Анализ методов машинного обучения на примере задачи многоклассовой классификации текста

М.В. Лаптев, email: mv@paq.su

ФГБОУ ВО «Вятский государственный университет»

Аннотация. *В работе рассматривается задача многоклассовой классификации текстов с большим количеством (около 350) несбалансированных классов на примере категоризации продуктов по их названию и описанию. Исследуются различные методы предобработки данных (лемматизация, стемминг) и модели представления (TF-IDF, GloVe, BERT). Лучшие результаты получены с использованием модели BERT: macro F1-score = 0.8512, weighted F1-score = 0.9360.*

Ключевые слова: *Многоклассовая классификация, машинное обучение, обработка естественного языка, TF-IDF, GloVe, BERT*

Введение

В машинном обучении многоклассовая классификация – это задача отнесения объектов к одному из трех или более классов [1]. В настоящее время многоклассовая классификация широко применяется во многих сферах, например, в задачах медицинской диагностики или классификации документов. В последние годы в обработке естественного языка произошел прорыв – появление нейросетевой архитектуры Transformer [2], а также модели BERT [3] и ее модификаций значительно улучшило результаты в задачах обработки естественного языка. Однако эти модели, как правило, тестировались на задачах с относительно небольшим количеством классов, соответственно, целью данной работы является сравнение результатов современных нейросетевых моделей и других подходов на примере решения реальной задачи многоклассовой классификации товаров по их названию и описанию с большим количеством (около 350) несбалансированных классов.

Работа имеет следующую структуру. Сначала рассматривается анализ и предобработка данных. Затем описываются подходы, основанные на частоте встречающихся в описании слов (TF-IDF) [4] и на преобразовании слов в векторное пространство (GloVe) [5]. Также обсуждается модель BERT и её модификации. В последнем разделе представлены заключительные результаты рассмотренных подходов и сделаны итоговые выводы.

1. Анализ и предобработка данных

В работе исследуется набор данных с описаниями товаров для животных[1], состоящий из 35749 записей и 349 различных классов. На рис. 1 показан пример информации о товаре, состоящий из наименования, описания и категории.

Название продукта	*Описание продукта*	*Категория продукта*
Blue Buffalo Life Protection Formula Adult Chicken & Brown Rice Recipe Dry Dog Food By Blue Buffalo	Blue Buffalo Life Protection Formula was created for the holistic health and well-being of adult dogs. All formulas start with real meat, whole grains, garden veggies and fruit, plus added LifeSource Bits, a precise blend of nutrients that have been enhanced with a Super 7 package of antioxidant-rich ingredients. This Adult Chicken & Brown Rice Recipe features delicious, protein-rich deboned chicken and other natural ingredients for a healthy meal your dog will love.	Dog, Food, Dry food

Рис. 1. Пример информации о товаре

После проверки на пропущенные значения и дубликаты, а также последующего удаления таковых записей, осталось 35330 записей. На рис. 2 показано распределение количества записей по классам в порядке убывания, что наглядно демонстрирует несбалансированность классов.

[1] Набор данных предоставлен компанией S3 Stores, Inc. (https://www.s3stores.com).

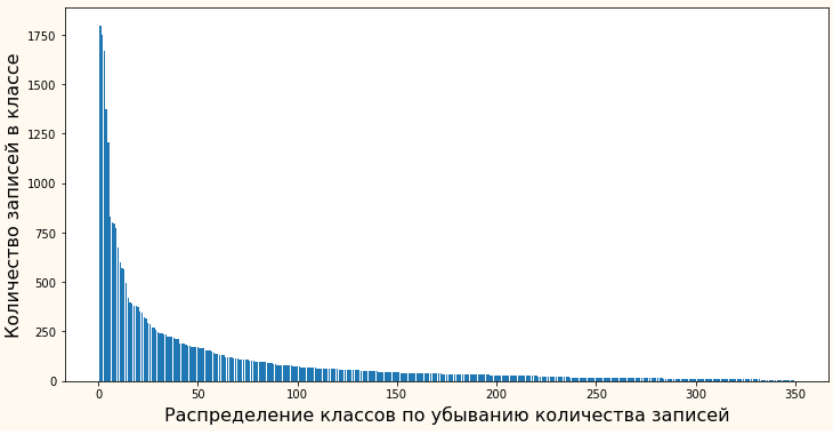

Рис. 2. Распределение классов по убыванию количества записей

Для последующих преобразований название продукта и его описание конкатенируются. С целью придания названию большего веса, оно дублируется. Знаки препинания, цифры и других неинформативные символы удаляются при помощи регулярных выражений – остаются только слова, приведенные к нижнему регистру.

