УДК 004.822
DOI: 10.22213/2410-9304-2024-2-103-113

# Методы и алгоритмы для поиска сходства между текстам

*И. М. Янников,* доктор технических наук, доцент, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия
*М. В. Ершова,* кандидат технических наук, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия
*А. Н. Исенбаев,* аспирант, ИжГТУ имени М. Т. Калашникова, Ижевск, Россия

*Обзорная аналитическая статья представляет собой комплексное исследование современных методов анализа текстов с целью выявления и измерения степени их сходства, что само по себе является весьма важной и актуальной задачей, поскольку рассматривает и анализирует инструментарий, применяемый для ее решения. Во введении рассматриваются цель данной работы, актуальность проблемы и важность разработки эффективных методов для сравнения текстов.*

*В основной части статьи отдельно рассматриваются и анализируются такие методы, как сходство Жаккара, алгоритм шинглов, расстояние Левенштейна, TF-IDF и BM25, BERT и использование нейросетей. Применение того или иного метода проиллюстрировано примерами, представленными в табличной форме и в виде иллюстраций. При рассмотрении и анализе сходства Жаккара отражаются способы его применения и ограничений. При анализе алгоритма шинглов выявляются преимущества метода в контексте поиска сходства.*

*В публикации подробно рассматриваются методы, основанные на расстоянии между строками, включая расстояние Левенштейна. При этом особое внимание уделяется области его применения и имеющимся преимуществам по сравнению с другими методами. При рассмотрении статистических методов, таких как TF-IDF и BM25, дается анализ их применения и эффективности в поиске сходства текстов.*

*Статья не ограничивается анализом только традиционных методов, но и охватывает современные, включая BERT и использование нейросетей. Производится сравнение данных категорий методов между собой, выявляются их преимущества и недостатки использования.*

*В разделе выводов проводится сравнительный анализ всех представленных методов по принципу объективности, выделяя их характеристики и области применения. Отмечается важность выбора наиболее подходящего метода поиска сходства текстов в зависимости от конкретных целей поиска, поставленных задач и требований, а также дается заключение о наиболее применяемом, широком и продуктивном методе – использовании нейросетей. В выводах подчеркивается, что статья, посвященная сравнительному анализу различных методов поиска сходства между текстами, преследует главную цель – разработку рекомендаций по выбору оптимального способа.*

**Ключевые слова:** алгоритмы поиска сходства, обработка текстов, сравнение текстов, сравнение уникальности, нейронные сети, искусственный интеллект, алгоритм шинглов.

## Введение

В эпоху широкого информационного поля и неограниченного доступа к данным вопрос эффективного анализа текстовых материалов становится все более актуальным и значимым. Актуальность рассматриваемого вопроса подтверждается необычайно быстрыми темпами внедрения цифровизации во все сферы человеческой деятельности и возникающим при этом большим количеством различных проблем, связанных с обработкой массивных объемов данных, подделкой сведений и документов, спамом и пр. Поэтому сравнение текстов, выявление схожести и измерение степени их схожести – важные аспекты для многих сфер деятельности, начиная от информационного поиска и заканчивая анализом больших объемов данных.

Сложность выбора методик заключается в том, чтобы разработать такие методы, которые не только корректно определят степень схожести, но и смогут адаптироваться к различным типам текстов и учесть специфику задачи. Решению данной проблемы посвящены работы целого ряда авторов [1–3].

Статья предлагает обзор современных подходов, начиная с классических методов, таких как сходство Жаккара и алгоритм шинглов, и заканчивая современными технологиями вроде BERT и нейросетевых методов. В ходе исследования рассмотрены особенности каждого метода, выявлены их преимущества и ограничения, а также предложена оценка объективности использования методов и рекомендации по их применению, что и является целью данной работы.

## Сходство Жаккара

Метод сходства Жаккара был впервые предложен Полем Жаккаром в 1901 году. Этот метод используется для измерения сходства между двумя наборами данных.

Сходство Жаккара вычисляется как отношение числа общих элементов в двух наборах дан-

ных к общему числу элементов в каждом из наборов данных. Если два набора данных имеют одни и те же элементы, их индекс сходства Жаккара будет равен 1. Если у них нет общих элементов, их сходство будет равно 0.

В контексте поиска сходства в тексте сходство Жаккара может быть использовано для оценки сходства между двумя текстами, заменяя «целые числа» на «токены».

Однако стоит отметить, что сходство Жаккара может дать высокие оценки даже для текстов, которые не имеют смыслового сходства, поскольку оно основывается только на наличии общих элементов. Эти недостатки могут быть частично устранены с помощью методов предварительной обработки, таких как удаление стоп-слов, стемминг/лемматизация и пр. [4].

Преимущества метода Жаккара для поиска сходства между текстами включают простоту и быстроту вычисления, а также возможность использования для любых типов текстов. Коэффициент Жаккара позволяет измерить сходство между двумя множествами, в данном случае множествами уникальных слов в текстах [5]. Этот метод может быть полезен для обнаружения плагиата, поиска дубликатов текстов, кластеризации текстов и других задач.

Из недостатков метода Жаккара для поиска сходства между текстами можно отметить:

– нечувствительность к порядку слов: метод Жаккара сравнивает наборы данных, но не учитывает порядок элементов, что может привести к тому, что два текста, которые имеют все те же слова, но в другом порядке, будут считаться не схожими;

– нечувствительность к семантике: метод Жаккара не учитывает семантику слов, например слова «кот» и «собака» могут быть синонимами в одном контексте, но не в другом, то есть метод Жаккара не сможет учесть эту семантическую связь;

– нечувствительность к частным случаям: метод Жаккара может дать высокие оценки даже для текстов, которые не имеют смыслового сходства, поскольку он основывается только на наличии общих элементов;

– недостаточность для больших текстов: метод Жаккара может быть неэффективным для больших текстов, поскольку он сравнивает наборы данных, а не отдельные элементы, что может привести к тому, что важные элементы будут пропущены, если они не встречаются достаточно часто;

– неэффективность для коротких текстов: метод Жаккара также может быть неэффективным для коротких текстов, поскольку он может дать высокие оценки для текстов, которые имеют только одно общее слово [6].

