominator:  
1  
-1.709589875829374605586963298264890909195  
1.133178688585446680647805806074757128954  
-0.262991828295842278162552929643425159156
```

Рис. 3. Коэффициенты ФНЧ фильтра Баттерворта 3-го порядка

5. Программная реализация фильтра

Программная реализация БИХ-фильтра осуществлялась для СнК «Мультиком-02», которая реализуется в виде набора отладочного модуля Салют-ЭЛ24Д1 и процессорного модуля Салют-ЭЛ24ОМ1 (НПЦ «ЭЛВИС»). Для программирования используется проприетарная среда (IDE) MCStudio4, построенная на основе свободной интегрированной среды разработки Eclipse, совместно с эмулятором MC-USB-JTAG использующий JTAG-порт СнК.

Особенностью построения программ для СнК заключается в необходимости программирования DSP-ядер, код для которого пишется на ассемблере, и программирования RISC процессора, код для которого код пишется на языке С. Данная особенность построения кода связана с архитектурой СнК [4].

После инициализации базовых регистров, используются коэффициенты фильтра, полученные на стадии проектирования в Matlab. Коэффициенты задаются в виде массива чисел с плавающей точкой, причем количество элементов массивов, формируется таким образом, чтобы он был кратен 2, для корректного выделения памяти [7].

Ниже (листинг 1, листинг 2) приведены листинги кода реализации фильтра.

Листинг 1

Фрагмент кода реализации фильтра на языке С под RISC

```
void filter(float* input, float* output, int length, float
B[], float A[])
{
    b0 = B[0];
    b1 = B[1];
    b2 = B[2];
    b3 = B[3];
    _a0 = A[0];
    _a1 = A[1];
    _a2 = A[2];
    _a3 = A[3];
    PC(0) = ((unsigned int)&Reset_DSP- 0x3a600000) >> 2;
    DCSR(0) = 0x4000;
    unsigned int startupPointer = ((unsigned
int)&Start_DSP- 0x3a600000) >> 2;
    for (int i = 0; i < length; i++)
    {
        In = input[i];

        //Run DSP core
        PC(0) = startupPointer;
        DCSR(0) = 0x4000;

        while (!(QSTR_DSP & (1 << 3)));
        output[i] = Out;    } }
```


*Фрагмент кода части реализации фильтра на ассемблере под
DSP*

```

Reset_DSP:
    MOVE    0, R6.L
    MOVE    0, R8.L
    MOVE    0, R10.L
    MOVE    0, R12.L
        STOP
Start_DSP:
    MOVE In, A0
    MOVE b0, A1
    MOVE _a0, A2
    MOVE (A0), R0.L ; Input
    MOVE (A1), R2.L ; B0
    MOVE (A2), R4.L ; A0
    FMPY R0.L, R2.L, R6.L
    MOVE b1, A1
    MOVE _a1, A2
    MOVE (A1), R2.L ; B1
    FMPY R0.L, R2.L, R14.L FADD R8.L, R6.L (A2),
R4.L ; A1

```

6. Настройка контроллеров и аудиокодека

Для настройки аудиокодека, необходимо реализовать на языке C доступ к регистрам контроллера I²C [3].

На СнК установлено 3 порта I²C, из документации к процессорному модулю следует, что для работы с аудиокодеком нам потребуется порт I²C1, т. к. именно он к нему подключен.

После нахождения адреса данного порта, необходимо произвести настройку контроллера указав требуемые значения в регистрах.

Для настройки контроллера нужно установить значения в битах регистра управления IC_CON, определяющие режим работы в master-режиме и скорость передачи (используется стандартная скорость).

В регистре адреса абонента IC_TAR указывается устройство, с которым производится взаимодействие, т. е. указывается адрес аудиокодека [2, 5].

После данной настройки производится взаимодействие с аудиокодеком. Передача информации аудиокодеку производится по 2 байта, т. к. адреса регистров имеют 16 битный формат. На рис. 4 указана структура данных, алгоритм чтения и записи в регистры аудиокодека.

Write Single Location

S	Device Address	W (0)	A	ADDR byte 1	A	ADDR byte 0	A	DATA byte 1	A	DATA byte 0	A	P
---	----------------	-------	---	-------------	---	-------------	---	-------------	---	-------------	---	---

Read Single Location

S	Device Address	W (0)	A	ADDR byte 1	A	ADDR byte 0	A	Sr	Device Address	R (1)	A	DATA byte 1	A	DATA byte 0	N	P
---	----------------	-------	---	-------------	---	-------------	---	----	----------------	-------	---	-------------	---	-------------	---	---

Рис. 4. Схема чтения и записи в регистры аудиокодека

Для настройки самого аудиокодека определяются соответствующие значения в регистрах питания и устанавливается путь потока данных от микрофона до цифрового вывода, с которого информация идет на шину I²S.

Контроллер I²S позволяет работать в режиме приемника и передатчика одновременно, при этом для передачи контроллер предоставляет 4 канала. Для получения данных с шины I²S, необходимо настроить контроллер I²S как приемник данных. При этом контроллер может работать в режиме приема только на нулевом канале.

Ниже (листинг 3 – листинг 5) приведены фрагменты кода настройки контроллеров и аудиокодека.

Листинг 3

Фрагмент кода настройки контроллера I²C

```
void init_i2c_tx(unsigned int i2cNum, unsigned int slave)
{
    I2C_CON_REG conReg;
    conReg._val = 0;
    conReg.mode = 1;
    conReg.speed = 1;
    conReg.ic_slave_dis = 1;
    conReg.rstrt_en = 1;

    I2C_TAR_REG tarReg;
    tarReg._val = 0;
    tarReg.ic10bit_addr_master = 0;
    tarReg.tar = 0x0A;
    tarReg.gc_or_start = 0;
    tarReg.special = 0;

    i2c->CON = conReg._val;
    i2c->TAR = tarReg._val;

    i2c->ACK_GENERAL_CALL = 1;
    i2c->ENABLE = 1; // Enable i2c;}

```

Фрагмент кода настройки контроллера I²S

```

void setup_i2s(unsigned int i2sNum)
{
    GATE_SYS_CTR |= CLK_I2S_EN;

    GPIO0(SWPORTD_CTL) |= GPIOD_I2S_SCLCO
        | GPIOD_I2S_SDI
        | I2S_GPIO_PINS[i2sNum][0]
        | GPIOD_I2S_WS
        | GPIOD_I2S_SCLK;

    IER_REG = 0;
    ITER_REG = 0;
    IRER_REG = 0;
    CER_REG = 0;
    RER0_REG = 0;
    TCR_REG(i2sNum) = 0;
    TER_REG(i2sNum) = 0;

    TFCR_REG(i2sNum) = 4;
    IMR_REG(i2sNum) = 0;
    IER_REG = 0x1;
    IRER_REG = 0x1;
    TCR_REG(i2sNum) = 0x4;
    TER_REG(i2sNum) = 1;
    CER_REG = 0x1;
}

```

Фрагмент кода настройки аудиокодека

```

write_sgtl_reg(SGTL5000_CHIP_LINREG_CTRL, 0x0008);
write_sgtl_reg(SGTL5000_CHIP_ANA_POWER, 0x7260);
write_sgtl_reg(SGTL5000_CHIP_REF_CTRL, 0x004E);
write_sgtl_reg(SGTL5000_CHIP_LINE_OUT_CTRL, 0x0322);
write_sgtl_reg(SGTL5000_CHIP_REF_CTRL, 0x004F);
write_sgtl_reg(SGTL5000_CHIP_SHORT_CTRL, 0x1106);
write_sgtl_reg(SGTL5000_CHIP_ANA_CTRL, 0x0133);
write_sgtl_reg(SGTL5000_CHIP_ANA_POWER, 0x6AFF);
write_sgtl_reg(SGTL5000_CHIP_DIG_POWER, 0x0073);
write_sgtl_reg(SGTL5000_CHIP_LINE_OUT_VOL, 0x0505);

i2cDelay();

write_sgtl_reg(SGTL5000_CHIP_ANA_HP_CTRL, 0x7f7f);