После этого, используя библиотеку NLTK [6], к тексту применяются раздельно *лемматизация* и *стемминг*, а затем для получившихся трех вариантов текста (третий вариант – оставить текст как есть) производится *токенизация* текста по словам. *Лемматизация* – это процесс приведения словоформы к лемме – её нормальной форме [7]. Например, “better” → “good”. *Стемминг* – это процесс нахождения основы слова для заданного исходного слова [8]. Например, “playing” → “play”. *Токенизация* по словам – это процесс разделения предложений на слова-компоненты [4]. Таким образом, после процесса токенизации имеются три варианта текстовых описаний, наиболее подходящее из них будет выбираться экспериментально.

В заключение этапа анализа и предобработки данных необходимо выбрать метрику качества для оценки результатов работы классификаторов. В качестве метрики была выбрана F1-мера, которая рассчитывается по следующей формуле:

$$F = \frac{2 \cdot (P \cdot R)}{P + R}, \quad (1)$$

где *P – точность (precision)* в пределах класса – это доля документов, действительно принадлежащих данному классу, относительно всех документов, отнесенных к данному классу. *R – полнота (recall)* – доля найденных классификатором документов, принадлежащих классу, относительно всех документов этого класса в выборке [4]. Так как задача многоклассовая с несбалансированными классами, то будут использоваться две метрики: F1-macro (далее обозначается F_m) – вычисляет F1-меру для каждого класса, а потом возвращает среднее арифметическое по всем классам, F1-weighted (далее обозначается F_w) – вычисляет F1-меру для каждого класса, а потом возвращает среднее арифметическое по всем классам с учетом доли объектов каждого класса в наборе данных. Разница между этими метриками будет показывать, насколько хорошо распознаются классы с небольшим количеством объектов.

2. TF-IDF

TF-IDF (term frequency – inverse document frequency) – это статистическая мера для оценки важности слова в документе, являющимся частью текстового корпуса [4]. В большом текстовом корпусе некоторые слова будут присутствовать очень часто (“the”, “a”, “is”), неся при этом очень мало значимой информации о фактическом содержании документа. Если передать результаты прямого подсчета частоты слов классификатору, эти слова отодвигали бы на второй план частоты более редких, но более информативных терминов. Чтобы избежать этого, используют удаление *стоп-слов* и преобразование TF-IDF. *Стоп-слова* – это такие слова, как “and”, “the”, “him”, которые считаются неинформативными при представлении содержания текста, и которые могут быть удалены, чтобы не истолковывать их как сигнал для предсказания [4].

TF означает частоту термина, то есть количество раз, когда термин встречается в данном документе, а TF-IDF означает частоту термина, умноженную на обратную частоту документа [4]:

$$tfidf\ (t, d) = tf\ (t, d) \cdot idf\ (t), \tag{2}$$

где t – термин, d – документ. В используемой в задаче библиотеке scikit-learn [4] IDF вычисляется следующим образом:

$$idf\ (t) = \log \frac{1 + n}{1 + df\ (t)} + 1, \tag{3}$$

где n – общее число документов в текстовом корпусе, а $df(t)$ – количество документов в текстовом корпусе, содержащих термин t. Полученные векторы TF-IDF затем нормализуются евклидовой нормой:

$$v_{norm} = \frac{v}{\sqrt{\sum_{i=1}^{n} v_i^2}} \quad (4)$$

Далее, создавая TF-IDF матрицы из трех вариантов моделей представления текста и удаляя стоп-слова на основе списка стоп-слов библиотеки scikit-learn, проверяются результаты на наиболее быстрых классификаторах при использовании пятикратной перекрестной проверки с сохранением распределения по классам [4]. В табл. 1 указаны результаты классификации.

Таблица 1

Классификация разных типов текстового описания

Модели представления текста	Logistic Regression		SVC		KNN	
	F_m	F_w	F_m	F_w	F_m	F_w
TF-IDF	0.4998	0.8032	0.7462	0.8978	0.5823	0.7314
TF-IDF с лемматизацией	0.5016	0.8050	0.7462	0.8989	0.5859	0.7376
TF-IDF со стеммингом	0.5025	0.8061	0.7476	0.8974	0.5870	0.7362

В данном случае в каждой из созданных TF-IDF матриц на каждый пример приходится не менее 25000 признаков. Учитывая большой размер признакового пространства, целесообразно сократить признаковое пространство, стараясь отобрать наиболее полезные признаки. Результат будем оценивать на SVC, так как он показал себя предварительно лучше других. Стандартный метод в TF-IDF подходе – частотный фильтр, отсеивающий слова, встречающиеся в текстовом корпусе менее заданного количества раз, на этапе построения TF-IDF матрицы. Сократим размерность признакового пространства до 10000. В табл. 2 указаны результаты классификации с использованием частотного фильтра.

