Московский физико-технический институт
(национальный исследовательский университет)

На правах рукописи
УДК 004.021

Сафин Камиль Фанисович

Комбинированные методы выявления заимствований в текстовых документах

Специальность 05.13.17 —
«Теоретические основы информатики»

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель:
к.ф.-м.н.
Чехович Юрий Викторович

Москва — 2022

Оглавление

Стр.

Стр.

Стр.

Введение

Поиск заимствований в текстовых документах является сложной, но в то же время востребованной задачей, особенно в академической и студенческой средах [1—3].

Можно выделить два глобальных подхода к задаче поиска заимствований в тексте: поиск внешних заимствований (external plagiarism detection) и поиск внутренних заимствований (intrinsic plagiarism detection). Поиск внешних заимствований представляет собой поиск по внешней коллекции документов, которые могли быть использованы в качестве источника заимствования. Такой подход в том или ином виде сводится к попарному сравнению исследуемого документа с каждым документом из коллекции.

Коллекция текстовых документов, по которой происходит поиск внешних заимствований, как правило, довольно большая, а значит и поиск по ней является тяжелой вычислительной задачей. Как правило, тексты представляют в виде перекрывающихся словесных n-грамм (т.н. шинглов), которые впоследствии сравнивают с n-граммами анализируемого документа [4]. Промышленные инструменты, работающие на таком принципе сравнения документов показывают высокую точность при поиске заимствований в текстовых документах [5]. Такой метод работает только в случае дословного заимствования фрагмента текста. Однако существуют методы обфускации (маскирования) заимствованных фрагментов, например, перефразирование или перевод текстового фрагмента из документа на другом языке. Конечно, системы поиска заимствований умеют находить и перефразирования [6], и переводные заимствования [7], однако это требует дополнительных расходов. Во-первых, требуется больше времени и вычислительных ресурсов на проверку одного документа, а во-вторых, необходимо постоянно расширять текстовую коллекцию потенциальных источников.

Поиск внутренних заимствований же, наоборот, не использует внешнюю коллекцию потенциальных источников, а анализирует текст изолированно [8]. При поиске анализируются различные стилистические, синтаксические, орфографические особенности текста.

Поиск внутренних заимствований обычно рассматривается как полноценный инструмент обнаружения текстовых заимствований. То есть, в результате работы алгоритма должны быть указаны конкретные фрагменты текста, кото-

рые были заимствованы [9]. Анализируемый текст при таком подходе, как правило, разбивается на отдельные сегменты. Например, текст делится на предложения [10], или определяется некоторая ширина шага, в соответствии с которой текст разделяется на сегменты одинаковой длины [11]. Полученные сегменты сравниваются со всем текстом и делается вывод о заимствовании для каждого сегмента. Для сравнения сегментов используются различные признаки, например, частота символьных n-грамм, из которых состоит текст [12; 13], или грамматические [14] и синтаксические признаки [15]. Иногда используются векторные представления, полученные с помощью нейронных сетей [16]. Довольно часто решается более общая задача диаризации авторов, в рамках которой нужно определить авторство для каждого фрагмента текста [17; 18]. Методы поиска внутренних заимствований, в силу ограничения на анализ только исследуемого текста, не отличаются высокими показателями точности [19].

Сравнивая эти два подхода, можно сделать вывод, что методы поиска заимствований по внешней коллекции являются точными, но ресурсоемкими, а методы поиска внутренних заимствований — гораздо менее точными, но не сильно требовательными к ресурсам. При этом, в периоды пиковой нагрузки (например, во время сессии у студентов), система поиска по внешней коллекции может перестать справляться со входящим потоком документов для проверки, что приведет либо к сильной задержке ответа либо к отказу от проверки. Оба случая крайне нежелательны со стороны системы проверки. Самый простой способ ускорить работу заключается в уменьшении количества проверок (например, отказ от поиска переводных заимствований) или в сокращении коллекции потенциальных источников заимствований. И то и другое сильно скажется на качестве поиска заимствований в каждом рассматриваемом документе.

В такой ситуации кажется логичным не упрощать работу точной, но ресурсоемкой системы, а каким-то образом сократить поток входящих документов. Так как основной целью работы системы является выявление документов с высоким процентом заимствований, то было бы выгодно сокращать поток за счет высокооригинальных (т.е. с малой долей заимствований) документов. Для этой цели предлагается использовать подход по поиску внутренних заимствований. Как было сказано, в качестве самостоятельного инструмента, такой подход имеет очень низкое качество работы. Но его можно использовать как грубый фильтр перед более точной проверкой, который будет отсеивать документы, которым не нужна детальная экспертиза.

Целью данной работы является разработка методов обнаружения некорректных текстовых заимствований без использования внешней коллекции потенциальных источников заимствований, а также реализация программного комплекса на основе предложенных методов. Задачей данного программного комплекса является повышение эффективности промышленной системы обнаружения текстовых заимствований за счет выбора набора методов, которыми будет осуществляться проверка. Выбор происходит таким образом, что для части документов выбираются методы с низкими требованиями к вычислительным ресурсам, а для части документов, требующих детальной проверки — методы с высокой вычислительной сложностью.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1. Исследовать существующие методы поиска текстовых заимствований без использования потенциальной коллекции источников.
2. Предложить метод обнаружения некорректных заимствований, использующий только информацию об исследуемом тексте, и оценить работоспособность такого метода
3. На основе предложенного алгоритма разработать способ фильтрации документов для последующего использования различных наборов методов при проверке на заимствования.
4. Протестировать и оценить качество алгоритма на реальных данных.