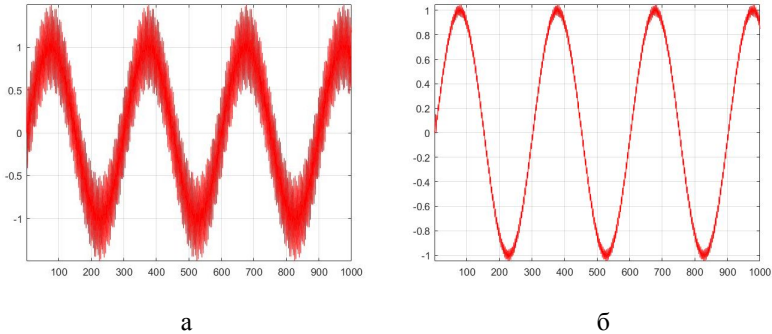
modify_sgtl_reg(SGTL5000_CHIP_ANA_CTRL, (1 << 4) ^
0xFFFF, 0);

```

7. Результат выполнения

В качестве входного сигнала используем тестовый аудиосигнал представляющий из себя сумму двух гармонических составляющих с частотами 100 и 8000 Гц. Наличие составляющей с частотой 8000 Гц приводит к появлению шипения в аудиосигнале, которое необходимо подавить фильтром.

На рис. 5 изображен исходный сигнал и результат фильтрации.



а – исходный сигнал, б – отфильтрованный сигнал

Рис. 5. Аудиосигнал сигнал

Как видно из рис. 5, б форма сигнала свидетельствует о подавлении синусоиды с частотой 8000 Гц, т. к. её частота больше частоты среза используемого фильтра, тем самым результат удовлетворяет поставленным изначально условиям.

Для оценки скорости работы фильтра на СнК сравним время работы фильтра на DSP-кластере модуля с временем выполнения фильтра в Matlab. Обработка тестовой выборки сигнала в среде Matlab равно 0.000154 секунд. Время работы фильтрации на DSP-кластере модуля равно 0.00001458 секундам, что на порядок быстрее обработки тестовой выборки в Matlab.

Заключение

В работе рассматривались особенности обработки звука на системы-на-кристалле 1892ВМ14Я НПЦ ЭЛВИС. Реализован программный фильтр низких частот Баттерворта 3-го порядка для уменьшения шума в аудиозаписи речи человека. Обработка сигнала с внешнего аудио устройства выполнена с использованием аудиокодека SGTL5000.

Список литературы

1. Baer, J. Microprocessor Architecture: From Simple Pipelines to Chip Multiprocessors / J. Baer. – Cambridge : Cambridge University Press, 2009. – 382 p.
2. Микросхема интегральная 1892VM14Я [Электронный ресурс] : Руководство пользователя – Режим доступа : https://multicore.ru/mc/data_sheets/Manual_1892VM14YA.pdf
3. DSP-кластер DELcore-30M. Архитектура [Электронный ресурс] : документация. – Режим доступа : https://multicore.ru/mc/data_sheets/Manual_DELcore-30M_031210.pdf
4. DSP-Ядро Elcore-30M. Система инструкций [Электронный ресурс] : Руководство пользователя. – Режим доступа : https://multicore.ru/mc/data_sheets/Manual_App1_DELcore-30M.pdf
5. SGTL5000: Ultra-Low-Power Audio Codec. Datasheet [Электронный ресурс] : Руководство пользователя – Режим доступа : <https://www.nxp.com/docs/en/data-sheet/SGTL5000.pdf>
6. Иванова, В. Цифровая обработка сигналов и сигнальные процессоры [Электронный ресурс] : учебное пособие / В. Иванова, А. Тяжев. – Самара : Поволжский государственный университет телекоммуникаций и информатики, 2017. – 253 с.
7. Уильямс, Г. Отладка микропроцессорных систем / Г. Уильямс. – М. : Энергоатомиздат, 1988. – 253 с.

Некоторые аспекты моделирования онтологии тестирования программного обеспечения

И. И. Стеблук

Студент магистр

И. В. Илларионов

Доцент

Введение

Оценка качества программного обеспечения – этап, выполняемый в том или ином виде в процессе разработки любого коммерческого программного обеспечения вне зависимости от масштаба проекта.

В ходе процесса тестирования генерируется большой объем данных, и это важно отметить: данные являются потенциальным источником знаний о лучших практиках, которые могут оформляться в виде руководств, справочных статей на корпоративном сайте и т. п.

Активный обмен знаниями внутри команды разработки, в свою очередь, повышает профессиональный уровень всех вовлеченных сторон, а также качество не только текущего проекта, но и последующих.

Особо заметим: чтобы извлечь знания из данных и оформить полученные результаты в доступном виде, необходимо минимизировать двусмысленность и неточность в используемых терминах и интерпретации полученной информации.

Описанная выше задача может быть эффективно решена посредством применения онтологии, однако в области тестирования ПО – в отличие, например, от биомедицинской области – к настоящему моменту устоявшаяся онтология предметной области отсутствует.

Таким образом, настоящая работа посвящена разработке онтологии предметной области тестирования программного обеспечения.

1. Основные теоретические положения

Настоящая работа представляет собой междисциплинарное исследование, предполагающее изучение тестирования ПО методами онтологического моделирования при широком использовании семантического анализа.

Обратимся к основным понятиям области настоящего исследования. Тестирование программного обеспечения, как известно, является одним из наиболее трудоемких и дорогостоящих этапов жизненного цикла ПО, и направлен на оценку соответствия его характеристик явным и подразумеваемым требованиям заказчика и конечного пользователя. В классическом понимании, цель тестирования – подтверждение качества ПО посредством систематической проверки программного обеспечения в тщательно контролируемых условиях [1].

Онтология, в свою очередь, задает словарь предметной области, определяя ее основные концепции и отношения между ними в машинно-интерпретируемом формате. Б. В. Добров отмечает, что термин «онтология» в настоящий момент связывают с широким спектром понятий, представляющих знания о той или иной предметной области, включая словарь с определениями, простую таксономию, модель с произвольным набором отношений, тезаурус и полностью аксиоматизированную теорию [2] – однако классически принятым определением считается емкая формула: онтология – это спецификация концептуализации [3]. Различают три уровня онтологии: верхнего уровня (междисциплинарные онтологии), уровня предметной области (онтологии областей знания) и прикладного уровня (онтологии конкретных задач и приложений). Вне зависимости от уровня, онтологии служат для описания типов, свойств и связей объектов конкретной области.

Важными областями приложения онтологий являются:

- Структурирование предметной области;
- Разработка информационной системы под управлением онтологии;
- Возможность получения информации при помощи SPARQL-запросов (для данных, напрямую содержащихся в онтологии) и SWRL-запросов (для данных, выводимых из содержащихся в онтологии).

2. Методология исследования и использованные инструменты

Проведенное исследование можно описать, используя фреймворк методологии научного проектирования [4], состоящий из шести шагов:

1. Проблема настоящего исследования – отсутствие устоявшейся онтологии предметной области тестирования ПО. Существенными вызовами также являлись следующие аспекты:

- состояние стандартизации предметной области: анализ стандарта ISTQB затруднен как отсутствием явной иерархии категорий, так и наличием дублирующих и пересекающихся категорий;
- состояние научной разработанности области исследования: мы обнаружили, что за рубежом аналогичные исследования начали

выполняться сравнительно недавно, а прямых отечественных аналогов им еще не существует;

– необходимость проводить семантический анализ и сопоставление категорий, изложенных на иностранном языке: проектирование и наполнение онтологии пересекались с переводческой деятельностью;

– необходимость отражения в онтологии практически значимых категорий и процессов: сопоставив реальную практику тестирования и существующие онтологии мы пришли к выводу, что в последних присутствует ряд упущений, особенно в отношении реализации свойств и связей классов.

2. Цель исследования – разработка такой онтологии, которая призвана обеспечить семантическую стандартизацию предметной области, направить разработку новых онтологий и выступить в качестве вспомогательного средства для согласования или улучшения существующих.

3. Проектирование и разработка – для разработки онтологии предметной области использовался смешанный метод (Ontology 101 [5] и Methontology [6] с учетом более современных концепций eXtreme Design [7], Experience Factory [8] и шаблонов проектирования онтологии [9]), а для наполнения онтологии – анализ и сравнение научной литературы, изучение стандарта предметной области ISTQB и учет собственного опыта практической работы в области тестирования. Ряд концепций был повторно использован из доступных онтологий предметной области.

4. Демонстрация – разработанный артефакт включает в себя онтологию предметной области с недвусмысленным набором сущностей, наиболее полно объединяющим область тестирования программного обеспечения.

5. Оценка – результат настоящей работы был оценен в несколько этапов: как по совокупным метрикам, так и по метрикам аксиом классов, свойств-связей и свойств-литералов в отдельности. Мы также дали количественную оценку выразительной силе онтологии, сравнили онтологию с существующими аналогами и обосновали отличительные качественные характеристики предлагаемой модели.

6. Коммуникация – результатами исследования являются текущая статья и готовящаяся магистерская диссертация, доступные прочим исследователям предметной области.

Упомянем также использованные в ходе исследования инструменты. Для моделирования онтологии мы избрали язык формального описания OWL, а для работы с данным языком было

решено применить широко используемое ПО Protégé, которое предоставляет набор инструментов для создания и визуализации онтологий – главными элементами онтологии в данной среде являются классы (сущности), свойства-связи (способы взаимосвязи классов) и свойства-литералы (свойства классов).

3. Оценка результатов исследования

Изложим и прокомментируем ключевые метрики предлагаемой онтологии. Во-первых, заметим, что объектов типа Individual мы не использовали. В онтологию вошло 138 классов, 72 связи-свойства и 119 свойств-литералов. Первый уровень классов состоит из одиннадцати сущностей: агент тестирования, вид тестирования, инструмент тестирования, дефект, метод рецензирования, объект тестирования, тестовое условие, тестовый артефакт, техника тест-дизайна, уровень тестирования (рисунок).

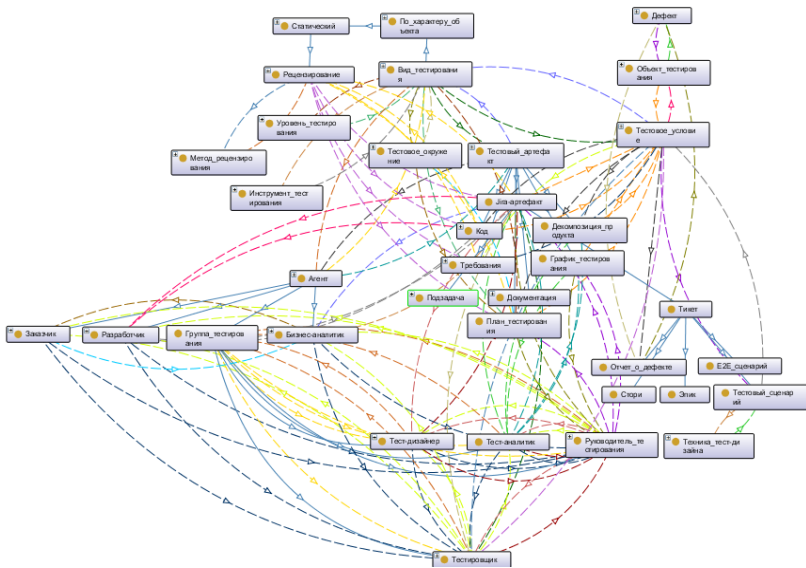


Рисунок. Свернутое представление предлагаемой онтологии

Особое внимание уделено проработке иерархии классов. Глубина вложенности отдельных подклассов внутри класса Вид тестирования, например, достигает пяти уровней, причем вопрос строгого разграничение сиблин-субклассов на основе Disjoint With стал отдельной задачей.

Была учтена взаимная сочетаемость видов тестирования по различным основаниям: например, как внутреннее, так и внешнее тестирование могут быть функциональным и нефункциональным – поэтому сиблинг-классы Вида тестирования первого уровня было решено не разграничивать. Иной пример – использование техник «белого ящика» не сочетается с техниками «черного ящика» и техниками, основанными на опыте, хотя последние свободно сочетаются между собой.

В работе над свойствами-связями мы уделили внимание реализации обратных связей, что обусловило широкое использование Inverse of, причем Domains и Ranges таких пар свойств взаимно обратны.

Свойства-литералы организованы особым образом. Возьмем класс Jira-артефактов «E2E сценарий» – ему соответствует единственное свойство «Свойства E2E-сценария», в которое входят семнадцать суб-свойств, причем каждое из них является Sub-Property для множества других наборов свойств, ассоциированных с прочими видами тикетов.

Дадим количественную оценку выразительной силе онтологии. Мы рассчитали среднюю величину Uses для различных классов и вложенных в них суб-классов и выделили сущности, наиболее сильно связанные с остальными. Наибольшее количество связей разработанной онтологии приходится на сущности: «Jira-артефакт», «Вид тестирования», «Тестирующим» и «Тестовое условие». Данные сущности являются ключевыми и наиболее проработанными.

Сравним полученную онтологию с существующими аналогами. Были выбраны TestTDO [10] и Top Domain Ontology for Software Testing [11], поскольку о них сообщается наибольшее количество данных, но ими возможности сопоставления и ограничиваются.

Таблица 1

Сравнение предлагаемой онтологии с аналогами

Онтология	N классов	N свойств-связей	N свойств-литералов
Авторская	138	72	119
TestTDO	46	42	50
TDOoST	107	25	4

Ввиду отсутствия прямых упоминаний о применении аксиом разделения типа Disjoint With нельзя исключать, что в TestTDO и TDOoST их применение не реализовано совершенно.

На сравнении количества свойств-связей следует остановиться особенно. Дело в том, что обратных связей между классами в рамках

TestTDO реализовано не было, а в авторской методологии и TDOofST они присутствуют. Отсюда можно сделать вывод, что TestTDO отражает даже большее количество связей, нежели авторская онтология – объяснение лежит области выделения в качестве классов таких виртуальных сущностей как «Test Information Need», «Test Goal», «Test Particular Situation» и т.п.

Ввиду подобных различий результаты количественного анализа и сравнения в области проектирования онтологий могут дать много данных, но мало сопоставимой информации – наиболее существенные различия лежат в качественной плоскости.

Настоящую работу можно рассматривать в качестве ответа на вызовы, стоявшие перед исследованием. Это обуславливает следующие особенности относительно аналогичных работ:

- Опора на стандарт предметной области ISTQB с учетом существующих аналогичных работ и практического опыта работы в тестировании;

- Полная русификация полученной модели;

- Учет ряда практически значимых связей и свойств, которые нашли отражение в ключевом суб-классе Документов тестирования под названием «Jira-артефакт». Jira – это система управления процессами, часто используемая на проектах разработки ПО для координации и мониторинга деятельности команд [12].

Следует далее наметить перспективы дальнейших исследований в предметной области. К ним можно отнести:

- дальнейшее углубление иерархии классов онтологии;

- детализация и уточнение свойств-связей и свойств-литералов по мере появления практической необходимости;

- дополнение классов онтологии лейблами с синонимами и названиями на иностранных языках (локализация);

- использование полученной онтологии в исследовательских проектах разработки программного обеспечения.

Обозначим перспективы практического применения разработанной онтологии. К ним относятся:

- использование в качестве интерактивной майнд-карты предметной области – данный способ наиболее прост во внедрении и предполагает использование онтологии как продвинутого словаря предметной области, с которым можно взаимодействовать посредством OntoGraf в обучающих целях;

- использование в качестве источника информации о тестировании при помощи SPARQL-запросов (для данных, напрямую

содержащихся в онтологии) и SWRL-запросов (для данных, косвенно выводимых из содержащихся в онтологии);

– использование для определения формата хранения метаданных о процессе оценки качества ПО на крупных проектах разработки для последующего извлечения практически значимой информации из базы данных. В идеале целесообразна реализация не только бэкенда с упором на базу данных, обновляющуюся в реальном времени, но и фронтенд в виде веб-сервиса, транслирующего пользовательские действия в SPARQL-запросы и выводящего полученные результаты. Данный способ может служить накоплению и распространению лучших практик тестирования, а также их использованию в последующих проектах.

Заключение

Данная статья посвящена разработке онтологии предметной области тестирования программного обеспечения. Описаны теоретические и методологические аспекты исследования, а также характеристики предлагаемой онтологии. Количественная оценка дана как по совокупным метрикам, так и по метрикам аксиом классов, свойств-связей и свойств-литералов в отдельности. Также приведены количественная оценка выразительной силы онтологии, сравнение онтологии с существующими аналогами и обоснование отличительных качественных характеристик предлагаемой модели.

В заключении приводятся направления дальнейших исследований и практическая ценность полученных результатов.

Список литературы

1. Miller E. F. Introduction to software testing technology //Tutorial: Software Testing & Validation Techniques, Second Edition, IEEE Catalog No. EHO. – 1981. – С. 180.
2. Добров Б. В., Лукашевич Н. В. Лингвистическая онтология по естественным наукам и технологиям для приложений в сфере информационного поиска //Ученые записки Казанского университета. Серия Физико-математические науки. – 2007. – Т. 149. – №. 2. – С. 49-72.
3. Souza É. F., Falbo R. A., Vijaykumar N. L. Ontologies in Software Testing: a Systematic Literature Review //VI Seminar on Ontology Research in Brazil. – 2013. – С. 71.
4. Rossi M., Sein M. K. Design research workshop: a proactive research approach //Presentation delivered at IRIS. – 2003. – Т. 26. – С. 9-12.
5. Noy N. F. et al. Ontology development 101: A guide to creating your first ontology. – 2001. – С. 24.

6. Fernandez M., Gomez-Perez A., Juristo N. Methontology: from ontological art towards ontological engineering //Proceedings of the AAAI97 spring symposium series on ontological engineering. – 1997. – C. 33-40.
7. Presutti V. et al. eXtreme design with content ontology design patterns //Proc. Workshop on Ontology Patterns. – 2009. – C. 83-97.
8. Von Wangenheim C. G, Althoff K. D., Barcia R. M. Goal-oriented and similarity-based retrieval of software engineering experienceware //International Conference on Software Engineering and Knowledge Engineering. – Springer, Berlin, Heidelberg, 1999. – C. 118-141.
9. Gangemi A., Presutti V. Ontology design patterns //Handbook on ontologies. – Springer, Berlin, Heidelberg, 2009. – C. 221-243.
10. Tebes G et al. TestTDO: A Top-Domain Software Testing Ontology //XXIII CIBSE. – 2020. – T. 20. – C. 1-14.
11. Asman A., Srikanth R. M. A top domain ontology for software testing: дис. – Master Thesis, Jönköping University, Sweden, 2016. – C. 65.
12. Filion L. et al. Using Atlassian tools for efficient requirements management: An industrial case study //2017 Annual IEEE International Systems Conference (SysCon). – IEEE, 2017. – C. 1-6.

Метод нормальных форм в задаче классификации голоморфно однородных поверхностей

Ю. Ю. Титова

Студент магистр

А. В. Атанов

Доцент

Введение

Идея преобразования объектов к удобному в каком-либо смысле виду перед их непосредственным изучением является одной из центральных в математике. Простейшими примерами здесь могут служить приведение матрицы к треугольной форме или приведение квадратичной формы к каноническому виду.

Похожие идеи активно используются и в комплексной геометрии – в частности, в задаче поиска и классификации однородных поверхностей многомерных пространств.

Под однородностью здесь понимается то, что для данной поверхности можно найти такое преобразование координат, которое переводит любую точку этой поверхности в любую другую точку этой же поверхности (элементарный пример: сфера).

Существует три основных подхода к решению этой задачи: дифференциально-геометрический; подход, использующий группы и алгебры преобразований; коэффициентный подход, в котором непосредственно анализируются уравнения поверхностей.

Несмотря на то, что эта задача чисто математическая, ее решение трудно реализуемо без привлечения компьютерных вычислений. В данной работе с использованием средств, которые предоставляют пакеты символьной математики, проводится автоматизация вычислений, возникающих при использовании коэффициентного подхода для решения задачи классификации однородных поверхностей в трехмерном комплексном пространстве.

1. Процедура построения нормального уравнения

Уравнение всякой невырожденной по Леви вещественно-аналитической гиперповерхности $M \subset \mathbb{C}^3$ может быть приведено (вблизи любой своей точки) к виду (см. [1]):

$$v = \langle z, z \rangle + \sum_{k,l \geq 2, m \geq 0} N_{klm}(z, \bar{z}) u^m, \quad (1)$$

где N_{klm} – однородные многочлены степени k по переменным $z = (z_1, z_2)$ и степеней l по переменным $\bar{z} = (\bar{z}_1, \bar{z}_2)$, $u = \operatorname{Re}(w)$, $v = \operatorname{Im}(w)$; $\langle z, z \rangle$ – невырожденная эрмитова форма (форма Леви поверхности), имеющая вид $z_1 \bar{z}_1 + z_2 \bar{z}_2$ (строго псевдовыпуклый случай) или $z_1 \bar{z}_2 + z_2 \bar{z}_1$ (индефинитный случай).

В работе изучаются (и нормализуются) так называемые жёсткие поверхности, то есть такие, в уравнениях которых нет зависимости от переменной u . Отметим, что среди однородных поверхностей свойство жёсткости является очень частым.

Уравнение жёсткой вещественной поверхности M трёхмерного комплексного пространства имеет вид

$$v = \langle z, z \rangle + \sum_{k,l \geq 0, k+l \geq 3} F_{kl}(z, \bar{z}) = \langle z, z \rangle + (F_{30} + F_{21} + F_{12} + F_{03}) + \dots \quad (2)$$

Здесь $F_{kl}(z, \bar{z})$ – однородный полином степени k по переменным $z = (z_1, z_2)$ и степеней l по переменным $\bar{z} = (\bar{z}_1, \bar{z}_2)$, $v = \operatorname{Im}(w)$.

Рассмотрим процедуру построения нормального уравнения по шагам.

Шаг 1. Замена переменных

$$w^* = w - 2i \sum_{k \geq 2} F_{k0}(z, \bar{z})$$

обнуляет слагаемые F_{k0} и F_{0k} , сохраняя при этом все остальные слагаемые уравнения (2).

Шаг 2. Слагаемые F_{k1} и F_{1k} в (2) удаляются с помощью замены

$$z_j^* = z_j + Q_j + T_j + C_j + \dots, \quad j = 1, 2, \quad (3)$$

где

$$Q_j = q_{j1} z_1^2 + q_{j2} z_1 z_2 + q_{j3} z_2^2;$$

$$T_j = t_{j1} z_1^3 + t_{j2} z_1^2 z_2 + t_{j3} z_2^2 z_1 + t_{j4} z_2^3$$

и так далее. Однако уравнение (2) записано в переменных z , а не z^* , поэтому последнюю замену (3) напрямую использовать мы не можем. Требуется найти обратную к ней замену, то есть фактически выразить z^* через z :

$$z_j = z_j^* + \phi_{j2} + \phi_{j3} + \phi_{j4} + \dots, \quad j = 1, 2, \quad (4)$$

где φ_{jk} – однородные полиномы, зависящие от переменных z_1^*, z_2^* .

Записав очевидное тождество

$$\left\{-z_j + z_j^* + \varphi_{j2} + \varphi_{j3} + \varphi_{j4} + \dots\right\}_{z_j^* = z_j + Q_j + T_j + C_j} \equiv 0, \quad j = 1, 2, \quad (5)$$

мы можем выделить в его левой части слагаемые степеней 1, 2, 3, ..., и, приравнявая их нулю, однозначно определить все полиномы φ_{jk} .

На этом этапе возникают самые громоздкие вычисления. Дело в том, что после замены (3) z_j^* становятся полиномами, и при подстановке этих переменных в полиномы φ_{j2} совокупные степени получающихся многочленов оказываются настолько большими, что, например, система Maple (см. [2]) справляется с такой подстановкой уже на самых младших слагаемых.

Однако, как было сказано ранее, для изучения однородных поверхностей достаточно рассматривать компоненты, имеющие степень не выше шестой. Поэтому при реализации данного алгоритма производилась автоматическая «редукция» степеней до 6-й включительно в каждом из слагаемых тождества (5). Это позволило существенно ускорить вычисления и однозначно определить полиномы φ_{j2} , а значит, и всю замену, удаляющую из уравнения (2) многочлены F_{k1} .

После выполнения шага 2 алгоритма уравнение (2) преобразуется к виду

$$v = \langle z, z \rangle + H_{22} + (H_{32} + H_{23}) + (H_{42} + H_{33} + H_{24}) + \dots \quad (6)$$

Здесь $H_{kl}(z, \bar{z})$ – однородный полином степени k по переменным $z = (z_1, z_2)$ и степеней l по переменным $\bar{z} = (\bar{z}_1, \bar{z}_2)$, $v = \text{Im}(w)$.

Обратим внимание, что после шага 1 полиномы, участвующие в записи уравнения (2), не менялись (обнулились лишь полиномы вида F_{k0}), в то время как после выполнения шага 2 произошло полное изменение слагаемых уравнения.

Шаг 3. Как видно, уравнение (6) уже достаточно близко к нормальной форме Мозера (1), однако для окончательного перехода необходимо выполнить ещё одно «нормализующее» преобразование координат

$$\begin{cases} z_j^* = z_j + f_{(3)}^j + f_{(4)}^j + f_{(5)}^j + \dots, & j = 1, 2, \\ w^* = w + g_{(4)} + g_{(5)} + \dots \end{cases} \quad (7)$$