Учитывая приведенные аргументы, можно сделать вывод об отсутствии широкого спектра применения методики, что характеризует его как весьма ограниченный, поскольку Метод Жаккара нельзя применять относительно любых текстов.

**Алгоритм шинглов**

Многие системы анализа текстов используют модули для проверки сходства документов, основанные на алгоритме шинглов. Этот алгоритм обычно используется для улучшения результатов поиска, исключая документы, которые уже были найдены, и для обнаружения плагиата. Реализация этого алгоритма включает следующие шаги:

1. Канонизация текста, что означает преобразование текста в единый формат, удаление ненужных символов и применение стандартных правил форматирования.

2. Разделение текста на части или шинглы.

3. Нахождение контрольных сумм для каждого шингла.

4. Поиск совпадающих последовательностей шинглов между различными документами.

В контексте программирования канонизация текста обычно включает в себя преобразование текста в нижний регистр, удаление специальных символов и знаков пунктуации, а также применение процессов стемминга и лемматизации для упрощения текста перед его анализом.

Процесс разбиения текста на шинглы на втором этапе включает в себя выделение последовательностей слов, которые следуют друг за другом, обычно на количество слов до десяти. Важно отметить, что для достижения лучших результатов выборка осуществляется внутрь каждого шингла, а не между ними. Пример выделения шинглов внахлест представлен на рис. 1 [7].

На третьем этапе для каждого шингла находится его контрольная сумма (хэш-функции crc32, md5 и др.). Нахождение контрольных сумм позволяет сгенерировать уникальный идентификатор для каждого шингла. Это помогает сократить объем информации, необходимой для сравнения, и ускоряет процесс поиска схожести.

Последний этап – поиск одинаковых последовательностей – выполняется путем сравнения контрольных сумм шинглов в разных текстах. Если контрольные суммы совпадают, значит, эти шинглы совпадают и, следовательно, тексты содержат повторяющуюся информацию.

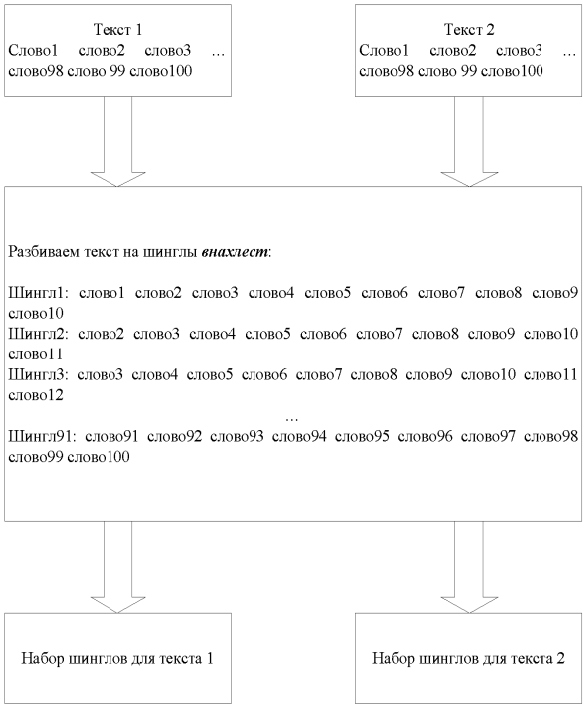


*Рис. 1.* Выделение шинглов из текстов

*Fig. 1.* Isolating shingles from texts

Данный алгоритм показывает достаточно низкую точность. Это напрямую связано с хэш-функциями. Достаточно изменить один символ в тексте, как контрольная сумма шингла полностью изменяется. Именно по этой причине применение данного метода сильно ограничено, что лишает его универсальности и объективности.

**Расстояние Левенштейна**

Расстояние Левенштейна, также известное как редакционное расстояние, представляет собой метрику, которая используется для измерения различий между двумя строками. Это минимальное количество операций (вставка, удаление или замена символов), которые требуются для преобразования одной строки в другую. Чем меньше расстояние Левенштейна, тем ближе похожи исходная и целевая строки.

Другими словами, расстояние Левенштейна – это число, которое показывает, насколько сильно отличаются две строки. Чем больше это число, тем больше различий между строками. Для двух идентичных строк расстояние равно нулю. В действительности, это минимальное количество преобразований символа, необходимое для преобразования одной строки в другую.

Расстояние Левенштейна широко применяется в различных областях. Например, оно может быть использовано в поисковых системах, проверке орфографии или сравнении геномных последовательностей.

Пример сравнения слов «Австрия» и «Австралия» с помощью метода Левенштейна (табл. 1) показывает, что расстояние между ними равно двум.

*Таблица 1.* **Пример сравнения слов «Австрия» и «Австралия» с помощью метода Левенштейна**

*Table 1.* **An example of comparing the words "Austria" and "Australia" using the Levenshtein method**

| А | В | С | Т | Р | – | -- | И | Я |
|---|---|---|---|---|---|---|---|---|
| А | В | С | Т | Р | А | Л | И | Я |

Алгоритм Левенштейна используется для определения минимального количества операций (вставок, удалений или замен), которые нужно выполнить над одной строкой, чтобы получить другую строку [8]. Это достигается с помощью динамического программирования следующим образом:

1. Создается таблица, где каждая ячейка представляет собой расстояние Левенштейна между префиксами двух строк.
2. Таблицу можно заполнять слева направо и сверху вниз.
3. Горизонтальные и вертикальные переходы соответствуют операциям вставки и удаления.
4. Стоимость каждой операции равна 1.
5. Диагональный переход может стоить либо 1, если символы в этой позиции не совпадают, либо 0, если они совпадают. В любом случае, каждая клетка минимизирует стоимость локально.

В результате число в правом нижнем углу этой таблицы представляет собой расстояние Левенштейна между двумя строками [2, 5].