Таблица 2

Результаты классификации с использованием частотного фильтра

Типы текстового описания	SVC	
	F_m	F_w
TF-IDF	0.7454	0.8968
TF-IDF с лемматизацией	0.7437	0.8961
TF-IDF со стеммингом	0.7466	0.8971

Таким образом, стемминг проявил себя лучше всего при использовании отбора признаков по сравнению с другими типами текстового описания. Рассмотрим другие методы отбора признаков на основе стемминга и SVC, а именно VarianceThreshold, χ^2 и Recursive feature elimination (RFE). VarianceThreshold – это метод, удаляющие признаки, имеющие дисперсию ниже заданного порога, таким образом, стараясь оставить наиболее значимые признаки [4]. χ^2 – метод, измеряющий зависимость между стохастическими переменными и в результате отсеивающий те признаки, которые с наибольшей вероятностью будут независимы от класса [4]. Recursive feature elimination (RFE) – метод, выбирающий наиболее важные признаки путем рекурсивного рассмотрения все меньших и меньших наборов признаков [4]. Процедура рекурсивно повторяется до тех пор, пока не будет достигнуто заданное количество признаков. Для корректного сравнения методов отбора признаков будем каждым методом пытаться сократить размерность признакового пространства до 10000. В табл. 3 указаны результаты классификации с использованием указанных методов отбора признаков в сравнении с примененным ранее частотным фильтром.

Таблица 3

Результаты классификации с использованием различных методов отбора признаков

Методы отбора признаков	TF-IDF со стеммингом + SVC	
	F_m	F_w
Частотный фильтр	0.7466	0.8971
VarianceThreshold	0.7449	0.8966
χ^2	0.7445	0.8958
RFE	0.7476	0.8979

Таким образом, RFE – оказался лучшим методом по отбору признаков в условиях данной задачи. Теперь протестируем конечную матрицу TF-IDF на различных классификаторах [4]. В табл. 4 указаны результаты классификации для конечной матрицы TF-IDF.

Таблица 4

Результаты классификации для конечной матрицы TF-IDF

Методы классификации	TF-IDF со стеммингом + RFE	
	F_m	F_w
LogisticRegression	0.5039	0.8056
SVC	0.7476	0.8979
KNN	0.5521	0.7227
DecisionTreeClassifier	0.6209	0.8177
RandomForestClassifier	0.6622	0.8447

AdaBoostClassifier	0.6636	0.8394
MLPClassifier	0.8146	0.9149

Таким образом, MLPClassifier [4] показал лучший результат в TF-IDF подходе, получив F1-macro = 0.8146 и F1-weighted = 0.9149.

3. GloVe

GloVe – это метод основанный на векторном представлении слов [5]. Идея таких подходов основана на дистрибутивной гипотезе, состоящей в том, что смысл слова заключается в том, среди каких слово оно чаще всего встречается [5]. Таким образом, при обучении модели формируются векторы для каждого слова так, что если семантические векторы двух слов «близки» друг к другу, то скорее всего эти векторы принадлежат словам, близким по смыслу в человеческом понимании [9]. На практике для классификации текстов зачастую используют усреднение всех векторов текста и классифицируют полученный усредненный вектор.

Для рассматриваемой задачи используем предобученную на текстовом корпусе Википедии модель из 400000 слов с векторами размера 300 [5]. Получим векторы для слов в трех используемых типах текстового описания и усредним их, чтобы получить векторное представление документов. Проверим получившиеся данные на SVC и MLPClassifier, как на наиболее хорошо работающих классификаторах в условиях данной задачи. В табл. 5 указаны результаты классификации при помощи пятикратной перекрестной проверки с сохранением распределения по классам на указанных классификаторах.