Здесь $f_{(k)}^j, g_{(l)}$ – однородные полиномы, зависящие от переменных z_1, z_2, w . В записи (7) в скобках указана не степень, а вес слагаемого, при этом вес равен сумме степени слагаемого по переменной z_1 , по переменной z_2 и удвоенному значению степеней по переменной w .

Очевидно, что при осуществлении замены (7) все проблемы, связанные с существенным возрастанием степеней получающихся полиномов, описанные на шаге 2, сохраняются и даже усугубляются. Однако для рассматриваемых жёстких уравнений в [3] выписаны конкретные формулы перехода от полиномов $H_{kl}(z, \bar{z})$ к нормализованным полиномам из уравнения (1). Используя эти формулы, получаем в итоге искомый вид нормального уравнения Мозера:

$$v = \langle z, z \rangle + N_{220} + (N_{320} + N_{230}) + (N_{420} + N_{330} + N_{240}) + \dots$$

Отметим, что после замены (7) жёсткое уравнение, не содержащее переменной u , становится нежёстким, поэтому в последней записи введена тройная нумерация, где третий индекс означает степень по переменной u .

2. Примеры нормализованных по Мозеру уравнений

Классификацию аффинно-однородных поверхностей трёхмерного вещественного пространства, приведённую в известной статье [4], можно использовать для описания некоторого семейства голоморфно-однородных поверхностей трёхмерного комплексного пространства. Для проверки работы нашего алгоритма продемонстрируем результат работы алгоритма для некоторых уравнений.

Пример 1. Рассмотрим уравнение

$$v = xy + \ln(x).$$

Разложим правую часть уравнения в ряд Тейлора

$$v = xy - \frac{1}{2}x^2 + \frac{1}{3}x^3 - \frac{1}{4}x^4 + \frac{1}{5}x^5 - \frac{1}{6}x^6.$$

Перейдем к комплексным координатам и выпишем матрицу эрмитовой формы

$$\begin{pmatrix} -1/4 & 1/4 \\ 1/4 & 0 \end{pmatrix}.$$

Её определитель равен $-1/16$, следовательно, эта поверхность относится к индефинитному типу. Сделаем замену переменных, которая приведет форму Леви к каноническому виду

$$z_1 = 2\sqrt{2}z_2^*, \quad z_2 = \sqrt{2}z_1^* + \sqrt{2}z_2^*.$$

Получим форму Леви

$$z_1\bar{z}_2 + z_2\bar{z}_1.$$

С помощью созданной библиотеки получилось записать полиномы $H_{22}, H_{32}, H_{42}, H_{33}$ уравнения (6):

$$H_{22} = -14z_2^2\bar{z}_2^2, \quad H_{32} = 72\sqrt{2}z_2^3\bar{z}_2^2,$$

$$H_{42} = -900z_2^4\bar{z}_2^2, \quad H_{33} = -\frac{2240}{3}z_2^3\bar{z}_2^3.$$

Пример 2. Рассмотрим еще одно уравнение

$$v = \frac{e^{\beta x} - y \sin(x)}{\cos(x)}.$$

Разложим правую часть уравнения в ряд Тейлора

$$v = xy - \frac{1}{2}(\beta^2 + 1) - \frac{1}{3}x^3y + \frac{1}{24}(\beta^4 + 6\beta^2 + 5) + \frac{1}{6}(\beta^2 + 3)\beta x^3 - \frac{2}{15}x^5y +$$

$$+ \frac{1}{720}(\beta^6 + 15\beta^4 + 75\beta^2 + 61)x^6 + \frac{1}{120}(\beta^4 + 10\beta^2 + 25)\beta x^5 + \dots$$

Перейдем к комплексным координатам и выпишем матрицу эрмитовой формы

$$\begin{pmatrix} (1+\beta^2)/4 & -1/4 \\ -1/4 & 0 \end{pmatrix}.$$

Её определитель равен $-1/16$, следовательно, эта поверхность относится к индефинитному типу. Сделаем замену переменных, которая приведет форму Леви к каноническому виду

$$z_1 = 2\sqrt{2}z_2^*, \quad z_2 = \sqrt{2}z_1^* + \sqrt{2}z_2^*.$$

Получим форму Леви

$$z_1\bar{z}_2 + z_2\bar{z}_1.$$

С помощью созданной библиотеки получилось записать полиномы $H_{22}, H_{32}, H_{42}, H_{33}$ уравнения (6). Выпишем только полином H_{22} ввиду громоздкости остальных:

$$H_{22} = -\frac{\beta^6 + 9\beta^4 + 15\beta^2 - 1}{(\beta^2 + 1)^3}|z_1|^4 + \frac{2(\beta^2 + 1)^3}{\beta^2 + 1}(z_1^2\bar{z}_1\bar{z}_2 + z_1z_2\bar{z}_1^2).$$

Заклучение

В ходе работы была разработана библиотека для построения нормального уравнения Мозера вещественной гиперповерхности трёхмерного комплексного пространства в системе символьной математики Maple. Изучены математические основы построения нормального уравнения, проанализированы уравнения некоторых гиперповерхностей.

Список литературы

1. Chern, S. S. Real hypersurfaces in complex manifolds / S. S. Chern, J. K. Moser // Acta Math. – 1974. – Vol. 133. – P. 219–271.
2. Лобода, А. В. Однородные строго псевдо-выпуклые гиперповерхности в \mathbb{C}^3 с двумерными группами изотропии / А. В. Лобода // Мат. сб. – 2001. – Т. 192, № 12. – С. 3–24.
3. Аладьев, В. З. Программирование и разработка приложений в MAPLE / В. З. Аладьев, В. К. Бойко, Е. А. Ровба ; Гродн. гос. ун-т им. Янки Купалы, Междунар. акад. ноосферы, Балт. отд-ие. – Гродно ; Таллинн : ГрГУ : Медунар. акад. Ноосферы, Балт.отд., 2007. – 454 с.
4. Doubrov, V. Homogeneous Levi non-degenerate hypersurfaces in \mathbb{C}^3 / V. Doubrov, A. Medvedev, D. The // Mathematische Zeitschrift. – 2021. – Vol. 297. – P. 669–709.

Плагин навигации игровых юнитов для Unity Engine

М. М. Тутельян

Студент магистр

Д. И. Соломатин

Старший преподаватель

Введение

Индустрия видеоигр претерпела крупные изменения за последние двадцать лет. Сейчас одним из самых перспективных направлений является создание мобильных игр. Именно там сейчас основной поток пользователей и, как следствие – максимальная прибыль. Если рассмотреть игры мобильного сегмента, то можно с уверенностью сказать, что спросом пользуются 2D игры или игры с 3D моделями окружения, которые все равно механически задействуют 2D плоскость. Одна из основных задач, с которой встречаются разработчики почти каждой игры – перемещение игровых юнитов и связанный с этим низкоуровневый искусственный интеллект. Часто задача усложняется тем, что игровые юниты должны не только идти по кратчайшему пути, но и выбирать путь относительно данных об окружении. То есть, к примеру, юнит, имеющий малое количество очков жизни, мог пройти по ловушке, которая делала выбранный путь летальным или нежелательным в текущей игровой ситуации. Также, помимо урона здоровью, есть и другие причины не выбрать кратчайший путь, зависящие от положения цели, поведения игрока (за которым, как правило следуют игровые юниты) и прочее.

Плагин, создание которого описано в данной работе, нацелен на упрощение разработки 2D игр на игровом движке Unity. Он включает в себя инструментарий для создания логики перемещения юнитов в игре и искусственный интеллект, предлагающий несколько различных методов по выбору кратчайшего пути до игровых объектов. Данные методы учитывают различные сведения об окружающем мире и обрабатывают их в унифицированную метрику, с которой может работать алгоритм поиска пути, что также дает игровым разработчикам добавить свои собственные данные о мире их игры. Это позволяет плагину быть достаточно универсальным для разного рода задач и жанров игр.

1. Постановка задачи

Целью данной работы является создание плагина поиска путей для игровых юнитов с использованием игрового движка Unity, который позволит разработчикам без больших временных затрат использовать для своей игры систему поиска путей и адаптировать ее под конкретный проект.

С системой поиска путей будут взаимодействовать 2 основных типа разработчиков:

- геймдизайнеры;
- клиентские программисты.

Геймдизайнеры — это люди, которые проектируют механики игр и производят их дальнейшую настройку. Нередко они являются feature owner (теми, кто контролирует работу по написанному ранее дизайну и распределяет задачи). В данном случае передвижение и искусственный интеллект юнитов, как раз относится к одной из механик игры.

Клиентские программисты занимаются целым рядом задач. В их работу может входить проектирование архитектуры, программирование клиент-серверного взаимодействия, механик игры, создание под них конфигураций «конфигов», верстка интерфейса игры, написание логики интерфейса, игровых систем и скриптов. Взаимодействуя с плагином, клиентский программист разрабатывает свои скрипты, которые поддерживаются системой плагина и приносят в его работу новые ключевые особенности под каждую конкретную игру.

2. Анализ предметной области

Программное решение, описанное в данной работе, можно отнести к плагинам или пакетам, которые могут поставляться с помощью Package Manager или заархивированные в asset для игрового движка Unity. Больше информации о Package Manager и об игровом движке Unity можно найти в источнике [1].

Игровой движок — это базовое программное обеспечение компьютерной игры. Разделение игры и игрового движка часто расплывчато, и не всегда студии проводят четкую границу между ними. Но в общем случае термин «игровой движок» применяется для того программного обеспечения, которое пригодно для повторного использования и расширения, и тем самым может быть рассмотрено как основание для разработки множества различных игр без существенных изменений.

Игровой ассет (англ. Game Asset) или Игровой ресурс — цифровой объект, преимущественно состоящий из однотипных данных, неделимая сущность, которая представляет часть игрового контента и обладает некими свойствами. Понятие «игрового ассета» используется при

разработке компьютерных игр по отношению к тем элементам контента, которые обрабатываются ресурсной системой как неделимые (атомарные, элементарные) сущности.

Плагин — это независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей. Плагины обычно выполняются в виде библиотек общего пользования. Если рассмотреть прочие плагины поиска путей для игрового движка Unity, то можно увидеть закономерность. Все плагины поиска пути имеют скрипты наследуемые от MonoBehaviour, то есть они могут быть применены к игровому объекту и тут же настроены под определенный вид карты и перемещения. Больше

Описанный в данной работе плагин поиска путей и искусственного интеллекта является таким ассетом и работает с движком Unity, главная цель плагина - упрощение цикла разработки 2D игр в сфере разработки механики перемещения юнитов и искусственного интеллекта в перемещении юнитов.

3. Анализ алгоритма AStar

Достижение пункта назначения по кратчайшему пути — это классическая задача в программировании. Ее решение имеет широкий список применений, но особенно часто эта задача встречается при создании разного рода видеоигр.

A-star (кратко – A*) является одним из наиболее успешных алгоритмов для поиска кратчайшего пути между узлами или графами. Это основанный на имеющейся информации алгоритм поиска, поскольку он использует сведения о стоимости пути и эвристические правила для поиска решения.

Временная сложность алгоритма A* зависит от эвристики. В худшем случае, число вершин, исследуемых алгоритмом, растёт экспоненциально по сравнению с длиной оптимального пути, но сложность становится полиномиальной, когда эвристика удовлетворяет следующему условию:

$$|h(x) - h^*(x)| \leq O(\log h^*(x))$$

где h^* — оптимальная эвристика, то есть точная оценка расстояния из вершины x к цели. Другими словами, ошибка $h(x)$ не должна расти быстрее, чем логарифм от оптимальной эвристики.

4. Архитектура

Реализуемая архитектура плагина, обеспечивает соответствие системы плагина и требований к его функциональности [2].

Наиболее обобщенное (глобальное) представление архитектуры отображено на рис. 1 и делится на несколько компонентов. Принцип построение диаграмм описан в источнике [3].

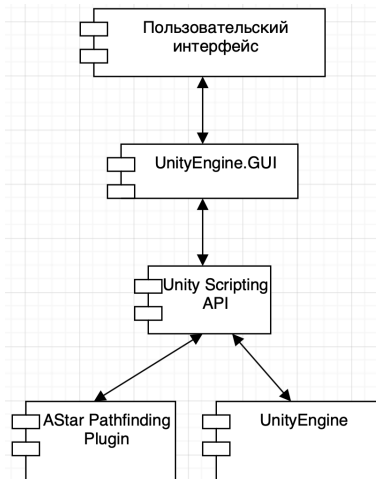


Рис. 1. Диаграмма компонентов

Через пользовательский интерфейс, который реализуется с помощью модуля UnityEngine.GUI пользователи (геймдизайнеры) могут настраивать поведения скриптов и систем.

Скрипты поведения юнитов, системы механик игры и сам плагин поиска пути взаимодействует с прочими компонентами UnityEngine через модуль Unity Scripting API. Используя различные библиотеки Unity описывается взаимодействие с физикой игры и игровыми объектами: инстанцирование объектов в сцене, их перемещение, физические коллизии, удаление из сцены или скрытие; открытие новых сцен и переход к ним из текущих.

5. Реализация алгоритма

В классической реализации A* существует несколько расчетных характеристик для узлов:

- $g(n)$ — это точная стоимость пути до узла, вычисляется как расстояние от текущего узла до соседнего узла n в сумме с накопленным расстоянием в текущем узле.

- $h(n)$ — это эвристическая расчетная стоимость от узла n до целевого узла, обычно используется евклидово расстояние;

- $f(n)$ — это наименьшая стоимость в соседнем узле n :

$$f(n) = h(n) + g(n)$$

В реализации алгоритма для плагина поиска путей характеристики пути были дополнены эвристическим весом пути и стоимостью пути до узла в метрике веса, по аналогии с предыдущими характеристиками:

- $ws(n)$ – стоимость пути до узла в весе;
- $wh(n)$ – эвристическая оценка стоимости пути от узла n до целевого узла в весе.

То есть итоговая метрика, которая применяется в выборе следующего узла в пути до целевого рассчитывается по формуле:

$$F(n) = f(n) + (ws(n) + wh(n)) * WeightRatio,$$

$$WeightRatio = [0..1].$$

При этом вес узла рассчитывается для каждого агента пути по своему и зависит от реализации интерфейса `IAgentWeightProvider`. По умолчанию в предложенных демонстрационных скриптах, которые поставляются вместе с плагином, реализован `MonoBehaviour` скрипт, который передает текущие очки жизни агента для сравнения с весом клеток, который передает урон, наносимый клеткой. Таким образом агенты могут обходить объекты, наносящие урон. Это только одно из многих применений расширенного алгоритма A^* . Архитектура плагина позволяет с легкостью дополнять логику расчета метрик веса программистами и вносить тем самым свою логику в поведение юнитов непосредственно под каждую игру. То есть система плагина может быть масштабирована в соответствии с требованиями пользователей. Больше информации о проектировании информационных систем и разработки программного обеспечения можно найти в источниках [4, 5].

6. Демонстрация работы плагина

На приведенных ниже примерах рассматривается использование веса клеток, как фактора избегания пути, то есть их проходимости или наносимого урона (опасности). Для демонстрации избегания кратчайших путей с большим весом показаны три варианта настройки избирательности пути относительно веса *WeightRatio*: низкий, высокий и средний.

На рис. 2 продемонстрирован скриншот работы плагина с отображением сразу трех вариантов путей:

1. Сплошная линия – путь с низким значением *WeightRatio*. Опасность пути не имеет значения (агенты идут по кратчайшему пути).

2. Пунктир черточками – путь со средним значением *WeightRatio*.
Агенты обходят опасные зоны только при удобной возможности (существенно не увеличивая путь).

3. Пунктир точками – путь с высоким значением *WeightRatio* (все опасные зоны избегаются).

Элементы сцены:

- прямоугольник с закругленным краем – это стена (непреодолимое препятствие для игрока и агентов);
- ромб – это агент, для которого рассчитывается путь;
- звезда – это цель, до которой рассчитывается путь;
- крест – это опасная область, которую могут обходить агенты.

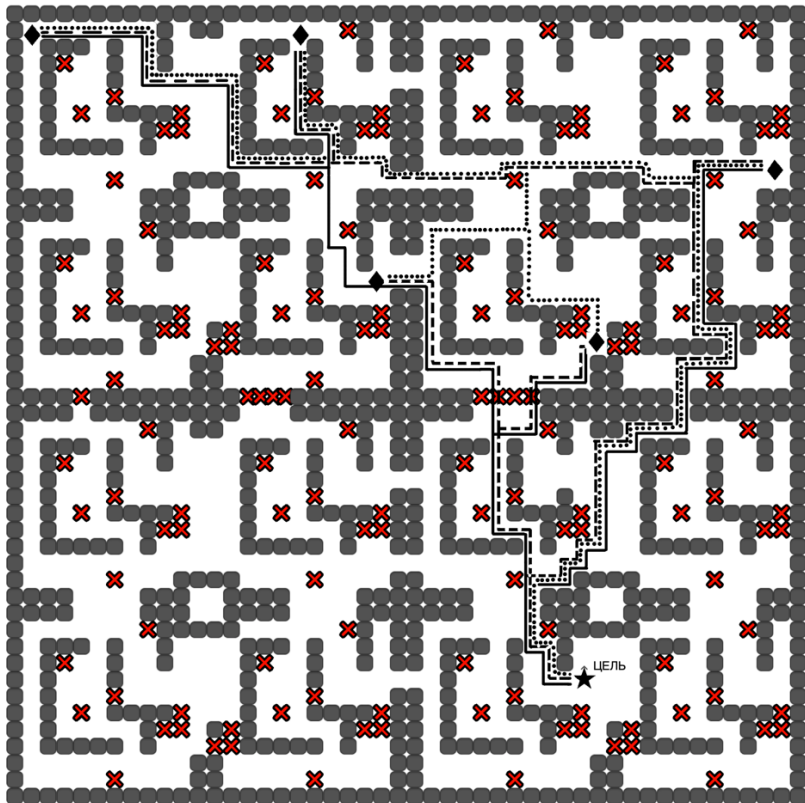


Рис. 2. Все три варианта рассчитанных путей

7. Заключение

В итоге проделанной работы, был создан плагин для игрового движка Unity, соответствующий основным требованиям по расширяемости системы и ее использованию, а также, продемонстрирована работа плагина на примере расчёта пути с избеганием опасных областей в сцене игры.

Список литературы

1. Документация Unity Packages [Электронный ресурс] : электронная документация. – Режим доступа : <https://docs.unity3d.com/Manual/PackagesList.html>
2. Архитектура информационных систем: учебник для студ. учреждений высш. проф. образования / Б. Я. Советов [и др.]. – М. : Издательский центр «Академия», 2012. – 288 с.
3. Методология IDEF [Электронный ресурс]: электронная энциклопедия. – Режим доступа : <https://ru.wikipedia.org/wiki/IDEF>
4. Белов, В. В. Проектирование информационных систем : учебник для студ. учреждений высш. проф. образования / В. В. Белов, В. И. Чистякова ; под ред. В. В. Белова – М.: Издательский центр «Академия», 2013. – 352 с.
5. Орлов, С. А. Технологии разработки программного обеспечения. Современный курс по программной инженерии: Учебник для вузов / Орлов С. А., Цилькер Б. Я. – 4-е изд. – СПб.: Питер, 2012. – 608 с.

Статистический анализ блоков размеченного текста

Е. В. Уваров

Студент магистр

А. В. Акимов

Старший преподаватель

Введение

Цифровизация современного общества и стремительное развитие искусственного интеллекта внесли свой вклад в методы и приемы работы с текстом. Возникла необходимость в ускорении процесса анализа текста с учетом тенденций использования гибридных методов научного исследования в интеграции с информационными технологиями. Несмотря на появление качественных и количественных подходов в гуманитарных исследованиях, основная проблема заключается в отсутствии базовых образовательных материалов по компьютеризированному анализу текстов.

В данной статье рассматривается задача маркировки массивов текста для их последующего статистического анализа. Задача маркировки текста состоит в том, чтобы разметить текст на произвольные фрагменты и дать возможность специалисту без технической подготовки провести статистический анализ размеченных фрагментов. Сложность задачи заключается в эффективной маркировке исходного текста и дальнейшем сопоставлении полученных статистических данных и соответствующих им блоков размеченного текста. Результатом данной работы является универсальный метод, который позволяет эффективно разметить текстовый документ и провести его статистический анализ, независимо от технических навыков специалиста и используемого языка программирования.

Исходными данными для решения задачи являются строки, полученные из текстовых файлов, выходными данными являются размеченные на фрагменты строки и массивы данных, полученные при статистическом анализе маркированных фрагментов исходного текста.

1. Методы и средства статистического анализа текста

Под статистикой понимают количественный учет некоего явления в некотором пространстве. Под статистическим пространством понимают

фиксированную совокупность неких объектов. При статистическом анализе важно определить статистическое пространство и искомое событие с устойчивыми характеристиками как константами.

Сегодня наиболее актуальными являются следующие методы статистического анализа текста: процедуры количественных исследований [1], частотный анализ [2], контент-анализ [3], ранжирование данных [4], закон Ципфа [5]. Данные методы используются при формировании библиотечных фондов и поиске информации по узкой тематике. Ранжирование данных часто применяется при организации информационных массивов в сети для работы поисковых систем.

Для реализации статистического анализа текста используют программные средства и алгоритмы, реализованные на таких языках программирования как Python, C#, C++ и др. К программным средствам можно отнести «АВАК-БОТ» [6], «ISTIO» [7] и др.

В настоящее время в сегменте анализа текстов на русском языке наблюдается ощутимое отставание от английского и других языков, а ряд математических методов позволяет реализовать только часть работ по разметке текстов. Чтобы подробно рассмотреть методы и средства разметки текста для анализа необходимо обратиться к Национальному корпусу русского языка [8].

Главной характеристикой корпуса является разметка. На ранних этапах моделирования текста с помощью разметки решаемые задачи полностью формализуемы, но углубленный анализ требует не только использования интеллектуальных средств, но и привлечения человека к решаемой задаче. Анализ текста на естественном языке предполагает многоуровневую разметку текста. Таким образом, размеченный текст, полученный на одном уровне анализа, используется как исходный для разметки текста на более глубоком уровне.

Глубоко аннотированный корпус [9] – фрагмент Национального корпуса русского языка, который содержит тексты с морфосинтаксической разметкой. На различных стадиях работы над корпусом были введены дополнительные типы разметки: лексико-семантическая, лексико-функциональная, анафорическая и микросинтаксическая.

Морфосинтаксическая разметка предложения включает в себя морфологический разбор каждого слова, а также синтаксическую структуру предложения в виде дерева зависимостей согласно модели «Смысл \Leftrightarrow Текст» [10]. Лексико-семантическая разметка предполагает, что для каждого слова указана соответствующая ему статья толково-комбинаторного словаря русского языка. Лексико-функциональная

разметка представляет собой выделение в текстах словосочетаний, допускающих интерпретацию в терминах лексических функций. Результатом анафорической разметки является маркирование антецедентов местоимений. Микросинтаксическая разметка идентифицирует встретившиеся в текстах синтаксические фраземы и некоторые нестандартные синтаксические конструкции.

Разметка новых текстов корпуса выполняется в несколько этапов. Сначала тексты обрабатываются лингвистическим процессором «ЭТАП-4» [11], который вносит в них морфосинтаксическую и лексико-семантическую разметку. Затем результаты работы процессора проверяются и при необходимости исправляются квалифицированными лингвистами. Далее на основе построенных морфосинтаксических структур «ЭТАП-4» выполняет лексико-функциональную и анафорическую разметку. После этого лингвисты проверяют и корректируют эти типы разметки и вручную вносят в тексты микросинтаксическую разметку.

Однако описанный выше подход не позволяет эффективно разметить текстовый документ на произвольные фрагменты и провести их последующий статистический анализ. Для решения этой задачи предлагается метод маркировки фрагментов исходного текста с использованием простого и понятного синтаксиса.

2. Средства для работы с текстом в MATLAB

MATLAB [12] – язык программирования, используемый для решения задач технических вычислений. Для работы с текстом в MATLAB используется программный пакет Text Analytics Toolbox [13].

Text Analytics Toolbox предоставляет алгоритмы и средства визуализации для предварительной обработки текстовых данных и их дальнейшего анализа и моделирования. Построенные с помощью Text Analytics Toolbox модели могут быть использованы в таких областях как анализ тональности [14], предиктивное обслуживание [15] и тематическое моделирование [16].

Text Analytics Toolbox содержит инструменты для обработки исходного текста из различных источников, включая новостные ленты, социальные сети, журналы учета эксплуатации оборудования, отчеты и исследования. С помощью программного пакета Text Analytics Toolbox можно извлечь отдельные слова, преобразовать текстовые данные в числовой вид или создать статистическую модель.

Методы машинного обучения, такие как Латентно-семантический анализ [17] и Латентное размещение Дирихле [18], можно использовать вместе с векторным представлением слов [19], чтобы находить кластеры и создавать функции на основе высокоразмерных наборов текстовых

данных. Такие функции можно комбинировать с функциями на базе других источников данных для создания моделей машинного обучения, способных использовать текстовые, числовые и другие типы данных.

Хотя большинство функций программного пакета Text Analytics Toolbox поддерживает русский язык, в полной мере поддерживаются только английский, японский, немецкий и корейский языки.

Несмотря на все удобство и функциональность программного пакета Text Analytics Toolbox, представленного в MATLAB, вопрос доступности для исследователей и лингвистов без технической подготовки остается открытым. Способствует этому и отсутствие готовых решений для расчета статистики в произвольных фрагментах анализируемого текста.

3. Применение синтаксиса для разметки блоков текста и их анализ

Для решения задачи маркировки массивов текста для их последующего статистического анализа предлагается ввести простой и понятный синтаксис для обозначения блоков текста, представляющих интерес для исследования.

Пример разметки фрагментов текста представлен на рис. 1.