На рис. 2 приведен пример сравнения слов «HONDA» и «HYUNDAI», расстояние между которыми равно 3.

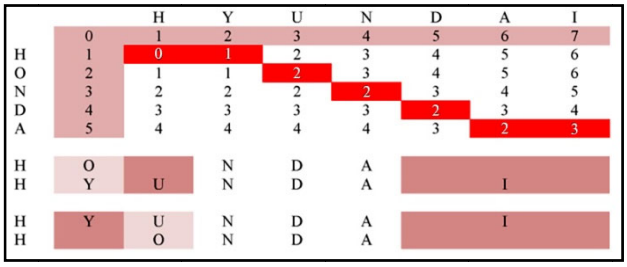


*Рис. 2.* Расстояние между «HONDA» и «HYUNDAI»

*Fig. 2.* Distance between HONDA and HYUNDAI

При использовании метода приближенного совпадения строк задача заключается в поиске коротких строк, которые совпадают или имеют минимальное количество отличий от более длинных текстов. Эти короткие строки могут

быть получены, например, из словаря, где одна строка короткая, а другая может быть любой длины. Этот подход широко используется в различных областях, включая программы проверки орфографии, системы коррекции для визуального распознавания символов, а также программное обеспечение для облегчения перевода на естественный язык на основе памяти переводов.

Однако расстояние Левенштейна, которое измеряет минимальное количество единичных операций редактирования (вставки, удаления или замены символов), необходимых для преобразования одной строки в другую, может быть вычислено только между короткими строками. Вычисление этого расстояния между двумя длинными строками является непрактичным из-за высокой стоимости, которая прямо пропорциональна произведению длин этих строк.

Алгоритм Левенштейна часто используется для проверки качества машинного перевода [4]. Для сравнения двух текстов различной длины используется показатель Distanceratio, который представляет собой отношение расстояния Левенштейна к сумме длин текстов. Этот показатель будет уменьшаться в основном за счет взаимного сокращения однокоренных слов в обоих текстах, что отражает разницу в лексическом наборе текстов. Данный метод подходит для задач перевода, но его применение в широком спектре задач не предполагается. Использование алгоритма Левенштейна самостоятельно, без дополнительных методик, лишает результаты объективности и точности.

**Алгоритм TF-IDF**

TF-IDF – метод, который позволяет оценить важность определенного слова в контексте всего текстового корпуса. Этот подход был разработан Джорджем Солтоном в середине 1970-х годов. Основной идеей TF-IDF является соотношение частоты использования слова в конкретном документе и обратное соотношение частоты его появления во всех документах. Если слово используется в данном документе чаще, чем в других, значит, оно имеет для него большую релевантность – именно такой критерий релевантности документа по запросу был у первых аналитиков поисковых систем [9].

TF (Term Frequency), или частота слова, отражает, насколько часто слово встречается в документе. Однако, если слово встречается чаще в одном документе, чем в другом, это не обязательно означает, что оно более важно для этого документа. Для учета этого фактора используется отношение количества использованных слов к общему количеству слов в документе.

IDF (Inverse Document Frequency), или обратная частота документа, помогает уменьшить вес слов, которые встречаются часто во всех документах. Эта часть формулы учитывает редкость слова в корпусе. Если слово встречается редко во всех документах, это означает, что оно имеет большую значимость для документа, где оно встречается.

Вместе эти два компонента позволяют создавать весовые коэффициенты для каждого слова в документе, которые затем могут быть использованы в алгоритмах машинного обучения для обработки естественного языка.

Показатель *TF-IDF* представляет собой произведение двух факторов:

$$TF - IDF(t,d,D) = TF(t,d) * IDF(t,D). \quad (1)$$

Слова с высокой частотой встречаемости в конкретном документе и низкой частотой использования в других документах будут иметь высокий вес в *TF-IDF*.

Основание логарифма может быть любым, так как IDF – относительная мера.

Возьмем условный документ X, состоящий из 456 слов и содержащий 2 вхождения термина «компьютер». TF для этого слова будет:

$$TF = \frac{2}{456} = 0,004386.$$

Рассчитаем IDF, предполагая, что в выборке всего 210 420 документов и 10 241 из них содержит термин «компьютер»:

$$IDF = \log\frac{210420}{10241} = 1,312745.$$

Умножив TF на IDF, получаем оценку важности термина «компьютер» для документа X: TFIDF=0,004386 – 1,312745=0,005758.

Другой пример: имеется 4 условных документа со следующим содержанием [пример из 2]:

1. Текстовая релевантность – это соответствие текста страницы или всего сайта определенному поисковому запросу. Чем выше релевантность текста, тем больше вероятность того, что ваша страница попадет в первую строку результатов поиска при прочих равных условиях.

2. Релевантность текста в поиске – это мера соответствия всех внутренних факторов страницы запросу пользователя. К внутритекстовым факторам относятся: заголовки, содержание текста, форматирование текста, атрибуты тегов.

3. Релевантность в общем смысле – это соответствие документа ожиданиям пользователя. Таким образом, релевантность в поиске – это степень удовлетворенности пользователя результатами поиска, которые появляются в ответ

на его запрос. Она рассчитывается с помощью алгоритмов поисковых систем.

4. Соответствие документа запросам пользователей, а также соблюдение правил поисковой системы позволяют сайту занимать более высокие позиции в поиске.

Расчет TF-IDF для этих документов будет с ключевыми словами:

– релевантность;

– релевантность текста;

– релевантность текста документа;

– текстовая релевантность документа в поиске.

По данной методике приведем в качестве примера расчеты, взятые из открытых источников [10].

