Исследование методов нечеткого сравнения строк и их применение в алгоритме поиска опечаток в тексте

А. А. Потапова, e-mail: kaisdilmah@gmail.com

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский Авиационный Институт (национальный исследовательский университет)»

Аннотация. *В работе рассмотрены методы обнаружения опечаток в тексте. Приведен анализ методов сравнения строк. Разработан алгоритм на основе выбранного метода и даны дальнейшие направления по его усовершенствованию с помощью правил на основе статистических данных о частоте появления опечаток в текстах.*

Ключевые слова: *алгоритм, нечеткий поиск, опечатка.*

Введение

Современные компьютеры являются мощными устройствами для работы с текстом, они осуществляют его хранение, передачу и обработку. Согласно статистике каждую минуту происходит создание порядка 16 миллионов текстовых сообщений, больше половины человечества используют электронную почту, отправляя ежедневно 267 миллиардов электронных писем, и число пользователей только увеличивается, так к 2023 году ожидается их рост до 4.3 миллиарда. В среднем 1 человек в день отправляет 36 email-сообщений. При этом почти 60 % электронных посланий содержат опечатки или написаны с ошибками [7].

Автоматическая проверка орфографии — одна из актуальных проблем в области обработки естественного языка, универсального решения для нее до сих пор не представлено, однако на протяжении всей своей истории коррекция орфографии являлась актуальной задачей прикладной лингвистики.

Две основные потребности данной области, которые определяют ее развитие: текст, который подвергли корректуре более удобен для дальнейшей автоматической обработки, например, различными системами по извлечению фактов, оптимизация поисковой выдачи и др. и удобством применения для пользователя (мобильная коррекция орфографии, автоматическая проверка орфографии в текстовых редакторах и др.).

1. Обзор методов нечеткого сравнения строк

Нечеткое сравнение строк, или просто неточное сравнение – это процесс поиска похожих, но необязательно в точности совпадающих строк. Существуют три основных группы методов сравнения строк: методы, основанные на множестве общих символов, методы на основе редакционного расстояния и методы, использующие N-граммное редакционное расстояние.

Метод Жаккара [3] или коэффициент сходства основан на идее того, что чем больше в строках одинаковых символов, тем больше вероятность того, что они похожи. Этот коэффициент вычисляется путем деления количества символов общих для двух строк на общее число символов в обеих строках. Тогда, представив множество символов первой строки как A, а множество символов второй строки как B, выразим меру Жаккара следующей формулой:

$$K_J = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

Основной недостаток методов на основе множества общих символов состоит в том, что невозможно учесть порядок символов, поэтому в результате вывод одинаков для одинаковых строк и для инверсированных. Метод Джаро-Винклера или расстояние Джаро-Винклера решает эту проблему путем ограничения сравнений окном символов второй строки, которое вычисляется на основе большей длины среди строк.

Вычисление редакционного расстояния – еще один метод определения того, насколько схожи строки, вычисляется количество операций редактирования, которые необходимо проделать, прежде чем строки станут идентичными. Стандартный набор операций состоит из вставки, удаления и замены символов. Простейший вид редакционного расстояния, где каждая операция имеет единичный вес, называется расстоянием Левенштейна [4].

Вычисление минимальной последовательности производится путем n × m сравнений, где n – длина одной строки, а m – длина другой.

На рисунке 1 представлена матрица расстояний для строк “поливаю куст” и “полваю кутт”, содержимое ячейки – это минимальное количество операций редактирования для получения из одной строки другой. Минимальное расстояние находится в правом нижнем углу и в данном примере оно равно 2.