```
{block First}MATLAB как язык программирования был разработан  
Кливом Моулером в конце 1970-х годов, когда он был деканом  
факультета компьютерных наук в Университете Нью-  
Мексико. {endblock} {block Second}Целью разработки служила задача  
дать студентам факультета возможность использования программных  
библиотек Linpack и EISPACK без необходимости изучения  
Фортрана. {endblock}
```

Рис. 1. Пример синтаксиса разметки текста на блоки

Синтаксическими конструкциями {block First} и {block Second} маркируется начало фрагмента текста, а конструкцией {endblock} – его окончание. Таким образом, независимо от технических навыков исследователь самостоятельно может разметить текст на необходимые для исследования фрагменты и провести их поблочный анализ с помощью предложенных ниже решений.

Для дальнейшего анализа необходимо извлечь из текста два массива данных массив, состоящий из размеченных блоков текста и массив, состоящий из имен блоков.

Функция, извлекающая отдельно поблочный массив строк текста и имена соответствующих блоков представлена на рис. 2.

```

function [textBody, textDiv] = stringDiv1(xlabeldoc)
str = extractFileText(xlabeldoc);
startPat = "{block ";
endPat = "{endblock}";
newStr = extractBetween(str, startPat, endPat);
textBody = extractAfter(newStr, " ");
textDiv = extractBefore(newStr, " ");
end

```

Рис. 2. Функция, которая извлекает массив с блоками текста и массив с именами блоков

Функция, описанная выше, принимает в качестве аргумента путь к файлу с текстовыми данными и возвращает массив с размеченными фрагментами исходного текста и массив с именами блоков.

В качестве примера для демонстрации вышеописанного подхода предлагается использовать ряд функций для подсчета статистики текстовых фрагментов.

4. Тестирование разработанного подхода

Предварительное тестирование разработанного подхода производилось на примере текстовых данных, взятых из поэмы «Мертвые души» Николая Васильевича Гоголя. Текст был предварительно размечен на блоки, соответствующие одиннадцати главам произведения. Затем с помощью функции, принимающей в качестве аргумента путь к файлу, были получены два массива размерностью 11x1, состоящие из текстовых данных блоков-глав на рис. 3 и соответствующих им имен на рис. 4 соответственно.

textBody = 11×1 string array

```

"В ворота гостиницы губернского города NN въехала довольно красивая рессорная..."
"Уже более недели приезжий господин жил в городе, разъезжая по вечеринкам..."
"А Чичиков в довольно расположении духа сидел в своей бричке..."
"Подъехавши к трактиру, Чичиков велел остановиться по двум причинам..."
"Герой наш трухнул, однако ж, порядком. Хотя бричка мчалась во всю пропащую..."
"Прежде, давно, в лета моей юности, в лета невозвратно мелькнувшего..."
"Счастлив путник, который после длинной, скучной дороги с ее холодами..."
"Покупки Чичикова сделались предметом разговоров. В городе пошли толки..."
"Поутру, ранее даже того времени, которое назначено в городе N..."
"Собравшись у полицеймейстера, уже известного читателям отца и благодетеля..."
"Ничто, однако же, не случилось так, как предполагал Чичиков. Во-первых..."

```

Рис. 3. Массив текстовых данных блоков-глав

```
textDiv = 11 × 1 string array
```

```
"Глава первая"  
"Глава вторая"  
"Глава третья"  
"Глава четвертая"  
"Глава пятая"  
"Глава шестая"  
"Глава седьмая"  
"Глава восьмая"  
"Глава девятая"  
"Глава десятая"  
"Глава одиннадцатая"
```

Рис. 4. Массив имен блоков

Для наглядности метода была использована функция для подсчета количества печатных знаков в каждом блоке текста с предварительным созданием пустого массива и его последующего заполнения статистическими данными в процессе реализации цикла. В результате был получен массив размерностью 11x1, состоящий из количественных данных, соответствующих каждому блоку. Пример представлен на рис. 5.