Таблица 5

Результаты классификации векторных представлений текстов

Типы текстового описания	SVC		MLPClassifier	
	F_m	F_w	F_m	F_w
Текст без изменений	0.6910	0.8519	0.6866	0.8442
Текст со стеммингом	0.6823	0.8420	0.6778	0.8335
Текст с лемматизацией	0.6848	0.8488	0.6802	0.8424

Таким образом, в данном подходе лучше себя проявил вариант текста без изменений. Так как максимальный размер вектора равен 300, то в попытке увеличить его был рассмотрен вариант разделения названия продукта и описания, чтобы по отдельности получить усредненные векторы названия продукта и описания, а затем конкатенировать их в вектор размера 600. В табл. 6 указаны результаты классификации данного подхода с использованием неизмененного текста.

Таблица 6

Результаты классификации конкатенированного векторного представления названия и описания на неизмененном тексте

Тип текстового описания	SVC		MLPClassifier	
	F_m	F_w	F_m	F_w
Текст без изменений	0.7471	0.8811	0.7396	0.8762

Таким образом, результат улучшился, но все равно оказался хуже, чем в подходе TF-IDF.

4. BERT и модификации

BERT – это нейронная сеть от Google на основе архитектуры Transformer [2]. BERT был предварительно обучен на огромном корпусе из более чем 2.5 миллиардов слов. В отличие от рассмотренных ранее подходов BERT представляет собой контекстно-зависимую модель. Это означает, что, например, слово «банка», которое может иметь совершенно разные значения («вышел из банка», «банка огурцов»), рассмотренные ранее методы идентифицируют единственным способом, а BERT и её модификации сгенерирует представление слова в зависимости от окружающих слов, т.е. от контекста.

Рассмотрим четыре модели: Bert-base-uncased, Roberta-base, Deberta-base – модификация BERT от Microsoft и Bert-large [10]. Исследуем случай подачи текста без изменений. В табл. 7 указаны результаты классификации на данных четырех моделях.

Таблица 7

Результаты классификации на BERT и его модификациях

Модели BERT	Текст без изменений	
	F_m	F_w
Bert-base-uncased	0.8380	0.9323
Roberta-base	0.8355	0.9258
Deberta-base	0.8275	0.9213
Bert-large	0.8512	0.9360

5. Сравнение результатов рассмотренных подходов

Подводя итоги, сравним лучшие результаты каждого из подходов. В табл. 8 указаны лучшие результаты классификации в каждом из трех подходов.

Таблица 8

Итоговые результаты классификации

Подходы	Текст без изменений	
	F_m	F_w
TF-IDF	0.8146	0.9149
Glove	0.7471	0.8811
BERT	0.8512	0.9360

Таким образом, модель Bert-large показала лучший результат, и кроме того, каждая рассмотренная модификация BERT превзошла лучшие результаты, полученные в других подходах.

Заключение

В ходе работы была рассмотрена задача многоклассовой классификации текстов с большим количеством (около 350) несбалансированных классов на примере задачи категоризации продуктов по их названию и описанию. Также были проанализированы и опробованы на практике различные методы предобработки данных (лемматизация, стемминг) и модели представления (TF-IDF, GloVe, BERT). По итогу проделанной работы лучшие результаты были получены с использованием модели BERT: macro F1-score = 0.8512, weighted F1-score = 0.9360, что подтверждает на практике эффективность модели BERT в задачах с большим количеством несбалансированных классов.

Список литературы

1. A. Mohamed: Survey on Multiclass Classification Methods // Technical Report, Caltech – November 2005
2. A. Vaswani: Attention Is All You Need // Computer Science – 2017
3. J. Devlin: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // Computer Science – 2018
4. Scikit-learn [Электронный ресурс]: Сайт, посвященный машинному обучению на Python – Режим доступа: https://scikit-learn.org/stable
5. https://nlp.stanford.edu [Электронный ресурс]: Сайт, посвященный обработке естественного языка / Стенфордский университет – Режим доступа: https://nlp.stanford.edu/projects/glove/
6. https://www.nltk.org/ [Электронный ресурс]: Сайт, посвященный обработке естественного языка – Режим доступа: https://www.nltk.org/
7. J. Plisson, N. Lavrac, D. Mladenić: A Rule based Approach to Word Lemmatization // In the Proceeding of the SiKDD at multiconference IS. – Slovenia, 2004
8. I. Smirnov: Overview of stemming algorithms // Mechanical Translation. – 2008
9. J. Pennington: Global Vectors for Word Representation // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) – Qatar, 2014 – P. 1532-1543.
10. Huggingface [Электронный ресурс]: Сайт, посвященный машинному обучению – Режим доступа: https://huggingface.co/models