Общее количество слов в документах и количество вхождений каждого слова из запроса в документ приведено в табл. 2–4:

*Таблица 2.* **Общее количество слов в документах и количество вхождений**

*Table 2.* **Total number of words in documents and number of occurrences**

| № документа | Всего слов | Текстовая | Релевантность | Документа | Поиске |
|---|---|---|---|---|---|
| 1 | 25 | 2 | 2 | 0 | 0 |
| 2 | 22 | 1 | 1 | 0 | 1 |
| 3 | 26 | 0 | 3 | 1 | 1 |
| 4 | 17 | 0 | 0 | 1 | 1 |
| Итого: | | 2 | 3 | 2 | 3 |

*Таблица 3.* **Расчет множителей TF и IDF для каждого слова**

*Table 3.* **Calculation of TF and IDF factors for each word**

| Слова | TF | | | | IDF |
|---|---|---|---|---|---|
| | 1 док | 2 док | 3 док | 4 док | |
| Текстовая | 0,080 | 0,045 | 0,000 | 0,000 | 0,301 |
| Релевантность | 0,080 | 0,045 | 0,115 | 0,000 | 0,125 |
| Документа | 0,000 | 0,000 | 0,038 | 0,059 | 0,301 |
| Поиске | 0,000 | 0,045 | 0,308 | 0,059 | 0,125 |

*Таблица 4*. **Полученные результаты TF-IDF**

*Table 4.* **TF-IDF resultsobtained**

| Слова | TF-IDF | | | |
|---|---|---|---|---|
| | 1 док | 2 док | 3 док | 4 док |
| Текстовая | 0,024 | 0,014 | 0,000 | 0,000 |
| Релевантность | 0,010 | 0,006 | 0,014 | 0,000 |
| Документа | 0,000 | 0,000 | 0,012 | 0,018 |
| Поиске | 0,000 | 0,006 | 0,005 | 0,007 |

Вес запросов относительно документов рассчитывается как сумма TF-IDF каждого слова из запроса (табл. 5).

Вопрос «Релевантность» наиболее важен в 3-м документе, что неудивительно, поскольку он содержит наибольшее количество экземпляров. По аналогичной причине вопрос «релевантность текста» наиболее важен в 1-м документе.

*Таблица 5.* **Расчет веса запросов относительно документов**

*Table 5.* **Calculation of the weight of requests relative to documents**

| Запросы | Σ(TF·IDF) | | | |
|---|---|---|---|---|
| | 1 док | 2 док | 3 док | 4 док |
| Релевантность | 0,010 | 0,006 | 0,014 | 0,000 |
| Текстовая релевантность | 0,020 | 0,011 | 0,010 | 0,000 |
| Текстовая релевантность документа | 0,020 | 0,011 | 0,031 | 0,018 |
| Текстовая релевантность документа в поиске | 0,020 | 0,017 | 0,036 | 0,025 |

Первый документ не содержит слов «документ», «поиск», но все равно содержит большее значение, чем у остальных по TF-IDF. Стоит отметить, что абсолютно все искомые слова не содержат ни одного документа. Больше всего – 3 слова из 4 – содержит третий документ. Результаты после добавления слова «текст» в третий документ представлены в табл. 6.

*Таблица 6.* **Результаты расчетов после добавления слова «текст» в третий документ**

*Table 6.* **Calculation results after adding the word “text” to the third document**

| Запросы | Σ(TF·IDF) | | | |
|---|---|---|---|---|
| | 1 док | 2 док | 3 док | 4 док |
| Релевантность | 0,010 | 0,006 | 0,014 | 0,000 |
| Текстовая релевантность | 0,034 | 0,020 | 0,014 | 0,000 |
| Текстовая релевантность документа | 0,034 | 0,020 | 0,026 | 0,018 |
| Текстовая релевантность документа в поиске | 0,034 | 0,026 | 0,031 | 0,025 |

Результаты изменились в пользу документа 3, за ним следует документ 4, а первый занимает третье место. В то же время в документах 4 и 1 не произошло никаких изменений. Если удалить предыдущее изменение и добавить слово «Релевантность» в документ 4, IDF для этого запроса будет равняться 0, потому что запрос есть во всех документах. Поэтому документ 4 получает оценку TF-IDF для запроса «Релевантность текста документа в поиске».

При увеличении числа вхождений ключевого слова в документ значение TF пропорционально увеличивается. Таким образом, добавление вхождений ключевых слов на страницу значительно повышает ее релевантность.

Функция BM25 была призвана устранить этот недостаток.

**Алгоритм BM25**

BM25 – это функция, которая используется для определения релевантности текстовых до-

кументов. Эту функцию разработали Стивен Робертсон и Карен Спарк Джоунс из Великобритании в 1994 году. BM25 основан на эмпирических данных и предназначен для улучшения результатов работы критерия TF-IDF. 25-й алгоритм в списке показал наиболее точное соответствие между ожидаемым и рассчитанным результатами, отсюда и произошло его название «Best matching», или BM25. BM25 впервые был реализован в поисковой системе Okapi, а затем стал основой для текстовых анализаторов современных поисковых систем.

Стивен Робертсон утверждает, что формула BM25 была получена с помощью вероятностной модели, однако некоторые специалисты считают это «подгонкой» под нужный результат. BM25 включает в себя свободные коэффициенты, которые могут принимать разные значения. Эти коэффициенты подбираются таким образом, чтобы «подогнать» результат работы поиска под уже имеющиеся данные. Документы сначала оценивают асессоры, которые делают вывод о том, что хорошо, а что плохо [11]. Затем на основе этих данных выбирают упомянутые коэффициенты, чтобы расположить документы так же, как это сделали асессоры, – так называемый принцип обезьянки.

В период расцвета SEO, когда для высокой релевантности активно внедрялись ключевые слова в большом количестве, этот способ достижения высоких рейтинговых позиций хорошо работал.

Используя расчеты количества документов, содержащих слова из запроса, и среднюю длину документа в тех текстах, получатся следующие результаты (табл. 7) [10].

Результаты поиска IDF для каждого слова из запроса представлены в табл. 8.

Результатом становится следующее значение TF и оценка Score по запросу «Текстовая релевантность документа в поиске» (табл. 9).