```
statsAllCharacters =
```

```
21274  
34985  
35130  
45018  
35260  
39473  
36561  
43801  
33868  
35269  
58811
```

Рис. 5. Массив количественных данных

Заключение

В данной статье был предложен подход для реализации маркировки массивов текста для их последующего статистического анализа. Описан процесс разметки текстовых данных и приведены примеры их статистического анализа. Использование данного решения позволяет специалисту без технических навыков эффективно разметить текст на произвольные фрагменты и провести их дальнейший независимый анализ.

Список литературы

1. Хоменко, А. Ю. Лингвистическое атрибуционное исследование коротких письменных текстов: качественные и количественные методы / А. Ю. Хоменко // Политическая лингвистика. – 2019. – № 2 (74). – С. 177-187.
2. Шумилина, Т. В. Применение частотного анализа текстов СМИ для оптимизации процесса коммуникации / Т. В. Шумилина // Вестник Московского университета. Сер. 10. Журналистика. – 2017. – № 2 – С. 67-79.
3. Чернобровкина, Е. П. Контент-анализ в лингвистических исследованиях / Е. П. Чернобровкина // Вестник Бурятского государственного университета. – 2011. – № 11 – С. 125-129.
4. Змеева, Н. Б. Ранжирование учебных текстов методом экспертных оценок / Н. Б. Змеева // Проблемы современной науки и образования. – 2016. – № 04 (46). – С. 132-137.
5. Манин, Ю. И. Закон Ципфа и вероятностные распределения Левина / Ю. И. Манин // Функциональный анализ и его приложения. – 2014 – Т. 48, № 2 – С. 51-66.
6. АВАК-БОТ: сборник калькуляторов [Электронный ресурс]. – URL: <https://abakbot.ru/online-5/97-freq-letter/> (дата обращения 28.04.2021).
7. ISTIO: SEO анализ текста онлайн [Электронный ресурс]. – URL: <https://istio.com/> (дата обращения 17.05.2021).
8. Национальный корпус русского языка: информационно-справочная система [Электронный ресурс]. – URL: <https://ruscorpora.ru/new/> (дата обращения 17.05.2021).
9. Иншакова, Е. С. СинТагРус сегодня / Е.С. Иншакова, Л.Л. Иомдин, Л.Г. Митюшин, В.Г. Сизов, Т.И. Фролова, Л.Л. Цинман // Труды Института русского языка им. В.В. Виноградова. М. – 2019. – Вып. 21 – С. 14-40.
10. Мельчук, И. А. Опыт теории лингвистических моделей «Смысл – Текст» / И. А. Мельчук – Школа «Языки русской культуры». – 1999.
11. ЭТАП-4 с интерфейсом командной строки: Лаборатория компьютерной лингвистики [Электронный ресурс]. – URL: <http://proling.iitp.ru/ru/etapbare/> (дата обращения 17.05.2021).
12. MathWorks: MATLAB documentation [Электронный ресурс]. – URL: <https://www.mathworks.com/help/matlab/index.html/> (дата обращения 17.05.2021).

13. MathWorks: Text Analytics Toolbox documentation [Электронный ресурс]. – URL : <https://www.mathworks.com/help/textanalytics/index.html/> (дата обращения 17.05.2021).

14. Романов, А.С. Анализ тональности текста с использованием методов машинного обучения / А.С. Романов, М.И. Васильева, А.В. Куртукова, Р.В. Мещеряков // Proceedings 2nd International Conference "R. PIOTROWSKI'S READINGS LE & AL'2017" (Saint-Petersburg, 2017). Санкт Петербург : М. Jeusfeld c/o Redaktion Sun SITE, Informatik V. – 2018 – С. 86-95.

15. Власов, А. И. Модель предиктивного обслуживания оборудования с применением беспроводных сенсорных сетей / А. И. Власов, П. В. Григорьев, А. И. Кривошеин // Надежность и качество сложных систем. – 2018. – № 2(22). – С. 26-35.

16. Глушков, Н. А. Анализ методов тематического моделирования текстов на естественном языке / Н. А. Глушков // Молодой ученый. – 2018. – № 19 (205). – С. 101-103.

17. Бондарчук, Д.В. Использование латентно-семантического анализа в задачах классификации текстов по эмоциональной окраске / Д.В. Бондарчук // Бюллетень результатов научных исследований. – 2012. – № 2(3). – С. 146–152.

18. Митрофанова, О. А. Моделирование семантических связей в текстах социальных сетей с помощью алгоритма LDA (на материале русскоязычного сегмента Живого Журнала) / Митрофанова О. А., Шиморина А.С., Кольцова О.Ю., Кольцов С.Н. // Структурная и прикладная лингвистика. – 2014. – Вып. 10 – С. 151-168.

19. Лыченко, Н. М. Сравнение эффективности методов векторного представления слов для определения тональности текстов / Н. М. Лыченко, А. В. Сороковая // Математические структуры и моделирование. – 2019. – № 4(52). – С. 97-110.

Анализ использования методологии «Дизайн-мышление» при проектировании программного обеспечения

И. А. Фирсова

Студент магистр

В. В. Таратухин

Профессор

Введение

В настоящее время развиваются различные способы доступа к опыту пользователей. Традиционные исследования и методы дизайнеров были сосредоточены в основном на наблюдательных исследованиях (т. е. изучении того, что люди делают и используют).

С другой стороны, традиционные методы исследования рынка больше ориентированы на то, что говорят и думают люди (через фокус-группы, интервью и анкеты). Новые инструменты ориентированы на то, что люди делают, т. е. что они создают из наборов инструментов, которые мы им предоставляем для использования, как они выражают свои мысли, чувства и мечты. Многие из этих инструментов основаны на невербальных способах выражения

UX – это User Experience (дословно: «опыт пользователя»). То есть это то, какой опыт/впечатление получает пользователь от работы с вашим интерфейсом. Удастся ли ему достичь цели и на сколько просто или сложно это сделать.

Предлагается определить UX как нечто индивидуальное (а не социальное), возникающее в результате взаимодействия с продуктом, системой, услугой или объектом [1]. Результаты исследования закладывают основу для понимания, анализа и определения концепции пользовательского опыта. Огромный интерес к UX в академических кругах и промышленности можно объяснить тем фактом, что исследователи хорошо осознали ограничения традиционной структуры юзабилити, которая фокусируется в первую очередь на познании пользователя и производительности пользователя при взаимодействии человека и технологии. Данная работа посвящена разработке новых принципов этапа тестирования ПО в рамках методологии «Дизайн-мышление».

1. Сложности в определении UX

Понятие пользовательского опыта (UX) было широко распространено и быстро принято в сообществе взаимодействия человека и компьютера, однако оно не было четко определено или хорошо понято [2]. UX подчеркивает влияние пользователя, ощущение и значение, а также ценность таких взаимодействий в повседневной жизни. Следовательно, UX рассматривается как нечто желательное, хотя то, что именно означает, остается открытым и спорным.

Возникает непреодолимый вопрос: почему так сложно достичь общего определения UX?

Существует несколько причин, по которым трудно получить универсальное определение UX [3].

Во-первых, UX ассоциируется с широким спектром нечетких и динамических понятий, включая эмоциональные, аффективные, эмпирические, гедонические и эстетические переменные.

Во-вторых, единица анализа для UX слишком податлива, начиная от одного аспекта взаимодействия отдельного конечного пользователя с автономным приложением и заканчивая всеми аспектами взаимодействия нескольких конечных пользователей с компанией и ее слиянием услуг из нескольких дисциплин.

В-третьих, ландшафт исследований UX фрагментирован и осложнен разнообразными теоретическими моделями с различными фокусами, такими как прагматизм, эмоции, аффект, опыт, ценность, удовольствие, красота, гедонистические качества и т. д.

2. Жизненный цикл разработки Юзабилити

Жизненный цикл, состоящий из нескольких фаз, представлен на рис. 1. Основными фазами являются:

1. Исследование пользователей
 - наблюдение за пользователями и интервью;
 - классификация пользователей в соответствии с их характеристиками;
 - составление профиля пользователя для каждого класса пользователей на основе поведенческих и демографических переменных;
 - определение целей и установок пользователей;
 - анализ рабочего процесса и контекста работы;
 - составление набора типичных пользовательских сценариев.
2. Бенчмаркинг юзабилити:
 - анализ конкурирующих продуктов или интерфейсов эвристически и эмпирически;

– установление измеримых целей удобства использования для интерфейса.

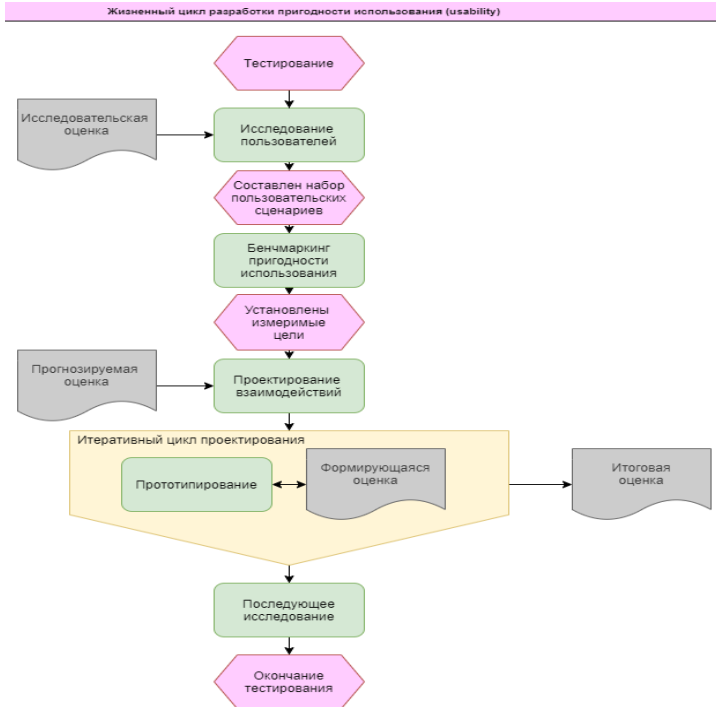


Рис. 1. Жизненный цикл разработки Юзабилити

3. Проектирование взаимодействия:
 - целенаправленное начальное проектирование интерфейса.
4. Итеративный цикл проектирования:
 - проектирование, тестирование, редизайн;
 - построение и оценка прототипа интерфейса;
 - оценка серьезности обнаруженных проблем юзабилити;
 - исправить проблемы → новая версия интерфейса;
 - обоснование дизайна: записать причины, по которым были внесены изменения;
 - оценить новую версию интерфейса.
- 4.1. Создание Прототипов:
 - словесное описание;
 - бумажный прототип;

- интерактивный прототип;
- рабочий прототип;
- реализация окончательного проекта.

4.2. Методы оценки удобства использования сгруппированы в зависимости от того, кто их выполняет:

- методы проверки удобства использования;
- методы юзабилити-тестирования.

5. Последующие исследования.

Важные данные об удобстве использования могут быть собраны после выпуска продукта для следующей версии:

- конкретные исследования (интервью, анкетирование, наблюдение);
- стандартные маркетинговые исследования (что говорят люди в группах новостей и списках рассылки, обзоры и тесты в журналах и т. д.);
- анализ жалоб пользователей на горячую линию, запросов на модификацию, сообщений об ошибках;
- исследования более длительного использования продукта;
- ведение журнала программного обеспечения: инструментальные версии программного обеспечения → данные журнала.

3. Проведение опроса среди тестировщиков.

В рамках анализа использования методологии «Дизайн мышление» был проведён опрос среди тестировщиков о их знаниях и умениях работать в рамках данной методологии. Была сделана выборка из 16 человек, работающих в разных IT- кампаниях, на разных должностях, с целью выявить популярность данной методологии и понимания сотрудниками принципов её работы.

Среди опрошиваемых 87.5% знали, что такое UX-дизайн. Диаграмма представлена на рис. 2. Следовательно, данная группа имеет опыт работы с юзабилити тестированием.

Знаете ли Вы, что такое UX - дизайн?
16 ответов

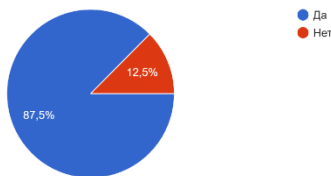


Рис. 2. Диаграмма о знания UX – дизайна

Так же были получены ответы на вопрос «Используется ли методология «Дизайн-мышление» на вашей работе». 50% ответили «Да», 31.3% ответили «Нет» и 18.8% ответили «Нет, но хотел бы». Следовательно большая часть опрашиваемых заинтересованы в этой методологии и понимают принципы работы.

Используется ли методология "Дизайн - мышление" на вашей работе?
168/189;ответов

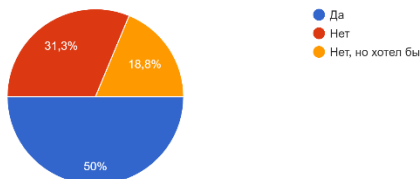


Рис. 3. Диаграмма использования методологии «Дизайн-мышление»

Заключение

В данной статье были рассмотрены фазы жизненного цикла разработки Юзабилити при проектировании программного обеспечения. Так же были рассмотрены и описаны сложности в определении UX в сообществе взаимодействия человека и компьютера. Был проведён опрос среди тестировщиков, и выявлена большая заинтересованность в методологии «Дизайн-мышление». В итоге проделанной работы мы имеем модель жизненного цикла Юзабилити и протестированную заинтересованную группу, на ответах которой будет проводиться дальнейшее исследование.

Список литературы

1. Bullinger H.-J. Human-Computer Interaction-INTERACT '87 / H.-J. Bullinger, B. Shackel – Elsevier Science Publishers B.V. – North-Holland. : IFIP, 1987. – 18 p.
2. Understanding, Scoping and Defining User eXperience: A Survey Approach/ Effie L-C. Law [и др.] // CHI User Experience. – 2009. P. 719-728.
3. Hassenzahl M. User experience – a research agenda / M.Hassenzahl, N.Tractinsky // Behaviour & Information Technology. – Darmstadt, 2006. – P. 91-96.

Методика оценки риска информационной безопасности систем обработки информации

Д. И. Хацкевич

Студент магистрант

В. Ю. Храмов

Профессор

Введение

Проблема обеспечения информационной безопасности приобретает в последние годы все большую остроту. Следствием этого является повышенное внимание общества и государства к созданию нормативной и законодательной базы, регламентирующей основные вопросы проведения аудита и оценки защищенности информационных систем, разработки, внедрения и эксплуатации систем защиты информации. За последние 10-15 лет создана обширная система национальных стандартов информационной безопасности, охватывающих общие вопросы анализа и управления информационными рисками.

Существует достаточно большое число методов и алгоритмов, позволяющих получить оценку уровня риска [1]. Важное место среди методов, направленных на получение качественной оценки рисков, занимают методы когнитивного моделирования, предназначенные для исследования плохо формализуемых ситуаций и проблем путем построения нечетких когнитивных карт (Fuzzy Cognitive Maps) [2]. Данный класс моделей включает в себя обобщенные НКК, реляционные НКК, интервальные («серые») НКК, продукционные НКК [3] и многие другие модификации НКК. В основе построения этих моделей оценки рисков используется так называемая трехфакторная формула риска:

$$\text{Риск} = \text{Угроза} \cdot \text{Уязвимость} \cdot \text{Ценность}_\text{ресурса} \quad (1)$$

характеризующая происхождение и основные составляющие формирования риска. Данная работа посвящена разработке методики оценки риска информационной безопасности и рассмотрению особенностей ее применения для системы обработки информации, используя при этом один из перспективных классов когнитивных моделей – нечеткие продукционные когнитивные карты.

2. Нечеткие продукционные когнитивные карты

Под нечеткой продукционной когнитивной картой обычно понимается ориентированный граф (орграф), задаваемый парой множеств:

$$K = \{C, F\}, \quad (2)$$

где $C = \{C_i\}$ ($i = 1, 2, \dots, n$) – множество узлов (вершин) орграфа, называемых концептами; $F = \{F_i\}$ ($i, j = 1, 2, \dots, n$) – множество дуг – связей (отношений) между концептами; n – число концептов НПКК. Предполагается, что переменная состояния X_i каждого концепта C_i рассматривается как лингвистическая переменная, принимающая значения из некоторого нечеткого терм-множества $\{T_{i1}, T_{i2}, \dots, T_{im}\}$, подмножества (термы) которого T_{ik} ($k = 1, 2, \dots, m$), в свою очередь, задаются функциями принадлежности: $T_{ik} = \{(\mu_{ik}(X_i), X_i)\}$, $\mu_{ik}: X_i \rightarrow [0, 1]$, где $X_i \in [0, 1]$ или $X_i \in [-1, 1]$. Различают два вида концептов: уровни (levels), которые представляют абсолютные значения состояния концепта в данный момент времени, и вариации (variations), которые представляют изменения состояния концепта по отношению к предыдущему моменту времени. Последнее важно для описания динамики поведения исследуемых систем. Для определения взаимного влияния концептов ($C_i \rightarrow C_j$) используются нечеткие продукционные правила, позволяющие представить предпосылки (условия) и заключения нечетких правил на основе нечетких множеств.

3. Алгоритм логического вывода

Для реализации процедуры нечеткого логического вывода (применительно к конкретным «четким» значениям входных переменных X_i и X_j концепта C_k и получению фазсифицированного значения выходной переменной X_k концепта C_k) был разработан следующий алгоритм (рис. 1):

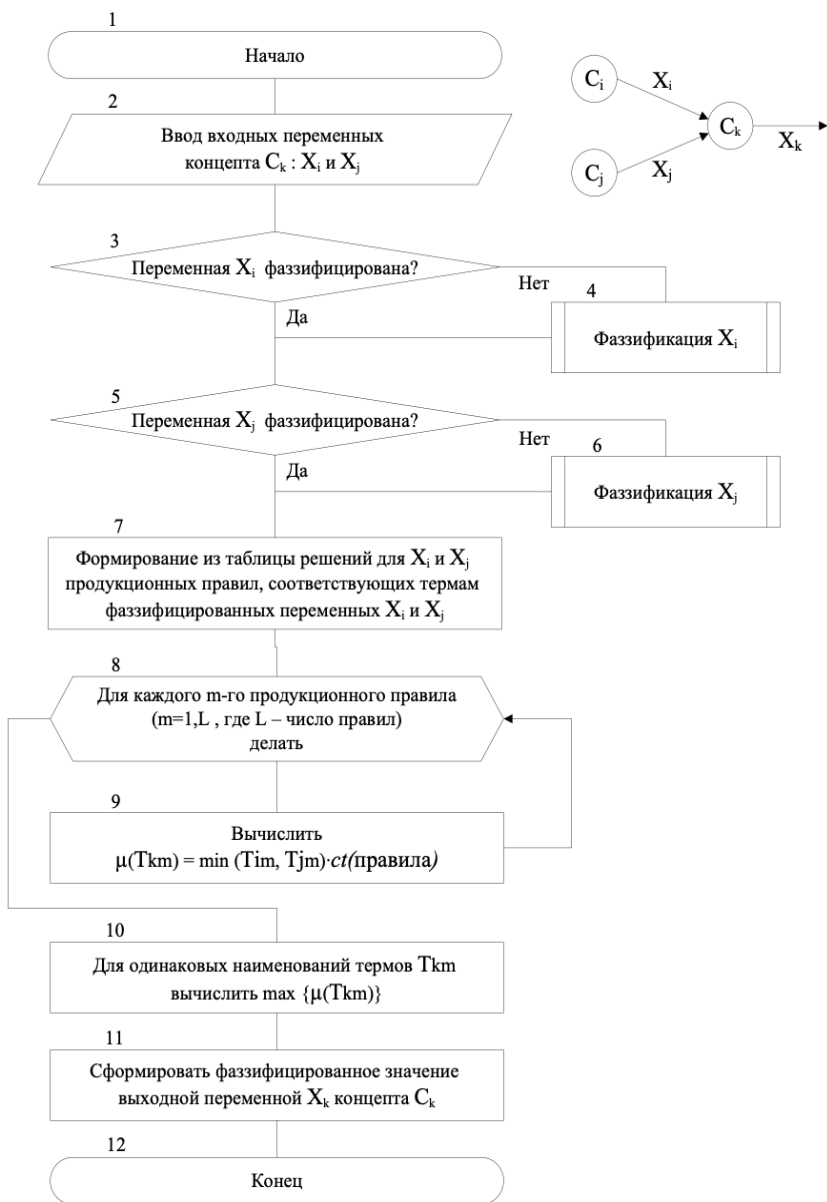


Рис. 1. Блок-схема алгоритма

4. Методика оценки риска информационной безопасности и пример ее реализации

Методика оценки рисков информационной безопасности строится с использованием нечетких продукционных когнитивных карт (НПКК), функций принадлежности (ФП) нечетких множеств и матриц риска, а также нечеткого логического вывода с использованием алгоритма, основанного на теореме гипотез (формуле Байеса). Разработанная методика включает пять шагов.