		п	о	л	и	в	а	ю		к	у	с	т
	0	1	2	3	4	5	6	7	8	9	10	11	12
п	1	0	1	2	3	4	5	6	7	7	8	9	10
о	2	1	0	1	2	3	4	5	6	7	8	9	10
л	3	2	1	0	1	2	3	4	5	6	7	8	9
в	4	3	2	1	1	1	2	3	4	5	6	7	8
а	5	4	3	2	2	2	1	2	3	4	5	6	7
ю	6	5	4	3	3	3	2	1	2	3	4	5	6
	7	6	5	4	4	4	3	2	1	2	3	4	5
к	8	6	6	5	5	5	4	3	2	1	2	3	4
у	9	7	7	6	6	6	5	4	3	2	1	2	3
т	10	8	8	7	7	7	6	5	4	3	2	2	3
т	11	8	9	8	8	8	7	6	5	3	3	3	2

Рис. 1. Матрица вычисления редакционного расстояния

Распространенный вариант расстояния Левенштейна – расстояние Дамерау-Левенштейна, в котором дополнительно опускается операция перестановки соседних букв, то есть, вместо реализации через две операции удаления и вставки с итоговым весом 2, назначается вес 1. Алгоритм Вагнера-Фишера позволяет вычислить кратчайшее расстояние Левенштейна. В N-граммном редакционном расстоянии использована идея расстояния Левенштейна, только символом является N-грамма.

Алгоритм шинглов был разработан для поиска копий и дубликатов рассматриваемого текста в документе. Шинглом называется фрагмент, состоящий из нескольких слов текста, обработанного для анализа. Алгоритм шинглов (w-shingling) позволяет обрабатывать входные данные, используя набор шинглов (N-грамм, смежных подпоследовательностей токенов в строках). Реализация включает несколько этапов: нормализацию строк, разбиение на шинглы, нахождение контрольных сумм и поиск совпадений последовательностей.

2. Виды ошибок в тексте

В исследовании поисковых запросов Яндекса выделено два наиболее распространенных типа ошибок: 1) опечатки, то есть случаи, когда человек промахивается мимо нужной клавиши или вводит запрос в неправильной раскладке; 2) орфографические ошибки [6]. В таблице 1 представлены наиболее частые ошибки, встречающиеся у пользователей.

Таблица 1

Наиболее распространенные ошибки

Номер	Вид ошибки	Частота (в %)
1	Замена “и” - “е”	27
2	Замена “а” - “о”	25
3	Лишний пробел, слово должно быть написано слитно	9.1
4	Отсутствие пробела, вместо одного слова два	8.5
5	Потеря одной из удвоенных букв	6.6
6	Замена глухой буквы звонкой и наоборот	3.6
7	Гласные после ц	2.7
8	Удвоение одиночной буквы	2.6
9	Потеря “ь”	1.3
10	Лишний “ь”	0.6
11	Замена “ё” - “е”	0.1

Около 74% опечаток поддаются исправлению средствами автоматической коррекции, так как не зависят от контекста предложения, поэтому, благодаря обнаруженным наиболее распространенным видам ошибок, может быть увеличена точность выбранного методов сравнения строк.

3. Предлагаемый алгоритм поиска опечаток

После получения на вход двух строк для сравнения, осуществляется проверка их схожести с помощью алгоритма шинглов, если различий не найдено, то работа алгоритма завершается, иначе начинается следующий шаг и определяется редакционное расстояние.

Проблема нечеткого сравнения полученных данных сводится к проблеме нечеткого сравнения множества строк. Для нечеткого сравнения строк используется метод на основе вычисления значений расстояния Дамерау – Левенштейна. Чтобы определить расстояние Дамерау – Левенштейна, используется алгоритм Вагнера – Фишера.

Если полученное редакционное расстояние строк будет равно 1 или меньше, то можно начать проверку по правилам, приведенным в табл.1 и провести сравнение с данными морфологической библиотеки фреймворка TAWT [8], иначе с большой долей вероятности строки являются различными. На рисунке 2 представлена блок-схема разработанного алгоритма.