*Таблица 7.* **Результаты расчета количества документов, содержащих слова из запроса и средней длины документа**

*Table 7*. **Results of calculating the number of documents containing words from the query and the average length of the document**

| | Текстовая | Релевантность | Документа | Поиске | |
|---|---|---|---|---|---|
| Количество документов содержащих слово | 2 | 3 | 2 | 3 | |
| | Док.1 | Док.2 | Док.3 | Док.4 | Средняя |
| Длина документа | 25 | 22 | 26 | 17 | 22,5 |

*Таблица 8.* **Результаты поиска IDF для каждого слова из запроса**

*Table 8*. **IDF search results for each word from the query**

| | | Слова из запроса | | | |
|---|---|---|---|---|---|
| | | **Текстовая** | **Релевантность** | **Документа** | **Поиске** |
| Док.1 | Частота слова | 0,080 | 0,080 | 0,000 | 0,000 |
| Док.2 | | 0,045 | 0,045 | 0,000 | 0,045 |
| Док.3 | | 0,000 | 0,115 | 0,038 | 0,038 |
| Док.4 | | 0,000 | 0,000 | 0,059 | 0,059 |
| | IDF | 0 | -0,368 | 0 | -0,368 |

Это наглядный пример того, как IDF может принимать отрицательные значения для слов, которые встречаются больше чем в половине документов. Вместо отрицательного значения IDF учитывалось фиксированное IDF=0,01. Наибольшую оценку получил «документ 3», хотя в классической TF-IDF формуле – документ 1, который теперь имеет самую низкую оценку Score по BM25.

*Таблица 9.* **Результат расчета TF и оценка Score по запросу «Текстовая релевантность документа в поиске»**

*Table 9.* **Result of TF calculation and Score for the query “Text relevance of a document in search”**

| | | **Текстовая** | **Релевантность** | **Документа** | **Поиске** | | |
|---|---|---|---|---|---|---|---|
| Док.1 | TF | 0,107 | 0,107 | 0,000 | 0,000 | Score | 0,00107 |
| Док.2 | TF | 0,068 | 0,068 | 0,000 | 0,068 | Score | 0,00136 |
| Док.3 | TF | 0,000 | 0,147 | 0,051 | 0,051 | Score | 0,00198 |
| Док.4 | TF | 0,000 | 0,000 | 0,104 | 0,104 | Score | 0,00104 |

Недостатки BM-25 заключаются в том, что этот алгоритм не может учитывать:

– взаимного расположения слов. Не принимает во внимание порядок и взаимное расположение слов в тексте. Это может привести к неправильной оценке релевантности, особенно в случаях, когда ключевые слова находятся в близком контексте или имеют определенную логическую связь;

– позиции слова относительно начала документа: не учитывает важность слова, которое находится близко к началу документа. В некоторых случаях, особенно когда ключевое слово находится в первом предложении, это может быть критичным для правильной оценки релевантности;

– различные формы слова. Например, если пользователь ищет «книги», а документ содержит слово «книга», BM25 может неправильно оценить релевантность, не учитывая семантическую связь между словами;

– положение ключевого слова в документе или его важность в определенной зоне документа. В некоторых случаях, например если ключевое слово находится в заголовке или подзаголовке, это может быть важным фактором для оценки релевантности [12].

Данная методика не всегда дает адекватную оценку текста, поэтому не является универсальной и не может применяться без привлечения дополнительных оценочных средств.

**Нейронные сети**

Один из эффективных способов решения сложных проблем – деление ее на более мелкие задачи. Сети – один из подходов к достижению этой цели [13]. Существует большое количество различных типов сетей, но все они характеризуются следующими компонентами: набором узлов и соединениями между узлами.

Узлы можно рассматривать как вычислительные единицы. Они получают входные данные и обрабатывают их для получения выходных данных [14]. Эта обработка может быть очень простой (например, суммирование входных данных) или довольно сложной (узел может содержать другую сеть...).

Соединения определяют информационный поток между узлами. Они могут быть однонаправленными или двунаправленными. И самое главное – взаимодействия узлов через соединения приводят к глобальному поведению сети, которое невозможно наблюдать в элементах сети. Системный эффект означает, что возможности сети превосходят возможности ее элементов, что делает сети очень мощным инструментом [15].

Один тип сетей называется «искусственная нейронная сеть». Искусственный нейрон – это вычислительная модель, созданная по принципу естественных нейронов [16].

На сегодняшний день существует несколько моделей с архитектурой «Трансформер»; наиболее распространенными являются BERT и GPT.

GPT-модель (generativepre-trainingmodel) представляет собой языковую модель, носящую авторегрессионный характер и обладающую односторонним алгоритмом обучения.

Алгоритм GPT первого поколения был обучен на датасете, составленном из статей «Википедии» и литературных произведений, схема алгоритма приведена на рис. 3 [17]. Однако оказалось, что для работы в системах коммуникации с людьми такой набор обучающих данных не подходит и его требуется заменить на массив данных, состоящий из текстов, близких к разговорной речи людей. Дальнейшие версии сети были обучены на постах из интернета.

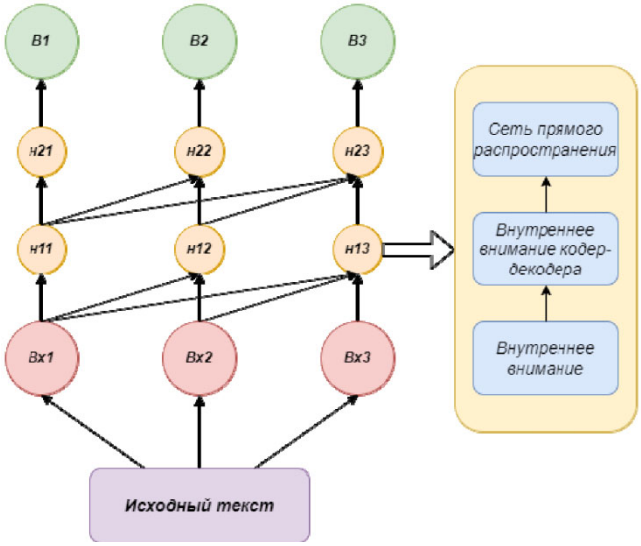


*Рис. 3*. Схема алгоритма GPT

*Fig. 3.* Scheme of the GPT algorithm

Модель GPT показала высокую эффективность в генерации текстов, однако для лучшего сравнения текстов больше подошел алгоритм BERT.