Допустим, что требуется оценить риск возможного воздействия вирусной атаки на некоторый информационный ресурс, размещаемый на сервере, рассматривая в качестве уязвимости отсутствие обновлений антивирусного ПО.

Шаг 1. Построение нечеткой продукционной когнитивной карты.

Возвращаясь к упомянутой выше трехфакторной формуле риска (1), представим соответствующую ей схему расчета в виде НПКК на рис. 2, где C_1 – угроза, C_2 – уязвимость, C_3 – информационный ресурс, C_4 – реализация угрозы, C_5 – риск (потенциальный ущерб); соответственно X_1 – вероятность возникновения угрозы, X_2 – вероятность наличия уязвимости, X_3 – ценность информационного ресурса, X_4 – вероятность успешной реализации угрозы, $X_5 = R$ – уровень риска (значение ожидаемого потенциального ущерба).

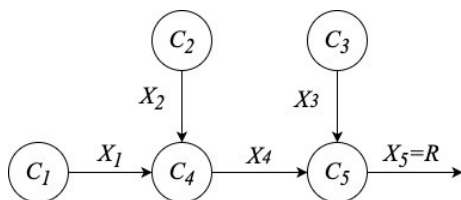


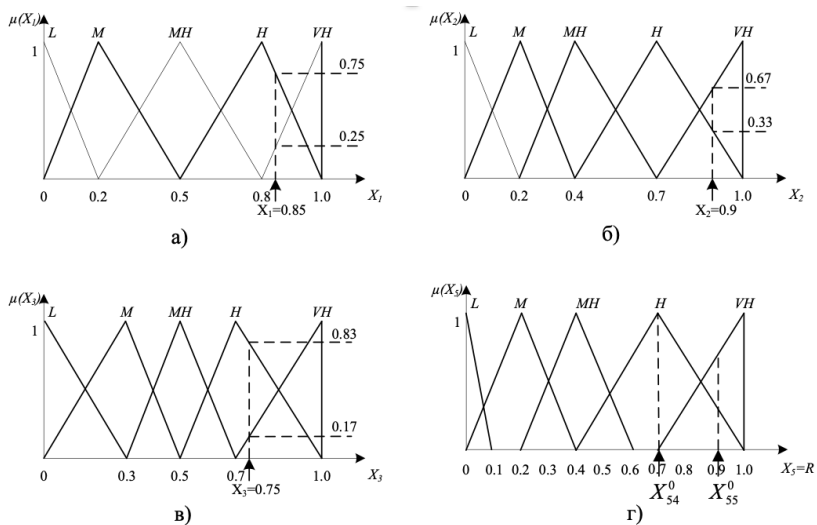
Рис. 2. Схема НПКК для оценки риска

Шаг 2. Построение ФП для термов лингвистических переменных, характеризующих переменные состояния концептов НПКК.

Каждая из указанных переменных состояния представляет собой лингвистическую переменную, принимающую одно из следующих значений: L – Low (Низкая (-ий)); M – Medium (Средняя (-ий)); MH – Medium High (Довольно высокая (-ий)); H – High (Высокая (-ий)); VH – Very High (Очень высокая (-ий)). Три входных и конечное нечеткое подмножество задаются, в свою очередь, собственной функцией

принадлежности, построенной экспертным путем, вид которых показан на рис. 3.

Допустим, что входные переменные НПКК принимают следующие значения: $X_1^* = 0,85$, $X_2^* = 0,9$, $X_3^* = 0,75$.



а – угроза C_1 , *б* – уязвимость C_2 , *в* – ценность ресурса C_3 , *г* – риск C_5

Рис. 3. Функции принадлежности нечетких множеств

Шаг 3. Построение системы нечетких продукционных правил (матриц риска), описывающих состояние концептов C_k .

Правила записываются в форме продукций: ЕСЛИ a , ТО b , где a – условие, b – заключение. Вычисление коэффициента определенности заключения $ct(b)$ осуществляется по формуле:

$$ct(b) = ct(a) \cdot ct(\text{правила}), \quad (3)$$

где $ct(a)$ – коэффициент определенности условия ($ct(a) \in [0, 1]$), $ct(\text{правила})$ – коэффициент определенности правила ($ct(\text{правила}) \in [0, 1]$).

Систему нечетких продукционных правил, описывающих состояние концептов C_4 и C_5 , можно записать в виде:

концепт C_4 :

P_1 : Если X_1 есть Низкая и X_2 есть Низкая, то X_4 есть Низкая;

...

P_{25} : Если X_1 есть Очень_высокая и X_2 есть Очень_высокая, то X_4 есть Очень_высокая;

концепт C_5 :

P_{26} : Если X_3 есть Низкая и X_4 есть Низкая, то X_5 есть Низкая;

...

P_{50} : Если X_3 есть Очень_высокая и X_4 есть Очень_высокая, то X_5 есть Очень_высокая.

Полученные правила удобно представить в виде таблиц решений (матриц риска), приведенных на рис.4, $ct(правил) = 1$.

X_1	VH	MH	H	H	H	VH
	H	M	MH	H	H	H
	MH	M	M	MH	MH	H
	M	L	M	M	M	MH
	L	L	L	L	M	M

						X_3
VH	M	M	MH	H	VH	
H	L	M	MH	H	H	
MH	L	M	M	MH	MH	
M	L	L	M	M	M	
L	L	L	L	L	L	

X_2

L M MH H VH

а

X_4

L M MH H VH

б

а – реализация угрозы C_4 , *б* – риск C_5

Рис. 4. Матрицы риска

Используя рис. 3 и рис. 4, определяем, что переменные X_1 , X_2 , X_3 , X_4 принимают только значения H и VH .

Шаг 4. Нахождение значений промежуточных переменных состояния концептов и переменной риска по алгоритму, представленному на рис. 1.

Вычислим значение промежуточной переменной состояния X_4 концепта C_4 .

Продукционные правила для концепта C_4 имеют следующий вид:

1. Если $X_1 = H$ и $X_2 = H$, то $X_4 = H$;
2. Если $X_1 = H$ и $X_2 = VH$, то $X_4 = H$;
3. Если $X_1 = VH$ и $X_2 = H$, то $X_4 = H$;
4. Если $X_1 = VH$ и $X_2 = VH$, то $X_4 = VH$.

По функциям принадлежности определяем значения $\mu(X_1^*)$ и $\mu(X_2^*)$, которые соответственно равны $\langle\langle 0,75 / H \rangle, \langle 0,25 / VH \rangle\rangle$ и $\langle\langle 0,33 / H \rangle, \langle 0,67 / VH \rangle\rangle$.

Далее в соответствии с алгоритмом для каждого правила применяется операция MIN:

5. $\min(0,75; 0,33) = 0,33$;
6. $\min(0,75; 0,67) = 0,67$;
7. $\min(0,25; 0,33) = 0,25$;
8. $\min(0,25; 0,67) = 0,25$.

Так как $ct(\text{правил}) = 1$, то полученные значения не изменяются.

Для одинаковых наименований термов продукционных правил, стоящих в заключении продукций (H (Высокая) и VH (Очень_высокая)) применяется операция MAX:

- $\max(0,33; 0,67; 0,25) = 0,67$;
- $\max(0,25) = 0,25$.

Таким образом, переменная состояния X_4 концепта C_4 принимает следующее нечеткое значение:

$\langle\langle 0,67 / H \rangle, \langle 0,25 / VH \rangle\rangle$ / вероятность успешной реализации угрозы>.

Аналогичным образом осуществляется вычисление переменной состояния X_5 концепта C_5 (значение потенциального ущерба):

$\langle\langle 0,67 / H \rangle, \langle 0,17 / VH \rangle\rangle$ / потенциальный ущерб>.

Шаг 5. Вычисление значения выходной переменной риска по формуле метода взвешенного среднего:

$$X_5^* = X_R = \frac{\sum_{i=1}^m \alpha_i X_{Ri}^0}{\sum_{i=1}^m \alpha_i} \quad (4)$$

где X_{Ri}^0 ($i = 1, 2, \dots, m$) – центральные значения (центр тяжести) нечетких подмножеств (термов) переменной X_R , α_i – уровень активности i -го правила – значение степени принадлежности i -го термина переменной X_R , m – число термов лингвистической переменной X_R .

Центральные значения переменных риска X_{55}^0 (Очень высокий) и X_{54}^0 (Высокий) вычисляются с помощью интегрального исчисления как центры тяжести фигур, которые образуют графики функций соответствующих термов. Таким образом, получаем: $X_{55}^0 = 0.91$, $X_{54}^0 = 0.7$.

Применяя формулу (4), вычисляем дефаззифицированное значение переменной $X_R = 0.74$. Следовательно, искомое значение ожидаемого потенциального ущерба от действия угрозы составит 74 %.

Заключение

В данной статье на конкретном примере реализации вирусной атаки на информационный ресурс рассмотрены особенности применения разработанной методики оценки риска информационной безопасности.

В основе построения когнитивной модели используется описание взаимодействия между концептами, образующими НПКК, с помощью системы нечетких правил (продукций), отражающих знания и опыт экспертов в данной предметной области.

К числу преимуществ предложенного подхода к оценке риска, помимо наглядности и учета факторов неопределенности, относятся также гибкость и универсальность использования НПКК, заключающиеся в возможности расширения перечня учитываемых угроз, уязвимостей, защищаемых информационных ресурсов, а также категорий оценки рисков по видам ущерба.

Список литературы

2. Петренко С. А. Управление информационными рисками. Экономически оправданная безопасность. / С. А. Петренко, С. В. Симонов – М.: ДМ К Пресс, 2005. – 384 с.
3. Борисов В. В. Нечеткие модели и сети. / В. В. Борисов, В. В. Круглов, А. С. Федулов – Изд. 2-е, стереотип. – М.: Горячая линия – Телеком, 2012. – 284 с.
4. Борисов В. В. Анализ динамики состояния сложных систем на основе обобщенных нечетких продукционных когнитивных карт / В. В. Борисов, А. С. Федулов, Е. С. Устиненков // Нейрокомпьютеры: разработка, применение. – 2007. – № 1. – С. 17-23.

Применение нейронных сетей для идентификации осетровых рыб

Ю. Г. Хозиев

Студент магистр

П. С. Лысачев

Старший преподаватель

Введение

Сегодня современные технологии проникают во многие традиционные отрасли промышленности, например, рыбководство. Известный ихтиолог Сергей Борисович Подушка изобрел способ прижизненного получения икры у осетровых – до него рыб, из которых извлекали икру, убивали. Однако, для этого метода требуется идентификация каждой особи. Сейчас для идентификации в рыбных хозяйствах используются RFID-метки, но этот способ плохо зарекомендовал себя. Например, метки иногда вместе с икрой попадают на тарелки людям. Для нового метода предполагается разработать систему распознавания и идентификации рыб по их пигментным пятнам на основе нейронных сетей.

В рамках данной работы будет рассмотрена возможность применения нейронных сетей для идентификации осетровых рыб.

Для решения поставленной задачи была выбрана сверточная нейронная сеть Resnet18. Это классическая нейронная сеть, используемая в качестве основы для многих задач компьютерного зрения. Программа, реализующая данную сеть, будет написана на языке программирования Python [1] с использованием некоторых библиотек, основной из которых является PyTorch [2]. Данная библиотека предоставляет большое количество алгоритмов для глубокого обучения. Средой для разработки программы выступает Microsoft Visual Studio 2019. Также был собран и размечен набор изображений осетровых рыб или набор данных, на основе которого нейронная сеть будет обучаться.

1. О нейронной сети

Искусственная нейронная сеть – это математическая модель, а также ее программные или аппаратные реализации, построенные в некотором смысле по образу и подобию сетей нервных клеток живого

организма. Благодаря такой структуре, машина обретает способность анализировать и даже запоминать различную информацию. Нейронные сети также способны не только анализировать входящую информацию, но и воспроизводить ее из своей памяти.

Для данной задачи классификации используются сверточная нейронная сеть Resnet18, разработанная компанией Microsoft [3]. Модель архитектуры сети изображена на рис. 1.

Layer Name	Output Size	ResNet-18
conv1	$112 \times 112 \times 64$	$7 \times 7, 64, \text{stride } 2$
		$3 \times 3 \text{ max pool, stride } 2$
conv2_x	$56 \times 56 \times 64$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$
conv3_x	$28 \times 28 \times 128$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$
conv4_x	$14 \times 14 \times 256$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$
conv5_x	$7 \times 7 \times 512$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$
average pool	$1 \times 1 \times 512$	$7 \times 7 \text{ average pool}$
fully connected	1000	$512 \times 1000 \text{ fully connections}$
softmax	1000	

Рис. 1. Описание архитектуры сети Resnet18

На вход подается изображение размером 224 на 224 пикселя. В начале оно проходит через сверточные слои. Над ним несколько раз производятся операции свёртки или convolution, суть которой в том, что каждый фрагмент изображения умножается на матрицу (ядро) свёртки поэлементно, а результат суммируется и записывается в аналогичную позицию выходного изображения. Веса ядра свёртки являются обучаемыми параметрами. Изображение можно представить в виде трёхмерной матрицы, так как, кроме длины и ширины, у него могут быть каналы или слои, например, красный, зеленый и синий. Применение операции свёртки уменьшает изображение, но увеличивает число каналов [4]. Пример операции свёртки показан на рис. 2.

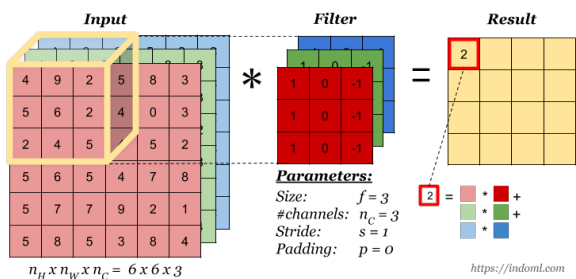


Рис. 2. Операция свертки

Также применяются функции пулинга, когда группа пикселей (например, размером 3 на 3) проходит через нелинейное преобразование и уплотняется до 1 значения. Чаще всего используется функция максимума (max pooling) или среднего (average pooling) [5]. Пример операции average pooling представлен на рис. 3.

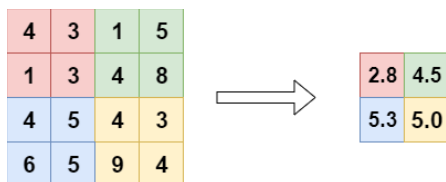


Рис. 3. Операция average pooling

В итоге, изображение сводится к вектору с 512 признаками. Они образуют входной полносвязный слой из 512 нейронов. Количество нейронов на выходном слое зависит от количества классов, которые сеть должна определять (в нашем случае их три).

На каждый нейрон выходного слоя подаются значения из всех нейронов прошлого слоя. Значения на входе в нейрон умножаются на веса для каждого входа, суммируются, и к этой сумме добавляется значение смещения. Значения весов и смещения являются обучаемыми параметрами. Выходное значение нейрона определяется функцией активации. В нашем случае, используется нелинейная функция активации ReLU или Rectified linear unit, которая возвращает поданное на вход значение, если оно больше либо равно нулю, или число 0 при подаче на него отрицательного значения [6]. Данная формула имеет следующий вид:

$$f(x) = \begin{cases} 0, & x < 0, \\ x, & x \geq 0. \end{cases} \quad (1)$$

На выходе из сети логическая функция Softmax преобразует выходные данные к вероятности принадлежности изображения к одному из классов [7]. Она имеет следующий вид:

$$\sigma(y)_i = \frac{e^{y_i}}{\sum_{j=1}^N e^{y_j}} \quad (2)$$

После одного прохода, сеть делает предсказание о принадлежности картинки к одному из классов. Отношение правильно определенных картинок к их общему числу является точностью.

С использованием функции потерь или loss-функции рассчитываются потери путем сопоставления целевого (фактического) значения и значения, прогнозируемого нейронной сетью [8]. После расчёта функции потерь используется метод градиентного спуска для обновления весов таким образом, чтобы потери были сведены к минимуму. В качестве функции потерь выступает кросс-энтропия (3), которая описывается как:

$$H(p, q) = -\sum_{i=1}^N p(x_i) \log q(x_i) \quad (3)$$

Вся описанная выше последовательность действий представляет собой одну эпоху обучения. Необходимо выбрать такое количество эпох, чтобы в итоге точность предсказаний была максимальная, а среднее значение функции потерь – минимальной. При этом при слишком длительном обучении возникает эффект переобучения или overfitting, когда обучаемая модель хорошо распознает примеры из обучающего множества, но при этом не распознает или плохо распознает любые другие примеры, не участвовавшие в процессе обучения [9]. Поэтому необходимо выбрать оптимальное количество эпох. В данном случае, оптимальным числом эпох было выбрано 30.

Для обучения сверточной нейронной сети и проверки ее точности необходимо собрать большое количество изображений и разметить их. Для этого было сделано множество фотографий трех особей рыб одного вида из семейства осетровых. Из них было выбрано 100 штук для каждой особи и сформирован тренировочный набор данных для обучения сети. Также было взято еще 30 фотографий для тестового набора данных, по которому мы будем судить об итоговой точности классификации.

2. Обучение и проверка точности классификации

Далее запускается программа, написанная на языке программирования Python, которая реализует описанную выше сеть, ее

обучение и проверку точности классификации. В директории со скриптом находится тренировочный и тестовый наборы данных с изображениями рыб.

В начале работы программы тренировочный набор данных делится на тренировочную и валидационную подвыборки. Этап валидации или проверка на соответствие – важная часть процесса и является подтверждением того, что модель будет работать именно так, как это ожидалось. Подмножество валидации не зависит от цикла обучения и используется для генерации графика проверки. На стадии тренировки нейронная сеть будет обучаться, а на стадии валидации можно увидеть, насколько сеть хорошо работает, какую точность она показывает на тех данных, которых она не видела, чтобы понять, нет ли переобучения.

На обучение изображения подаются группами, батчами, по 10 штук. Изображения сжимаются до размера 300 на 224 пикселя, где потом случайно берется квадрат размером 224 пикселя. После чего они подаются на первый сверточный слой сети. Каждую эпоху рассчитываются точность и среднее значение функции потерь для тренировочного и валидационного наборов данных. В конце обучения выводятся графики изменения значений функции потерь и точности от количества проведенных эпох. Пример работы программы представлен на рис. 4 и рис. 5.