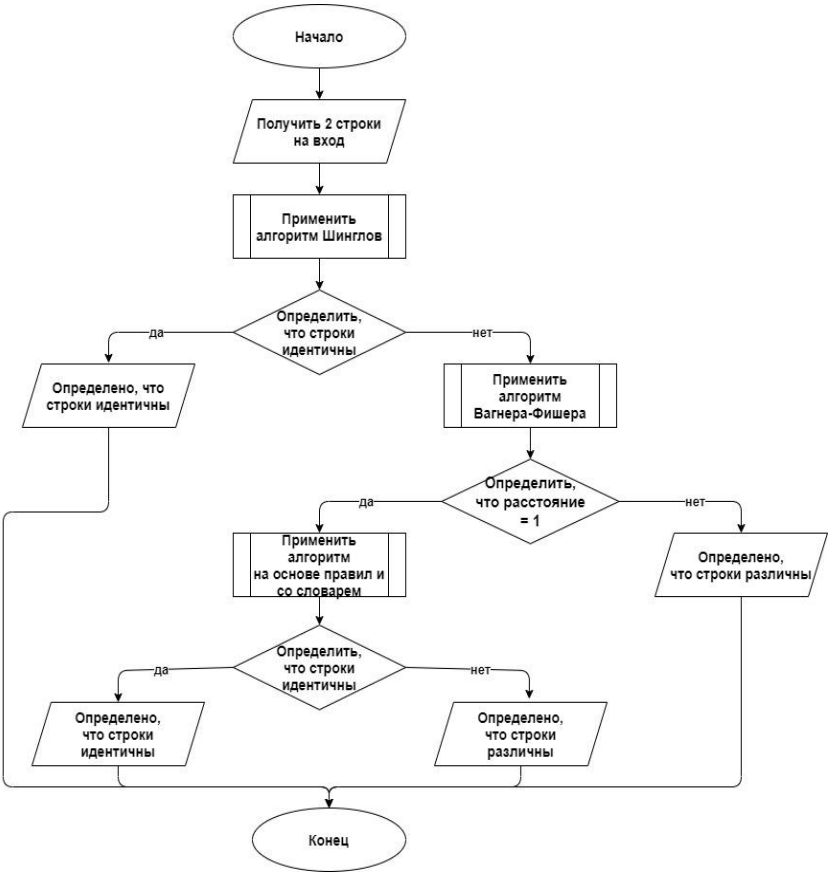

Рис. 2. Блок-схема алгоритма поиска опечаток

Заключение

В статье проанализированы методы нечеткого сравнения строк, а также предложен способ улучшения точности одного из них с помощью выделения правил на основе частоты распространения конкретных

ошибок. Предложен алгоритм нечеткого сравнения строк на основе совместного использования методов шинглов, Вагнера-Фишера и проверке по словарям. Реализация алгоритма позволит улучшить качество работы существующих систем автоматизированного анализа текста за счет поиска и исправления опечаток во входных текстах.

Литература

1. Мортон, С. Т. Обработка неструктурированных текстов. Поиск, организация и манипулирование. / С. Т. Мортон, Г. С. Игрерсолл, Э.Л. Фэррис// ДМК Пресс, 2015. – 414 с.
2. Прикладная и компьютерная лингвистика. / ред. Митренина О. В., Ландо Т. М. // Ленанд, 2016. – 320 с.
3. String Matching Algorithms and their Applicability in various Applications. / International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-I, Issue-6, January 2012
4. G. Navarro / Faster Approximate String Matching. / R. Baeza-Yates, G. Navarro // Algorithmica (1999) 23: 127–158
5. Paramonov, V.V. Polyphon: An Algorithm for Phonetic String Matching in Russian Language / V.V. Paramonov, A.O. Shigarov, G.M. Ruzhnikov, P.V. Belykh // International Conference on Information and Software Technologies. – 2006. – С. 568-579
6. Исследования Яндекс [Электронный ресурс]: – Электрон. журн. – Режим доступа: yandex.ru/company/researches/2012/ya_orfo [Дата обращения 20.12.2020]
7. B. Marr [Электронный ресурс]: – Электрон. журн. – Режим доступа: https://www.bernardmarr.com/default.asp?contentID=1438 [Дата обращения 20.12.2020]
8. Фреймворк TAWT [Электронный ресурс]: – Режим доступа: https://textanalysis.ru/jce/details/tawt [Дата обращения 20.12.2020]