В отличие от традиционных методик сравнения текстов, нейросети шагнули далеко вперед. Главное их достижение – учет интернет-ресурсов, т. е. область проверяемых материалов существенно расширяется. В этой связи нейросети можно назвать весьма удачным выбором для широкого спектра задач, поскольку их результативность заметно повышается.

**Алгоритм BERT**

BERT (Bidirectional Encoder Representations from Transformers) – это модель на основе архитектуры трансформера, которая использует двунаправленный подход к обучению. Это позволяет модели определить контекст слов в обоих направлениях: прямом и обратном. Эта особенность делает BERT уникальным среди других языковых моделей (рис. 3).

Одной из ключевых задач, которые BERT может решать, является прогнозирование замаскированных слов (Masked LM). В процессе обучения модель учится предсказывать слова, которые были случайно замаскированы в исходном тексте, основываясь на контексте окружающих эти слова токенов.

Другой важной задачей, которую BERT может выполнять, является определение логической связи между двумя отдельными предложениями (Next Sentence Prediction – NSP). Модель учится определять, следуют ли два

предложения одна за другой в исходном тексте, или они не имеют прямой связи.

В результате, благодаря двунаправленному обучению, BERT способен эффективно определять контекст слов и предложений, а также их семантическую нагрузку.

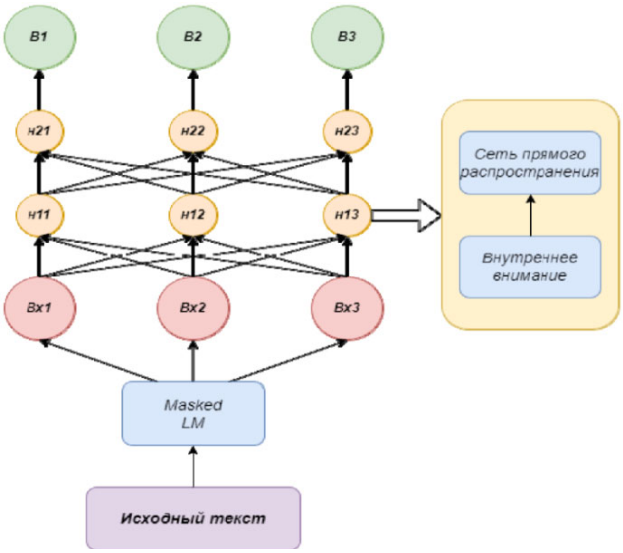


*Рис. 4.* Схема алгоритма BERT

*Fig. 4.* BERT algorithm diagram

Эта модель не требует обработки текста в определенном порядке в документе, что позволяет ей легко распараллеливаться и быстро обучаться. В процессе обучения для каждого документа, представленного во входном наборе данных, BERT учится создавать векторное представление, которое затем передается на вход уровня классификации. Уровнем классификации в этом методе является нейронная сеть прямого распространения [18].

В отличие от других подобных моделей, BERT предназначена для предварительного обучения двунаправленных представлений на неразмеченном текстовом корпусе, обучаясь на контексте слева и справа одновременно [18]. Под двунаправленностью понимается обучение предсказанию токенов в зависимости как от префикса, так и от суффикса, которые окружают маскированное слово. Случайным образом выбирается любой токен входной последовательности, который маскируется, после чего модели необходимо предсказать ее исходное значение, исходя из контекста.

По сравнению с однонаправленными моделями, двунаправленные модели предлагают значительно более широкие возможности предварительного обучения, что приводит к увеличению эффективности языковых моделей в решении задач обработки текстовых данных.

BERT – это наиболее актуальный и объективный подход среди всех. Этот алгоритм обладает способностью к «самообучению» на новейших информационных материалах, что делает его использование наиболее объективным.

**Выводы**

В условиях быстро расширяющихся информационных потоков и множества текстовых данных поиск сходства между текстами становится ключевой задачей для эффективной обработки и анализа информации. В статье рассмотрены разные методы, отличающиеся подходами к решению этой задачи. К традиционным методикам были отнесены: метод Жаккара, алгоритм шинглов и расстояние Левенштейна, хорошо дополняющие друг друга, но не отвечающие изменчивому информационному полю.

Метод Жаккара, основанный на множествах токенов, предоставляет простой и интуитивно понятный способ измерения сходства. Алгоритм шинглов дополняет этот подход, позволяя учитывать порядок слов и обнаруживать схожие участки в тексте.

Расстояние Левенштейна эффективно измеряет разнообразие между текстами, учитывая вставки, удаления и замены символов. TF-IDF и BM25, стоящие в основе информационного поиска, предоставляют метрики, основанные на важности терминов в документах и их частоте.

Современные методы, такие как BERT, отражают переход к контекстно-зависимым представлениям слов, что позволяет учитывать смысловые оттенки в текстах. Использование нейросетей для анализа семантической близости открывает новые возможности, но требует больших вычислительных ресурсов. Для поиска точных совпадений более подходят классические методы, в то время как для учета контекста и семантики более предпочтительны современные подходы, такие как BERT и нейросети.

В рамках исследования предлагается учитывать объективность и универсальность использования методик. В этой связи наиболее подходящими для различных задач признаются современные методы, поскольку они обладают возможностями самообучения через усвоение новейших информационных тенденций, а также их можно назвать относительно

доступными для объективной интерпретации результатов. Оптимальными методами следует определить алгоритмы искусственного интеллекта и BERT.

**Библиографические ссылки**

1. Установление сходства текстовых документов / А. А. Хорошилов, А. В. Кан, Е. А. Евдокимова, С. Г. Пицхелаури // Моделирование и анализ данных. 2023. Т. 13, № 4. С. 45–58. URL: https://doi.org/10.17759/mda.2023130403.