```
P:\Microsoft Visual Studio\Shared\Python37_64\python.exe
100%|██████████| 100/100 [00:00<00:00, 294.90it/s]
100%|██████████| 100/100 [00:00<00:00, 255.78it/s]
100%|██████████| 100/100 [00:00<00:00, 104.88it/s]
Количество изображений в тренировочной выборке: 250
Количество батчей: 25
Идет вычисление на cuda:0
Epoch 1/30:
100%|██████████| 25/25 [00:42<00:00, 1.72s/it]
train Loss: 0.7974 Acc: 0.6311
100%|██████████| 6/6 [00:06<00:00, 1.03s/it]
val Loss: 0.2768 Acc: 0.9000
Epoch 2/30:
100%|██████████| 25/25 [00:29<00:00, 1.17s/it]
train Loss: 0.3767 Acc: 0.8711
100%|██████████| 6/6 [00:05<00:00, 1.02it/s]
val Loss: 0.1668 Acc: 0.9500
Epoch 3/30:
100%|██████████| 25/25 [00:29<00:00, 1.17s/it]
train Loss: 0.2613 Acc: 0.9156
100%|██████████| 6/6 [00:05<00:00, 1.04it/s]
val Loss: 0.3166 Acc: 0.9167
Epoch 4/30:
52%|██████████| 13/25 [00:15<00:14, 1.18s/it]
```

Рис. 4. Процесс обучения сети

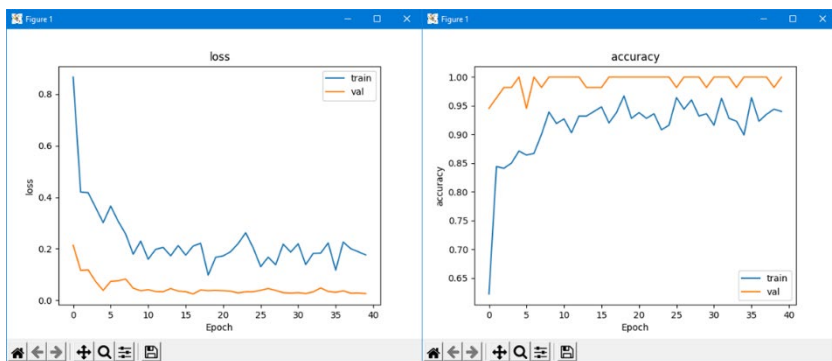


Рис. 5. Графики изменения значений функции потерь (loss) и точности (accuracy)

Далее обученная сеть классифицирует изображения из финальной тестовой выборки. Итоговые выводы записываются в файл формата csv, где каждое изображение соотносится с его предполагаемым классом. Как можно заметить по записям на рис. 6, у всех изображений класс был определен правильно.

answer.csv	
id	class
1	img_1648 (ostr1), ostr_1
2	img_1720 (ostr1), ostr_1
3	img_1777 (ostr2), ostr_2
4	img_1801 (ostr2), ostr_2
5	img_1842 (ostr2), ostr_2
6	img_1890 (ostr2), ostr_2
7	img_1944 (ostr2), ostr_2
8	img_2028 (ostr3), ostr_3
9	img_2101 (ostr3), ostr_3
10	img_2210 (ostr3), ostr_3
11	img_2250 (ostr1), ostr_1
12	img_2276 (ostr1), ostr_1
13	img_2292 (ostr1), ostr_1
14	img_2297 (ostr1), ostr_1
15	img_2314 (ostr1), ostr_1
16	img_2334 (ostr1), ostr_1
17	img_2383 (ostr1), ostr_1
18	img_2390 (ostr1), ostr_1
19	img_2420 (ostr2), ostr_2
20	img_2484 (ostr2), ostr_2
21	img_2499 (ostr2), ostr_2
22	img_2551 (ostr2), ostr_2
23	img_2553 (ostr2), ostr_2
24	img_2634 (ostr3), ostr_3
25	img_2661 (ostr3), ostr_3
26	img_2683 (ostr3), ostr_3
27	img_2704 (ostr3), ostr_3
28	img_2746 (ostr3), ostr_3
29	img_2786 (ostr3), ostr_3
30	img_2805 (ostr3), ostr_3
31	img_2805 (ostr3), ostr_3

Рис. 6. Классификация изображений из тестовой выборки

Заключение

В рамках данной статьи была рассмотрена возможность применения нейронных сетей для идентификации осетровых рыб.

Была написана программа, реализующая работу сверточной нейронной сети, которая показала высокую точность классификации

изображений. В итоге, удалось убедиться, что с помощью современных технологий компьютерного зрения и искусственных нейронных сетей можно с высокой точностью различать конкретные особи осетровых в рамках одного вида. Подобные решения могут быть использованы для создания более комплексной системы идентификации и учёта рыб для нужд рыбоводческой отрасли

Список литературы

1. Бэрри, П. Изучаем программирование на Python / П. Бэрри: Пер. с англ. – М.: Издательство «Э», 2017. – 624 с.
2. PYTORCH DOCUMENTATION [Электронный ресурс]. – Режим доступа: <https://pytorch.org/docs/stable/index.html>
3. RESNET DOCUMENTATION [Электронный ресурс]. – Режим доступа: https://pytorch.org/hub/pytorch_vision_resnet/
4. Сверточные нейронные сети [Электронный ресурс]. – Режим доступа: http://neerc.ifmo.ru/wiki/index.php?title=Сверточные_нейронные_сети
5. What are Max Pooling, Average Pooling, Global Max Pooling and Global Average Pooling? [Электронный ресурс]. – Режим доступа: <https://www.machinecurve.com/index.php/2020/01/30/what-are-max-pooling-average-pooling-global-max-pooling-and-global-average-pooling/>
6. Функции активации нейросети: сигмоида, линейная, ступенчатая, ReLu, tahn [Электронный ресурс]. – Режим доступа: <https://neurohive.io/ru/osnovy-data-science/activation-functions/>
7. Multi-Class Neural Networks: Softmax [Электронный ресурс]. – Режим доступа: <https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax>
8. Функции оценки, потеря, оптимизации – основы алгоритма Машинного Обучения [Электронный ресурс]. – Режим доступа: <https://id-lab.ru/posts/developers/funkcii/>
9. Сирота, А. А. Методы и алгоритмы анализа данных и их моделирование в MATLAB: учеб. Пособие / А. А. Сирота. – СПб.: БХВ-Петербург, 2016. – 384 с.

Поиск изоморфных алгебр Ли с помощью систем полиномиальных уравнений

О. А. Шеина

Студент магистр

А. В. Атанов

Доцент

Введение

В настоящее время задача поиска изоморфных алгебр Ли является одной из важнейших задач в различных областях современных наук: в рамках теории дифференциальных уравнений, в рамках теории дифференциальных уравнений с частными производными, а также в некоторых задачах комплексного анализа [1].

Установление наличия или отсутствия изоморфизма между алгебрами – достаточно сложная и нетривиальная математическая задача, которая может быть решена с использованием различных методов и подходов. Один из подходов предполагает нахождение изоморфных алгебр Ли на основе аналитических методов. Однако такой подход предполагает использование продвинутого математического аппарата теории групп и алгебр Ли, что в свою очередь усложняет процесс поиска. Другой метод позволяет найти изоморфную алгебру через систему полиномиальных уравнений, решения которой характеризуют коэффициенты матрицы перехода между базисами исследуемых алгебр. Такой подход хорош тем, что позволяет автоматизировать процесс нахождения изоморфных алгебр с помощью современных математических программ и пакетов.

В настоящей работе исследуется вопрос автоматического поиска изоморфных алгебр Ли с помощью систем полиномиальных уравнений.

1. Математические основы алгоритма нахождения изоморфных алгебр Ли

Введем ряд понятий и определений, используемых в настоящей работе. Векторное пространство L с операцией $L \times L \rightarrow L$, обозначаемой как $[x, y]$ и называемой коммутатором (или скобкой)

элементов x и y , называется алгеброй Ли, если выполняются следующие аксиомы:

- операция коммутирования билинейна;
- $[x, x] = 0$ для всех $x \in L$;
- $[x, [y, z]] + [y, [z, x]] + [z, [x, y]] = 0$ ($x, y, z \in L$).

Под билинейностью понимается билинейное отображение – функция, которая линейна по каждому из двух аргументов (например, матричное произведение). Две алгебры Ли L и L' будем называть изоморфными, если существует изоморфизм векторных пространств $\varphi: L \rightarrow L'$, удовлетворяющий следующему соотношению:

$$\varphi([x, y]) = [\varphi(x), \varphi(y)]$$

для всех $x, y \in L$ (и тогда φ называется изоморфизмом алгебр Ли).

Теперь перейдем к описанию математических основ алгоритма нахождения изоморфных алгебр Ли. Пусть \mathbb{F} – поле действительных (\mathbb{R}) или комплексных (\mathbb{C}) чисел, а L и \tilde{L} – алгебры Ли над полем \mathbb{F} размерности N , определенные своими структурными константами c_{ij}^k и

\tilde{c}_{ij}^k :

$$L: [X_i, X_j] = c_{ij}^k X_k, \quad \tilde{L}: [\tilde{Y}_i, \tilde{Y}_j] = \tilde{c}_{ij}^k \tilde{Y}_k,$$

где $\{X_i\}$ и $\{\tilde{Y}_i\}$ ($i=1, 2, \dots, N$) – базисы алгебр L и \tilde{L} соответственно. Здесь и везде далее мы будем использовать соглашение о суммировании по повторяющимся индексам. Предположим, что существует изоморфизм $\varphi: \tilde{L} \rightarrow L$, и обозначим образ базисного элемента \tilde{Y}_i как $Y_i := \varphi(\tilde{Y}_i)$. Множество $\{Y_i\}$ образует базис алгебры L со структурными константами \tilde{c}_{ij}^k :

$$[Y_i, Y_j] = \tilde{c}_{ij}^k Y_k. \quad (1)$$

Таким образом, существует базисное преобразование, представленное $N \times N$ -матрицей $A = (a_i^j)$, где $a_i^j \in \mathbb{F}$

$$Y_j = a_i^j X_j, \quad \det(A) \neq 0. \quad (2)$$

Если в левой части уравнения (1) заменить все Y на выражение из (2), то для левой части формулы (1) мы получим

$$[Y_i, Y_j] = a_i^k a_j^l [X_k, X_l] = a_i^k a_j^l c_{kl}^n X_n, \quad (3)$$

а для правой части уравнения (1)

$$[Y_i, Y_j] = c_{ij}^m a_m^n X_n. \quad (4)$$

Сравнение уравнений (3) и (4) показывает, что элементы матрицы преобразования должны удовлетворять соотношениям

$$a_i^k a_i^j c_{kl}^n - \tilde{c}_{ij}^m a_m^n = 0. \quad (5)$$

Для заданных структурных констант $\{c_{ij}^k\}$ алгебры L и $\{\tilde{c}_{ij}^k\}$ алгебры \tilde{L} равенства (5) образуют систему не более чем $N^2(N-1)/2$ квадратичных уравнений с $2N$ неизвестными a_i^j . Это следует из антисимметричности $c_{ij}^k = -c_{ji}^k$ структурных констант. Для решения системы (5) требуется ввести еще одну переменную d и еще одно уравнение $d - \det(a_i^j) = 0$, в эту систему. Справедлива (см. [2]) следующая теорема.

Теорема 1. Две N -мерные алгебры L и \tilde{L} над полем \mathbb{F} , заданные структурными константами $\{c_{ij}^k\}$ и $\{\tilde{c}_{ij}^k\}$, изоморфны тогда и только тогда, когда система уравнений

$$a_i^k a_i^j c_{kl}^n - \tilde{c}_{ij}^m a_m^n = 0, \quad d - \det(a_i^j) = 0 \quad (6)$$

с неизвестными $\{a_i^j, d\}$ (i, j, \dots, N) имеет решение в \mathbb{F} и при этом $d \neq 0$.

Подчеркнем следствие из данной теоремы. Если алгебры Ли L и \tilde{L} не изоморфны как комплексные, тогда при решении системы уравнений (6) должно получаться $d = 0$. Так как d входит в систему (6) линейным образом, то базис Грёбнера G идеала, порожденного полиномами из левых частей систем (8), в комплексном случае $\mathbb{F} = \mathbb{C}$ будет давать $d \in G$. Если быть точнее, этот факт следует из линейности (6) по d и возможности выбора упорядочения, при котором d будет иметь наивысший приоритет. Очевидно, что у системы (6) всегда имеется тривиальное решение $a_i^j = 0, d = 0$. Кроме того, имеет место следующая теорема.

Теорема 2. Если система (6) имеет нетривиальное решение, то она имеет бесконечное множество решений

2. Описание алгоритма нахождения изоморфных алгебр Ли

В настоящем разделе приводится краткое описание алгоритма нахождения изоморфных алгебр Ли. Реализация алгоритма выполнялась в системе символьной математики *Maple*. В работе исследовались изоморфизмы пятимерных алгебр Ли, однако представленный алгоритм

позволяет находить изоморфизм алгебр Ли любых размерностей. Опишем разработанный алгоритм.

1. Определяется пятимерная алгебра $L_1 : [e_i, e_j] = c_{ij}^k E_k$ (задаётся пользователем), а также пятимерная алгебра $L : [e_i, e_j] = s_{ij}^k e_k$ (задаётся пользователем или выбирается из библиотеки *LieAlgebras* в *Maple*).

2. Формируется система квадратичных уравнений (8).

3. Полученная система решается с помощью команды *solve* в *Maple*.

4. Отбрасываются решения, в которых $\det(A) = 0$. Если множество оставшихся решений не пусто, то алгебра L_1 изоморфна алгебре L .

5. Выводится матрица преобразования A .

Кратко опишем библиотеки, используемые в процессе работы вышеозначенного алгоритма. Библиотека *DifferentialGeometry* (в том числе ее составляющая *LieAlgebra*) использовалась для вычислений и преобразований, связанных с алгебрами Ли. Использование этой библиотеки упростило работу с вводом алгебр Ли, так как она содержит уже некоторый начальный набор известных алгебр. Библиотека *ListTools* использовалась во время работы со списками и массивами, а именно, для осуществления процесса поиска и определения индексов нужных элементов в списке.

Опишем также кратко набор программных процедур, имплементированных во время разработки алгоритма, необходимых для автоматизации процесса поиска изоморфных алгебр Ли. Была разработана процедура «getAlgebraTableRepresentation», выводящая представление необходимой алгебры Ли в табличной форме. Процедуры «getAlgebra» и «getAlgebra2» необходимы для перевода таблицы, полученной предыдущей функцией, в матричную форму, где каждый элемент матрицы является массивом (например, $[s_{ij}^1, s_{ij}^2, s_{ij}^3, s_{ij}^4, s_{ij}^5]$).

Ввиду ограничения объема статьи и громоздкости разработанных функций, программный код не приводится в настоящей работе.

3. Примеры нахождения изоморфных алгебр Ли с использованием разработанного алгоритма

В настоящем разделе рассмотрим пример работы алгоритма для пятимерной алгебры (в пятимерном случае известна полная классификация всех возможных алгебр – список Мубаракзянова, см [3]).

Пусть исследуемая алгебра Ли, определяемая как:

$$E_1 = \begin{pmatrix} t_1 & 0 & \frac{1}{2}t_1^2i & 0 \\ 0 & 0 & 0 & 0 \\ 4i & 0 & -t_1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$E_2 = \begin{pmatrix} 0 & 0 & 0 & i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_3 = \begin{pmatrix} \frac{1}{2}m_3 & \frac{1}{2}t_1(2\varepsilon+1) & \frac{1}{8}m_3t_1i & 0 \\ 0 & \frac{1}{2}m_3 & 0 & 1 \\ 0 & 2i(2\varepsilon+1) & m_3 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

$$E_4 = \begin{pmatrix} \frac{1}{2}m_4 & \frac{1}{2}t_1(2\varepsilon-1) & -\frac{1}{8}m_4t_1i & 0 \\ 0 & \frac{1}{2}m_4 & 0 & i \\ 0 & -4\varepsilon+2 & m_4 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix},$$

где t_1, m_3, m_4 не равны нулю, изоморфна алгебре из списка Мубаракзянова, которая представлена в виде коммутаторных соотношений:

$$[e_1, e_5] = 2e_1, [e_2, e_3] = e_1, [e_2, e_5] = e_2, [e_3, e_5] = e_3, [e_4, e_5] = e_4.$$

Используя процедуру «getAlgebra», создаём матрицу, в которой хранятся структурные константы алгебры L (см. рис. 1). Используя процедуру «getAlgebra2», создаём матрицу, в которой хранятся структурные константы алгебры L_1 (см. рис. 2). После, формируем левые части системы полиномиальных уравнений, включающих структурные константы обеих алгебр L и L_1 , а также подлежащие определению элементы неизвестной матрицы преобразований A . На рис. 2 представлены только некоторые левые части этой системы (так как она содержит 126 уравнений и имеет громоздкий вид).