2. Алгоритм поиска схожих публикаций средств массовой информации / А. Ю. Бородащенко, А. В. Потемкин, Е. А. Сазонова, С. В. Шекшуев // Науковедение. 2015. Т. 7, № 4. URL: http://naukovedenie.ru.

3. *Рафаева А. В. Компьютер* – Слово – Фольклор. М., 2014. 280 с.

4. Семантический анализ научных текстов: опыт создания корпуса и построения языковых моделей / Т. В. Батура, Е. П. Бручес, А. Е. Паульс, В. В. Исаченко, Д. Р. Щербатов // Программные продукты и системы. 2021. № 1. С. 132–144.

5. *Лыченко Н. М., Сороковая А. В*. Сравнение эффективности методов векторного представления слов для определения тональности текстов // Математические структуры и моделирование. 2019. № 4 (52). С. 97–110.

6. *Сорокин Д. И., Нужный А. С., Савельева Е. А.* Иерархическая рубрикация текстовых документов // Труды Института системного программирования РАН. 2020. Т. 23, вып. 6. С. 127–138.

7. *Краснов Ф. В., Смазневич И. С.* Фактор объяснимости алгоритма в задачах поиска схожести текстовых документов // Вычислительные технологии. 2020. Т. 25, № 5. С. 107–123.

8. *Мясоедова В. А., Голубничий А. А.* Обзор пакета STRINGDIST языка программирования R для алгоритма «расстояние Левенштейна». URL: https://cyberleninka.ru/article/n/obzor-paketa-stringdist-yazyka-programmirovaniya-r-dlya-algoritma-rasstoyanie-levenshteyna.

9. *Ананьев А. В.. Кузнецов И. А., Доброскок В. В.* Комбинированная методика определения качества машинного перевода // Успехи в химии и химической технологии. 2021. Т. 35, № 11. С. 37–39.

10. *Кравченко В.* Алгоритм OkapiBM25 – модификация формулы TF-IDF ранжирования документов. URL: https://weblinepromo.ru/blog/algoritm-okapi-bm25-modifikaciya-formuly-tf-idf-ranzhirovaniya-dokumentov.

11. *Рафаева А. В.* Автоматизированный поиск цвета в русских сказках // Вестник Воронежского государственного университета. Серия: Лингвистика и межкультурная коммуникация. 2015. № 3. С. 45–54.

12. Bashir Alam An Easy Introduction To Artificial Neural Networks // hands-on.cloud, 14.02.2023. URL: https://hands-on.cloud/introduction-to-artificial-neural-networks.

13. *Городецкий В. И., Тушканова О. Н.* Семантические технологии для семантических приложений. Ч. 2. Модели сравнительной семантики текстов // Искусственный интеллект и принятие решений. 2019. № 1. С. 49–61.

14. *Нужный А. С., Сорокин Д. И.* Создание программы интеллектуального анализа текстовой документации по вопросам захоронения РАО // Труды МФТИ. 2020. Т. 12, № 1 (45). С. 104–111.

15. Математическая составляющая / ред.-сост. Н. Н.Андреев, С. П. Коновалов, Н. М. Панюнин. М. : Математические этюды, 2019. 367 с.

16. *Чару Аггарвал.* Нейронные сети и глубокое обучение : учебный курс. М. : Диалектика-Вильямс, 2020, 752 с.

17. *Частикова В. А., Гуляй В. Г., Жерлицын С. А.* Подход к решению проблемы контроля качества в сфере услуг на основе построения системы интеллектуального анализа данных // Вестник «АГУ». 2022. Вып. 4 (311). С. 81–90.

18. *Салып Б. Ю., Смирнов А. А., Ничушкина Т. Н.* Анализ модели BERT как инструмента определения меры смысловой близости предложений естественного языка // StudNet : научно-образовательный журнал для студентов и преподавателей. 2022. № 5. С. 3509–3518.

**References**

1. Khoroshilov A.A., Kan A.V., Evdokimova E.A., Pitskhelauri S.G. [Establishing the similarity of text documents]. Modeling and data analysis. 2023. Vol. 13, no. 4. Pp. 45-58. Available at: https://doi.org/10.17759/mda.2023130403 (in Russ.).

2. Borodashchenko A.Yu., Potemkin A.V., Sazonova E.A., Shekshuev S.V. [Algorithm for searching similar media publications]. Science, Internet journal. Vol. 7, no. 4. Available at: http://naukovedenie.ru (in Russ.).

3. Rafaeva A.V. *Komp'yuter – Slovo – Fol'klor* [Computer – Word – Folklore]. Moscow, 2014. 280 p. (in Russ.).

4. Batura T.V., Bruches E.P., Pauls A.E., Isachenko V.V., Shcherbatov D.R. [Semantic analysis of scientific texts: experience in creating a corpus and constructing language models]. *Programmnye produkty i sistemy*. 2021. No. 1. Pp. 132-144 (in Russ.).

5. Lychenko N.M., Sorokovaya A.V. [Comparison of the effectiveness of methods for vector representation of words for determining the sentiment of texts]. *Matematicheskie struktury i modelirovanie*. 2019. No. 4. Pp. 97-110 (in Russ.).

6. Sorokin D.I., Nuzhny A.S., Savelyeva E.A. [Hierarchical categorization of text documents]. *Trudy Instituta sistemnogo programmirovaniya RAN.* 2020. Vol. 23, no. 6. Pp. 127-138 (in Russ.).

7. Krasnov F.V., Smaznevich I.S. [The factor of explainability of the algorithm in problems of searching for the similarity of text documents]. *Vychisli-*

*tel'nye tekhnologii*. 2020. Vol. 25, no. 5. Pp. 107-123 (in Russ.).

8. Myasoedova V.A., Golubnichiy A.A. *Obzor paketa STRINGDIST yazyka programmirovaniya R dlya algoritma «rasstoyanie Levenshteina».* [Review of the STRINGDIST package of the R programming language for the Levenshtein distance algorithm]. Available at: https://cyberleninka.ru/article/n/obzor-paketa-stringdist-yazyka-programmirovaniya-r-dlya-algoritma-rasstoyanie-levenshteyna (in Russ.).

9. Ananyev A.V., Kuznetsov I.A., Dobroskok V.V. [A combined method for determining the quality of machine translation]. *Uspekhi v khimii i khimicheskoi tekhnologii.* 2021. Vol. 35, no. 11, pp. 37-39 (in Russ.).

10. Kravchenko V. *Algoritm OkapiBM25 – modifikatsiya formuly TF-IDF ranzhirovaniya dokumentov* [Okapi BM25 algorithm - modification of the TF-IDF formula for ranking documents]. Available at: https://weblinepromo.ru/blog/algoritm-okapi-bm25-modifikaciya-formuly-tf-idf-ranzhirovaniya-dokumentov (in Russ.).

11. Rafaeva A.V. [Automated search for color in Russian fairy tales]. *Vestnik Voronezhskogo gosudarstvennogo universiteta. Seriya: Lingvistika i mezhkul'turnaya kommunikatsiya*. 2015. No. 3, pp. 45-54 (in Russ.).

12. Bashir Alam An Easy Introduction To Artificial Neural Networks // hands-on.cloud, 02/14/2023 - https://hands-on.cloud/introduction-to-artificial-neural-networks.

13. Gorodetsky V.I., Tushkanova O.N. [Semantic technologies for semantic applications. Part 2. Models of comparative semantics of texts]. *Iskusstvennyi intellekt i prinyatie reshenii*. 2019. No. 1. Pp. 49-61 (in Russ.).

14. Nuzhny A.S., Sorokin D.I. [Creation of a program for intellectual analysis of text documentation on the issues of radioactive waste disposal]. *Trudy MFTI.* 2020. Vol. 12, no. 1, pp. 104-111 (in Russ.).

15. *Matematicheskaya sostavlyayushchaya* [Mathematical component]. Editors and compilers N. N. Andreev, S. P. Konovalov, N. M. Panyunin. Moscow: *Matematicheskie etyudy*, 2019. 367 p. (in Russ.).

16. Charu Aggarwal. *Neironnye seti i glubokoe obuchenie* [Neural networks and deep learning]: training course. Moscow: Dialektika-Vil'yams Publ., 2020, 752 p. (in Russ.).

17. Chastikova V.A., Gulyai V.G., Zherlitsyn S.A. [An approach to solving the problem of quality control in the service sector based on the construction of a data mining system]. *Vestnik «AGU»*. 2022. Vol. 4, pp. 81-90 (in Russ.).

18. Salyp B.Yu., Smirnov A.A., Nichushkina T.N. [Analysis of the BERT model as a tool for determining the measure of semantic similarity of natural language sentences]. *StudNet : n auchno-obrazovatel'nyi zhurnal dlya studentov i prepodavatelei.* 2022. No. 5. Pp. 3509-3518 (in Russ.).

* * *

**Methods and Algorithms for Searching Similarities between Texts**

*I. M. Yannikov*, DSc in Engineering, Associate Professor, Kalashnikov Izhevsk State Technical University, Izhevsk, Russia

*M. V. Ershova,* PhD in Engineering, Associate Professor, Kalashnikov Izhevsk State Technical University, Izhevsk, Russia

*A. N. Isenbaev*, Postgraduate, Kalashnikov Izhevsk State Technical University, Izhevsk, Russia

*The review of the analytical article is a comprehensive study of text analysis modern methods in order to identify and measure the degree of their similarity, which itself is a very important and relevant task, since it examines and analyzes the tools used to solve it. The introduction discusses the purpose of this work, the relevance of the problem, and the importance of developing effective methods for comparing texts.*

*The main part of the article examines and analyzes such methods as "jaccard similarity", "shingle algorithm", "levenshtein distance", "tf-idf" and "bm25", "bert" and the use of neural networksseparately. The application of a particular method is illustrated by examples presented in tabular form and illustrations. When considering and analyzing the "jaccard similarity", the methods of its application and limitations are considered. When analyzing the "shingles algorithm", the advantages of the method in the context of similarity search are revealed. The publication discusses methods based on line spacingin detail, including levenshtein distance. In this case, special attention is paid to the scope of its application and its advantages over other methods. By reviewing statistical methods such as "tf-idf" and "bm25", theanalysis of their application and effectiveness in text similaritysearching is given. The article is not limited by analyzing only traditional methods, but it also covers modern ones, including "bert" and the use of neural networks. These methods are compared with each other, their advantages and disadvantages of use are identified.*

*The conclusion section provides a comparative analysis of all presented methods based on the principle of objectivity, highlighting their characteristics and areas of application. The importance of choosing the most appropriate method for text similaritysearching is noted, depending on the specific search goals, tasks and requirements, and a conclusion is given about the most used, vast and productive method i.e. The use of neural*

*networks. The conclusions emphasize that the article, devoted to a comparative analysis of various methods for similarity searching between texts, has the main goal of developing recommendations to choose the optimal method.*

**Keywords:** similarity search algorithms, text processing, text comparison, uniqueness comparison, neural networks, artificial intelligence, shingle algorithm.

Получено: 22.01.24

**Образец цитирования**

*Янников И. М., Ершова М. В., Исенбаев А. Н.* Методы и алгоритмы для поиска сходства между текстам // Интеллектуальные системы в производстве. 2024. Т. 22, № 2. С. 103–117. DOI: 10.22213/2410-9304-2024-2-103-113.

**For Citation**

Yannikov I.M., Ershova M.V., Isenbaev A.N. [Methods and Algorithms for Searching Similarities between Texts]. *Intellektual'nye sistemy v proizvodstve.* 2024, vol. 22, no. 2, pp. 103-117. DOI: 10.22213/2410-9304-2024-2-103-113.