```
c := getAlgebra(algebraRepresentation, basin, 19);
```

$$c := \begin{bmatrix} [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [_p, -1, 0, 0, 0, 0] \\ [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [1, _p, 0, 0, 0, 0] \\ [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, _g, -_s, 0, 0] \\ [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, 0, 0, 0, 0] & [0, 0, _s, _g, 0, 0] \\ [-_p, 1, 0, 0, 0, 0] & [-1, -_p, 0, 0, 0, 0] & [0, 0, -_g, _s, 0, 0] & [0, 0, -_s, -_g, 0, 0] & [0, 0, 0, 0, 0, 0] \end{bmatrix}$$

Рис. 1. Структурные константы алгебры L

```
> s := getAlgebra2(L1, basin)
```

$$s := \left[\left[[0, 0, 0, 0, 0, 0], [0, t_1, 0, 0, -4, 0], \left[-\frac{1}{2} m_3, 0, 0, 0, 0, 0 \right], \left[-\frac{1}{2} m_4, 0, 0, 0, 0, 0 \right], \left[0, \frac{1}{4} t_1^2, 0, 0, -t_1, 0 \right] \right], \right. \\ \left. \left[[0, -t_1, 0, 0, 4, 0], [0, 0, 0, 0, 0, 0], \left[0, -\frac{1}{2} m_3, 0, 0, 0, 0 \right], \left[0, -\frac{1}{2} m_4, 0, 0, 0, 0 \right], [0, 0, 0, 0, 0, 0] \right], \right. \\ \left. \left[\left[\frac{1}{2} m_3, 0, 0, 0, 0, 0 \right], \left[0, \frac{1}{2} m_3, 0, 0, 0, 0 \right], [0, 0, 0, 0, 0, 0], \left[0, t_1, -\frac{1}{2} m_4, \frac{1}{2} m_3, -4, 0 \right], \left[0, -\frac{1}{8} m_3 t_1, 0, 0, m_3, 0 \right] \right], \right. \\ \left. \left[\left[\frac{1}{2} m_4, 0, 0, 0, 0, 0 \right], \left[0, \frac{1}{2} m_4, 0, 0, 0, 0 \right], \left[0, -t_1, \frac{1}{2} m_4, -\frac{1}{2} m_3, 4, 0 \right], [0, 0, 0, 0, 0, 0], \left[0, -\frac{1}{8} m_4 t_1, 0, 0, m_4, 0 \right] \right], \right. \\ \left. \left[\left[0, -\frac{1}{4} t_1^2, 0, 0, t_1, 0 \right], [0, 0, 0, 0, 0, 0], \left[0, \frac{1}{8} m_3 t_1, 0, 0, -m_3, 0 \right], \left[0, \frac{1}{8} m_4 t_1, 0, 0, -m_4, 0 \right], [0, 0, 0, 0, 0, 0] \right] \right]$$

Рис. 2. Структурные константы алгебры L_1

Используя средства *Maple*, а именно функцию «solve», мы решаем эту систему. Всего у неё оказалось 56 решений, среди которых большинство приводят к обнулению определителя матрицы A преобразования, то есть нам они не подходят. Однако в полученном списке решений имеется несколько таких, что определитель матрицы преобразований не обращается в ноль. При этом элементам матрицы A , которые остались свободными, можно придать произвольные значения (но так, чтобы определитель не обратился в ноль). В итоге мы можем получить матрицу перехода от исходной алгебры к изоморфной, имеющую вид:

$$A = \begin{pmatrix} 0 & -t_1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & -2t_1 & -m_4 & m_3 & 8 \\ 0 & 0 & -2\frac{1}{m_3} & 0 & 0 \end{pmatrix}.$$

Далее, используя полученную матрицу, найдем представление матрицы L_1 в новом базисе. Непосредственной проверкой убеждаемся, что таблица коммутирования для определённой таким образом алгебры

полностью совпадает с таблицей коммутирования для алгебры L , то есть рассматриваемые алгебры действительно изоморфны.

Таким образом, рассмотренный пример показывает, что алгоритм верно определяет наличие изоморфизма между двумя данными алгебрами.

Заключение

В данной работе исследовалась проблема автоматического поиска изоморфных алгебр Ли, возникающая в ряде задач алгебры, комплексного анализа, а также теории дифференциальных уравнений. С помощью разработанного алгоритма в системе символьной математики Maple удалось создать программу, которая позволяет в автоматическом режиме выяснять, являются ли две представленные алгебры изоморфными.

Список литературы

1. Атанов, А. В. Аффинно-однородные поверхности типа $(0,0)$ в пространстве S^3 / А. В. Атанов, А. В. Лобода // Матем. заметки. – 2015. – Т. 97, № 2. – С. 309–313.
2. Gerdt, V. P. Isomorphism Verification for Complex and Real Lie Algebras by Grobner Basis Technique // V. P. Gerdt, W. Lassner. – Modern Group Analysis: Advanced Analytical and Computational Methods in Mathematical Physics. – 1993. – P. 245–254.
3. Мубаракзянов, Г. М. О разрешимых алгебрах Ли / Г. М. Мубаракзянов // Изв. вузов. Матем. – 1963. – № 1. – С. 114–123.

Стегоанализ цветных изображений с использованием глубоких нейронных сетей

Н. А. Экерт

Студент магистрант

А. А. Сирота

Профессор

Введение

Стегоанализ – это наука о выявлении факта передачи скрытой информации в анализируемых данных, ее извлечении, дешифровки и/или уничтожении.

Стегоанализ в изображениях представляет собой процесс, приводящий к подтверждению или опровержению факта наличия скрытой передачи сообщения в контейнере изображения.

Данная проблема является актуальной и тесно связана с отраслями защиты информации и интеллектуальной собственности [1], защиты подлинности документов, подтверждения достоверности и подлинности переданной информации, военной отраслью, банковской сферой и многими другими отраслями [2].

Данная статья посвящена сравнению эффективности распознавания четырех наиболее популярных алгоритмов стеганографии LSB, HUGO, WOW, S-UNIWARD с использованием сверточной нейронной сети, основанной на архитектуре YEDROUDJ-NET, методов SPAM и SRM.

1. Описание алгоритмов

1. Алгоритм LSB. LSB (Least Significant Bit) – алгоритм, основная суть которого заключается в использовании погрешности дискретизации, которая всегда существует в оцифрованных изображениях. Данная погрешность равна наименьшему значащему разряду числа, определяющему величину цветовой составляющей элемента изображения (пикселя).

Младший бит пикселя изображения в оттенках серого хранит младший разряд соответствующего двоичного числа и, следовательно, его изменение минимально влияет на изменение числа в целом.

Алгоритм заменяет наименее значимый бит в нескольких байтах файла-носителя, чтобы скрыть последовательность байтов, содержащих скрытые данные, что в большинстве случаев не вызывает значительной трансформации изображения и не обнаруживается визуально. Стоит отметить, что такая методика встраивания обеспечивает довольно высокую пропускную способность 1 бит/пиксель.

В настоящее время данный алгоритм является наиболее распространенным, но, в то же время, наименее стойким.

Пустой контейнер можно представить в двоичной форме следующим образом:

$$x_{ij} = \sum_{p=0}^8 (x_{ij})_p \cdot 2^p, \quad (1)$$

В заполненном контейнере к данному представлению прибавляется бит стегосообщения m_{ij} :

$$x_{ij} = \sum_{p=0}^8 (x_{ij})_p \cdot 2^p + m_{ij}, \quad (2)$$

2. Алгоритм HUGO. HUGO (Highly Undetectable steGO) – алгоритм, основанный на идее о том, что в процессе вложения, каждый пиксель изображения изменяется с вероятностью, обратно пропорциональной его «вкладу» в суммарное искажение изображения.

Данный стегоалгоритм оптимизирован для противодействия обнаружению против алгоритма стегоанализа SPAM. Однако, для некоторых других методов стегоанализа, алгоритм является так же сложным.

Матрица H размерности $k \cdot n$ выбирается специальным образом

$$HY = M, \quad (3)$$

где k – длина цепочки бит M , n – полное количество отсчетов (пикселей) контейнера, Y – вектор стего.

Из множества решений выбирается такое, которое при определенном X минимизирует суммарное искажение контейнера.

Нахождение цены изменения исходя из Марковских вероятностей между соседними пикселями является интересным и сложным подходом, используемым в HUGO. Происходит оно по восьми направлениям (горизонтальном, вертикальном, диагональном).

Алгоритм HUGO считается лучшим для высокоскоростных стегосистем на сегодняшний день, однако на данный момент нет

строгого доказательства, что данный алгоритм будет строго секретным для различных методов стегоанализа.

3. Алгоритм WOW. WOW (Wavelet Obtained Weights) – алгоритм, направленный на более эффективное скрытие информации в изображениях. Внедрение информации происходит в специально определенные места. После внедрения происходит небольшое изменение окрестности точки изображения для большей скрытности алгоритма.

В данном стегоалгоритме используется набор направленных высокочастотных фильтров для получения “направленных остатков”, которые связаны с предсказуемостью пикселя в определенном направлении. Измеряя влияние внедрения на каждую направленную остаточную нагрузку и агрегируя эти воздействия соответствующим образом, стоимость внедрения остаётся высокой, когда содержание предсказуемо, по крайней мере, в одном направлении (гладкие области и вдоль краев) и низкой, где содержание непредсказуемо в каждом направлении (например, в зашумленных зонах или текстурированных зонах).

В связи с этим, данный алгоритм является адаптивным и лучше сопротивляется стеганализу с использованием богатых моделей (SRM).

4. Алгоритм S-UNIWARD. S-UNIWARD (Universal Wavelet Relative Distortion) – алгоритм, в основе которого лежат наработки, используемые в алгоритмах HUGO и WOW. Однако, в случае с S-UNIWARD, используются наиболее зашумлённые места в изображении для улучшения показателей скрытности.

В данном стегоалгоритме оценивается гладкость изображения в разных направлениях с помощью 8-кратного набора вейвлет-направленных фильтров Добеши ($D-WDFB$), $B = \{K(1), K(2), K(3)\}$, который состоит из высокочастотных фильтров LH , HL и HH (ядер K). Эти три фильтра построены из одномерных низкочастотных (h) и высокочастотных (g) фильтров декомпозиции.

Если пара контейнер-изображений X (пустой) и Y (заполненный) являются изображениями JPEG, они сначала декомпрессируются в пространственную область, а затем к ним применяются вейвлет-преобразования. Искажение между обоими изображениями представляет собой сумму относительных изменений вейвлет-коэффициентов w.r.t. контейнер-изображения:

$$D(X, Y) \triangleq \sum_{k=1}^3 \sum_{uv} \frac{W_{uv}^{(k)}(X) - W_{uv}^{(k)}(Y)}{\xi + |W_{uv}^{(k)}(X)|}, \quad (4)$$

где $W_{uv}^{(k)}(X)$ и $W_{uv}^{(k)}(Y)$ – uv -й Вейвлет-коэффициент в k -ом разложении, ξ – стабилизирующая константа, чтобы избежать деления на ноль. Безопасность внедрения с использованием S-UNIWARD довольно нечувствительна к точному значению ξ .

Стоит понимать, что отношение в (4) меньше при изменении большого вейвлет-коэффициента контейнера, которое произойдет в текстурах, шумных областях и вблизи краев. С другой стороны, если хотя бы один небольшой коэффициент, который изменяется на сравнительно большую величину, величина искажения также окажется большой. Таким образом, происходит препятствование внесению изменений в регионы, где содержание является гладким (и, следовательно, его можно моделировать) по меньшей мере в одном направлении.

2. Архитектура сети Yedroudj-NET

Данная нейронная сеть [3] была основана на наработках Xu-Net [4], Ye-Net [5], в ее основную структуру входит несколько модулей: Preprocessing – предварительная обработка изображения, Convolution module – слои свёртки, Classification module – модуль классификации, состоящий из 3-х полносвязных слоев. Общая схема архитектуры нейронной сети Yedroudj-NET представлена на рис. 1.

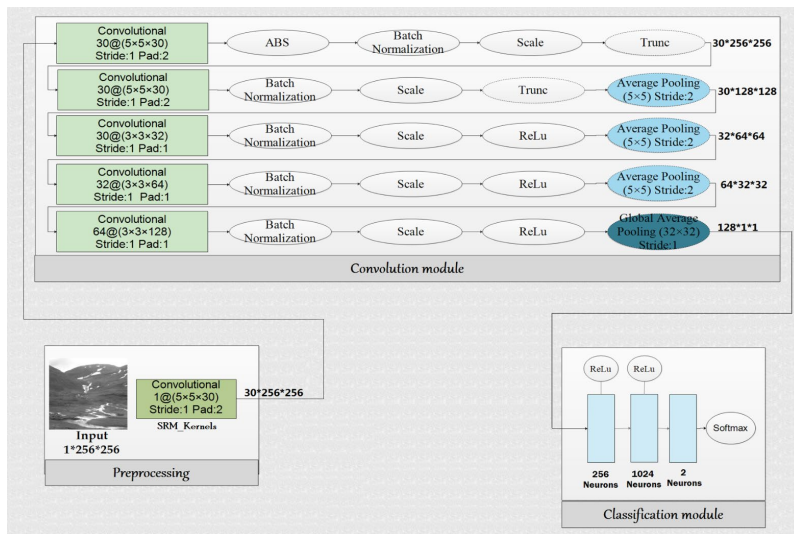


Рис. 1. Общая схема архитектуры сети Yedroudj-NET

Блок предварительной обработки фильтрует входное изображение с помощью фильтра высоких частот для извлечения остатков шумовой составляющей, после чего, обработанное изображение подается в сеть. Эта предварительная обработка призвана в значительной степени подавить содержание изображения, сузить динамический диапазон и, таким образом, увеличить отношение сигнала к шуму между слабым сигналом стего (если он присутствует) и сигналом изображения. Как результат, CNN может обучаться на более компактном и робастном сигнале. Стоит заметить, что значения ядер фильтров в блоке предварительной обработки, т. е. веса, не оптимизируются в процессе тренировки.

Остальную часть CNN можно разделить на свёрточный модуль, предназначенный для представления признаков, который преобразует входное изображение в вектор признаков, и классификационный модуль, состоящий из трех полностью связанных слоёв и слоя softmax, который принимает классификационное решение (присутствие, либо отсутствие стего).

Свёрточный модуль имеет пять блоков, каждый блок состоит из следующих слоёв:

- Уровень свёртки. Был установлен размер свёрточных ядер 5×5 для блоков 1, 2 и 3×3 для блоков 3-5.

- Уровень активации абсолютного значения (ABS). Этот уровень ABS используется только в блоке 1. Он заставляет статистическое моделирование учитывать знаковую симметрию остатков шума.

- Пакетная нормализация. Выполняет функцию нормализации распределения каждого признака к нулевому среднему и единичной дисперсии, и, в конечном итоге, масштабирует и переводит распределение. Преимуществом использования данного слоя является то, что он позволяет использовать большую скорость обучения и повышает точность обнаружения.

- Нелинейный уровень активации. Для блоков 1 и 2 используется функция усечения для ограничения диапазона значений данных и предотвращения моделирования больших значений более глубокими слоями.

- Объединение в единое целое. Этот слой объединения используется в блоках 2-5, что позволяет уменьшить выборку карт признаков и, таким образом, уменьшить размерность. Для последнего блока выполняется объединение глобального среднего значения для генерации одного элемента за другим для каждой соответствующей карты признаков, тем самым предотвращая получение статистическим моделированием информации о местоположении встроенных пикселей

из обучающих данных. В первом блоке данный слой отсутствует во избежание потери информации в начале обучения сети.

Признаки, извлекаемые из сверточного модуля, попадают в классификационный модуль, состоящий из трех полностью соединенных слоев. Количество нейронов в первом и втором слоях составляет 256 и 1024 соответственно, а последний слой имеет два нейрона, соответствующих числу классов выхода сети. В конце этого модуля используется функция активации softmax для получения классификационного решения о наличии, либо отсутствии стего в контейнере изображения.

3. Результаты исследования

Эксперименты проводились на базе данных BOSSbase 1.01 [6]. Эта база данных содержит 10000 изображений в оттенках серого, полученные семью цифровыми камерами в необработанном формате и затем ужаты до размера 512×512 пикселей в формате png. Используемый размер изображений для экспериментов был предварительно скорректирован до 256×256 пикселей с целью увеличения скорости производимых вычислений. Для оценки эффективности модели Yedroudj-NET эксперименты были проведены на четырех современных и наиболее известных стеганографических алгоритмах: LSB, HUGO, WOW, и S-UNIWARD. Результаты работы сети Yedroudj-NET сравнивались с двумя вручную подобранными наборами признаков: низкоразмерным набором SPAM [7] со 162 признаками, реализованным с помощью SVM на основе гауссовой радиальной базисной функции, и высокоразмерным набором SRM [8] с количеством признаков 548, одним из современных наборов признаков, реализованным с помощью ансамбля классификаторов.

Производительность всех методов оценивалась на четырех полезных нагрузках: 0.2, 0.3, 0.4 и 0.5 бит на пиксель (bpp). Сравнение вероятности ошибки обнаружения представлено в табл. 1.

На наборе данных BOSSbase, по всем четырем алгоритмам внедрения и полезным нагрузкам, алгоритм, основанный на применении сверточной нейронной сети Yedroudj-NET достигает гораздо более низкой ошибки обнаружения, чем набор SPAM. По сравнению с набором SRM, реализованным с помощью ансамбля классификаторов, ошибка обнаружения 1-10% меньше в зависимости от полезной нагрузки.

Таблица 1

Результаты сравнения вероятностей ошибок стегоанализа

Алгоритмы		bpp			
		0,2	0,3	0,4	0,5
LSB	SPAM	28,5%	22,7%	17,2%	12,1%
	SRM	17,9%	13,7%	9,9%	6,5%
	Yedroudj-NET	16,2%	12,3%	8,9%	5,7%
HUGO	SPAM	46,2%	42,9%	39,1%	35,7%
	SRM	34,6%	29,6%	25,2%	21,4%
	Yedroudj-NET	27,9%	20,9%	14,3%	8,1%
WOW	SPAM	46,3%	42,2%	38,2%	34,9%
	SRM	36,5%	31,2%	25,6%	22,1%
	Yedroudj-NET	27,8%	20,9%	14,1%	7,9%
S-UNIWARD	SPAM	44,7%	40,1%	35,1%	30,6%
	SRM	36,6%	31,5%	25,4%	21,4%
	Yedroudj-NET	36,7%	28,8%	22,8%	18,6%

График результатов сравнения вероятностей ошибок стегоанализа представлен на рисунке 2. Здесь наглядно видно преимущество используемого современного метода стегоанализа, основанного на применении сверточной нейронной сети с используемой архитектурой Yedroudj-NET.

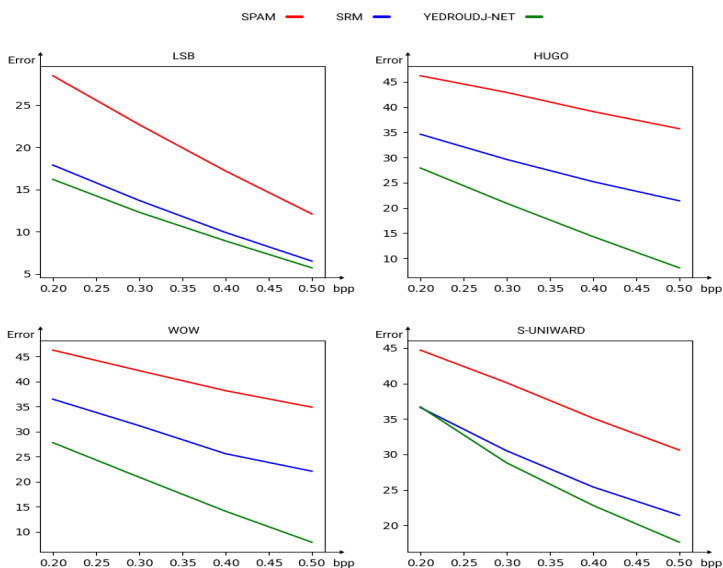


Рис. 2. График результатов сравнения вероятностей ошибок стегоанализа

Заключение

Данная статья посвящена сравнению эффективности распознавания четырех наиболее популярных алгоритмов стеганографии с использованием сверточной нейронной сети, основанной на архитектуре Yedroudj-NET, а также методов SPAM и SRM. В работе были рассмотрены методы LSB, HUGO, WOW, S-UNIWARD, была представлена архитектура сверточной нейронной сети Yedroudj-NET и проведено сравнение эффективности ее распознавания с методами SPAM и SRM. Алгоритм, основанный на применении сверточной нейронной сети Yedroudj-NET превосходит другие подходы, не использующие глубокие нейронные сети и способен улучшить качество проводимого стегоанализа.

Данный алгоритм в среднем выглядит более выигрышным на 10% в сравнении низкоразмерным набором SPAM и на несколько процентов в сравнении с высоко разрешающим набором SRM.

Результаты этих экспериментов подтверждают, что автоматическое вырабатывание признаков эффективно в разработанной модели и данный алгоритм позволяет эффективнее противодействовать сложным методам встраивания.

Список литературы

1. Исследование алгоритмов стегоанализа изображений с использованием глубоких нейронных сетей / Н. А. Нагорный, А. А. Сирота // Сборник студенческих научных работ факультета компьютерных наук ВГУ. — Воронеж, 2019. — Вып. 13, ч. 2 : Научные работы студентов-магистрантов. — С. 145-151 .
2. Аграновский, А.В, Стеганография. Цифровые водяные знаки и стеганоанализ // Аграновский А.В, Балакин А.В., Грибунин В.Г., Сапожников С.Н. – Москва: Феникс, 2009. – 220 с.
3. Mehdi Yedroudj, Frédéric Comby, Marc Chaumont., “Yedrouj-Net: An efficient CNN for spatial steganalysis“, 26 Feb 2018 [Электронный ресурс] / Режим доступа: <https://hal-lirmm.ccsd.cnrs.fr/lirmm-01717550>
4. G Xu, H. Z. Wu, and Y. Q. Shi, “Structural Design of Convolutional Neural Networks for Steganalysis,” IEEE Signal Processing Letters, vol. 23, no. 5, pp. 708–712, May 2016.
5. Jian Ye, Jiangqun Ni, and Y. Yi, “Deep learning hierarchical representations for image steganalysis,” IEEE Transactions on Information Forensics and Security, TIFS, vol. 12, no. 11, pp. 2545–2557, Nov. 2017.
6. Описание базы BOSSBase / [Электронный ресурс] / Режим доступа : <http://agents.fel.cvut.cz/boss/index.php?mode=VIEW&tmpl=materials>

7. Pevny, T., Bas, P., and Fridrich, J., “Steganalysis by subtractive pixel adjacency matrix,” *IEEE Transactions on information Forensics and Security* 5(2), 215–224 (2010).
8. Fridrich, J. and Kodovsky, J., “Rich models for steganalysis of digital images,” *IEEE Transactions on Information Forensics and Security* 7(3), 868–882 (2012).

Содержание

Труды студентов бакалавриата

Беспалов В. М., Соломатин Д. И. Обзор алгоритма monocular SLAM для определения позиционирования объекта в пространстве	4
Булавина Е. П., Соломатин Д. И. Использование нейронных сетей для определения местоположения радужной оболочки глаза на изображениях	11
Герасимов И. А., Абрамова Н. В. Разработка программного модуля бюджетирования предприятия для базовой конфигурации «1С: Бухгалтерия 8.3»	17
Гончаров М. С., Борзунов С. В. Разработка приложения для автоматизации работы с данными в области клинической фармакологии	23
Господарикова В. С., Гаршина В. В. Разработка системы генерации естественно-языковых поисковых запросов по структуре онтологии для обогащения ее новыми фактами	28
Данилова А. В., Борзунов С. В. Программный модуль для работы с торговыми наименованиями лекарственных препаратов	33
Дмитриев В. В., Митрофанова Е. Ю. Разработка программного обеспечения для решения проблемы восстановления лица в формате 3D модели	38
Дунаева Е. В., Кожевникова Н. Б., Серадская Д. М., Минин Л. А. Теорема Шеннона–Котельникова и формула суммирования Пуассона	45
Дынин М. А., Степанцов В. А. Реализация и исследование методов слежения за движущимися объектами в видеопотоке	50
Зверева А. С., Швырева А. В. Методы и алгоритмы обнаружения и оценки количества объектов в видеопотоке	58
Конищева Н. А., Меркутова П. Д., Манжула Е. В., Туровский Я. А. Подходы к оптимизации выравнивания последовательностей нуклеиновых кислот по алгоритму Нидлмана–Вунша на примере биоинформационного анализа последовательностей вируса бешенства	64
Коротков О. И., Туровский Я. А. Автоматизированная система выращивания тканей	72

Кропачев А. Н., Копытина Е. А. Формирование отчета деятельности наркоконтроля по Воронежской области средствами офисных приложений.....	80
Куприн М. А., Лобода А. В. О нильпотентных алгебрах Ли с «избыточными» симметриями.....	85
Лазарев Я. В., Соломатин Д. И. Разработка приложения для получения данных с 4D-рига.....	90
Макушин Д. В., Тарасов В. С. Приложение для совместного прослушивания музыки с персонализированным плейлистом	99
Миронкин А. П., Шаталов И. С. Универсальный испытательный стенд для тестирования и демонстрации работы интерфейсов человек-компьютер.....	104
Нурулаев Э. С., Киселев Е. А. О разложении в ряд по неортогональным цилиндрическим функциям Бесселя первого рода целого порядка.....	110
Первицкий Д. Е., Соломатин Д. И. Разработка приложения для голосового ввода с восстановлением пунктуации в тексте.....	115
Погорелов Р. И., Копытин А. В., Копытина Е. А. Адаптация тиражируемого решения от фирмы 1С (1С: Управление нашей фирмой) и разработка приложения на мобильной платформе 1С.....	121
Попков В. С., Степанцов В. А. Реализация и исследование методов прогнозирования отклика системы на входные воздействия	126
Проскуряков Е. Д., Чекмарев А. И. Разработка системы для мониторинга очагов распространения заболеваний.....	133
Саввин М. О., Митрофанова Е. Ю. Распознавание лиц в условиях ограниченной видимости.....	140
Стеблева Е. А., Иванков А. Ю. Синтез и анализ алгоритмов генерации музыки на основе текстовых данных.....	147
Турченко К. А., Черницын Д. В. Разработка алгоритмов парсинга для информационной системы анализа и визуализации данных маркетплейса Wildberries	154
Шишко Ю. В., Соломатин Д. И. Какие проекты интересны школьникам на примере учеников Яндекс.Лицея	161

Труды студентов магистратуры

Акопян Т. К., Иванков А. Ю. Проверка подлинности цифровых изображений.....	169
Аникин Е. Г., Вахгин А. А. Организация исследования предметной области и проектирования сервиса учета	

нетрудоспособности на примере цифровой медицинской платформы Numedy	173
Архипов С. Ю., Илларионов И. В. Учет SoD-рисков при разработке ролей в системе SAP R/3	180
Бейлина Е. С., Коротков В. В. Система построения таблиц лидеров на основе вероятностной модели для платформы турниров по программированию	185
Гаршин Т. С., Иванков А. Ю. Выделение именованных сущностей из текста на основе предобученной модели bert_mult	192
Господарикова И. С., Гаршина В. В. Методы выявления семантической близости новостных текстовых сообщений	197
Дорофеев Р. С., Акимов А. В. Использование цветовых моделей для улучшения работы алгоритмов распознавания	202
Жевандров Н. М., Лысачев П. С. Реализация модулей взаимодействия с различными типами приборов Against	209
Жуков В. С., Дрюченко М. А. Разработка математического и программного обеспечения для решения задачи высокоточного обнаружения опорных точек на изображении лица человека	216
Зайцев С. А., Глушков М. А., Семенов М. Е. Имитационное моделирование радиодеструкции полимеров	223
Зинькевич Г. В., Храмов В. Ю. Алгоритм устранения неопределенных значений в нормализованных реляционных базах данных	230
Игнатов М. И., Савинков А. Ю. Алгоритм распределения канальных ресурсов в сетях когнитивного радио	238
Илларионов И. В. Система контроля обработки производственных заказов	245
Ильина А. Д., Фертиков В. В. Алгоритм построения скелета с помощью графа смежности для бинарных изображений	251
Казаков К. А., Самойлов Н. К. Модуль сбора, анализа и загрузки информации как подсистема ситуационного центра	256
Карпов Е. А., Семенов М. Е. Новые аспекты колебаний балки с учетом распределенных гистерезисных свойств	261
Карпова М. А., Туровский Я. А. Разработка информационной системы общения с детектированием эмоционального состояния человека	267
Ключкова И. В., Абрамов И. В. Разработка системы учёта заболеваемости «COVID-19» по Воронежской области	275
Коновской А. С., Борзунов С. В. Решение ресурсоемких задач с помощью технологий программирования графических процессоров	283

Копылова А. Е., Киселев Е. А. Разработка игры-квеста для психологического анализа.....	289
Кушнеренко В. К., Вахгин А. А. Разработка информационной системы учёта посещаемости факультета.....	294
Левчук П. О., Иванков А. Ю., Максимов А. В. Система анализа респираторной активности	302
Леонтьев М. Б., Лысачев П. С., Зуев С. А. Нагрузочное тестирование микроконтроллера ESP8266.....	308
Леонтьев М. Б., Лысачев П. С., Зуев С. А. Разработка универсальной системы управления микроконтроллерами.....	314
Матвеев А. В., Самойлов Н. К. Автоматизация тестирования RESTful API на основе BDD-подхода.....	321
Мидов Н. В., Вахгин А. А. Расширение для поддержки мультиарендности в работе приложения	326
Митин Д. С., Акимов А. В. Фреймворк для обучения и тестирования детекторов лиц на изображении	333
Никонов И. Е., Савинков А. Ю. Разработка ПО контроллера системы отопления с адаптивными функциями управления	339
Онищук А. Е., Епифанцев А. А. Оценка масштабируемости протоколов маршрутизации на имитационной модели беспроводной ячеистой сети в среде симулятора NS3.....	346
Отырба Р. Р., Дрюченко М. А. Использование методов компьютерного зрения и машинного обучения для анализа изображений спутниковой и аэрофотосъемки в интересах мониторинга изменений на местности.....	354
Панькова А. А., Митрофанова Е. Ю. Реализация и исследование нейросетевого алгоритма переноса стиля изображения.....	363
Папина А. А., Матвеев М. Г. Разработка робота для построения карты местности	370
Полюк Я. С., Соломатин Д. И. Методы распознавания сарказма в тексте	376
Прохорченко Л. А., Соломатин Д. И. Преимущества предварительного выравнивания трёхмерных моделей при поиске разреженных соответствий	382
Рузин С. Б., Илларионов И. В. Применение методологии Behavior-Driven Development в процессах разработки и автоматизированного тестирования ПО.....	391
Сафонов В. Г., Матвеев М. Г., Алейникова Н. А. Алгоритмы ранжирования товаров для электронной коммерции	397

Смирнова А. А., Акимов А. В. Обучение классификатора для распознавания лиц по алгоритму Виолы-Джонса	404
Солодухин В. М., Борисов Д. Н. Обработка аудиосигнала на DSP-кластере СпК САЛЮТ-ЭЛ24Д	411
Стеблюк И. И., Илларионов И. В. Некоторые аспекты моделирования онтологии тестирования программного обеспечения.....	421
Титова Ю. Ю., Атанов А. В. Метод нормальных форм в задаче классификации голоморфно однородных поверхностей.....	429
Тутельян М. М., Соломатин Д. И. Плагин навигации игровых юнитов для Unity Engine	435
Уваров Е. В., Акимов А. В. Статистический анализ блоков размеченного текста.....	442
Фирсова И. А., Таратухин В. В. Анализ использования методологии «Дизайн-мышление» при проектировании программного обеспечения	450
Хацкевич Д. И., Храмов В. Ю. Методика оценки риска информационной безопасности систем обработки информации.....	455
Хозиев Ю. Г., Лысачев П. С. Применение нейронных сетей для идентификации осетровых рыб	463
Шейна О. А., Атанов А. В. Поиск изоморфных алгебр Ли с помощью систем полиномиальных уравнений	470
Экерт Н. А., Сирота А. А. Стегоанализ цветных изображений с использованием глубоких нейронных сетей.....	477

Научное издание

**ТРУДЫ МОЛОДЫХ УЧЁНЫХ
ФАКУЛЬТЕТА КОМПЬЮТЕРНЫХ НАУК ВГУ**

В ы п у с к 1

Под редакцией *Д. Н. Борисова*

Издано в авторской редакции

Минимальные системные требования:
PC не ниже класса Pentium I, 32 Mb RAM,
свободное место на HDD 16 Mb,
Windows 95/ 98, Adobe Acrobat Reader,
дисковод CD-ROM 2-х, мышь.

Подписано к использованию 11.06.21.
Объем данных 21 Мб. 1 электрон. опг. диск (CD-ROM).
Тираж 500 экз. Заказ 212.

ООО «Вэлборн»
394068, г. Воронеж, Московский пр-т, 98
Тел. +7 (930) 4035-418
<http://wellborn@scirep.ru>, e-mail: wellborn@scirep.ru
Изготовлено фирмой «Большой формат» (ООО «Твой выбор»)
394018, г. Воронеж, ул. Кости Стрелюка, д. 11/13, офис 6
Тел. +7 (473) 238-26-38

<http://big-format.ru>, e-mail: 382638@mail.ru