Научная новизна данной работы заключается в разработке набора алгоритмов по обнаружению некорректных текстовых заимствований. Предложен способ обнаружения границ смены авторского стиля письма, основанный на анализе частот употребления словесных и символьных n-грамм. На основе данного способа предложен метод фильтрации высокооригинальных текстов, которые не нуждаются в детальной проверке через систему поиска внешних заимствований.

Практическая значимость данной работы заключается в том, что предлагаемые методы предназначены для предварительного анализа документов на предмет заимствований. Документы, которые по результатам этой проверки имеют очень мало потенциальных некорректно использованных фрагментов, могут быть исключены из очереди на проверку по полноценной системе поиска заимстований, что частично снизит нагрузку на эту систему. Предложенные методы не требуют больших вычислительных мощностей, что позволяет

использовать их для экономии машинного времени и ресурсов в периоды высокой нагрузки на систему поиска заимствований. Также важно упомянуть, что предлагаемые методы предназначены в том числе для работы на русском языке. Это важно ввиду того, что основные методы, предлагаемые в научном сообществе, изначально предназначены для английского языка и не адаптированы для русского.

Методология и методы исследования. Для достижения заявленных целей, используется метод, основанный на анализе частот употребления слов и символьных n-грамм [20]. Используется адаптация метода векторизации с помощью статистик tf-idf [21] применительно к задаче векторизации текстовых сегментов.

Основные положения, выносимые на защиту:

1. Предложен способ векторизации фрагментов текста, основанный на частотах встречаемости символьных и словесных n-грамм в анализируемом тексте и в каждом фрагменте по отдельности.
2. Разработан способ обнаружения заимствованных фрагментов текста, основанный на сегментировании анализируемого текста и анализе ряда статистик, построенных для каждого из полученных сегментов, на предмет наличия выбросов.
3. Разработан метод обнаружения и фильтрации высокооригинальных текстовых документов без внешней коллекции потенциальных источников и с использованием малых вычислительных мощностей.
4. Обоснована работоспособность предложенного алгоритма путем реализации и тестирования на подготовленных данных. Экспериментально показано, что предложенный алгоритм может отфильтровывать до 30% высокооригинальных документов, не сильно проигрывая в качестве полноценной проверке.

Апробация работы. Основные результаты работы докладывались и обсуждались на следующих научных конференциях:

1. «Определение заимствований в тексте без указания источника», Всероссийская конференция «59-ая научная конференция МФТИ с международным участием», 2016.
2. «Style Breach Detection with Neural Sentence Embeddings», Международная конференция «Conference and Labs of the Evaluation Forum», 2017

3. «Detecting a Change of Style using Text Statistics», Международная конференция «Conference and Labs of the Evaluation Forum», 2018
4. «CrossLang: The System of Cross-lingual Plagiarism Detection», Международная конференция «Workshop on Truth Discovery and Fact Checking: Theory and Practice at conference on Knowledge Discovery and Data mining», 2019
5. «CrossLang: The System of Cross-lingual Plagiarism Detection», Международная конференция «Workshop on Deep Learning for Education at conference on Knowledge Discovery and Data mining», 2019
6. «Определение факта заимствования в текстовых документах без указания источника», Всероссийская конференция «Математические методы распознавания образов (ММРО)», 2021.

Личный вклад. Все приведенные результаты, получены диссертантом лично при научном руководстве к.ф.-м.н. Ю. В. Чеховича.

Публикации. Основные результаты по теме диссертации изложены в 5 печатных изданиях, 2 из которых изданы в журналах, рекомендованных ВАК, 4 — в периодических научных журналах, индексируемых Web of Science и Scopus.

1. *К. Ф. Сафин.* Определение заимствований в тексте без указания источника / К. Ф. Сафин, М. П. Кузнецов, М. В. Кузнецова // Информ. и её примен. 2017. т. 11, № 3
2. *Safin, K.* Style Breach Detection with Neural Sentence Embeddings / K. Safin, R. Kuznetsova // Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Dublin, Ireland, September 11-14, 2017. Vol. 1866 / ed. by L. Cappellato [et al.]. CEUR-WS.org, 2017. (CEUR Workshop Proceedings)
3. *Safin, K.* Detecting a Change of Style using Text Statistics: Notebook for PAN at CLEF 2018 / K. Safin, A. Ogaltsov // Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, Avignon, France, September 10-14, 2018. Vol. 2125 / ed. by L. Cappellato [et al.]. CEUR-WS.org, 2018. (CEUR Workshop Proceedings)
4. Near-duplicate handwritten document detection without text recognition / O. Bakhteev [et al.] // Computational Linguistics and Intellectual Technologies. 2021
5. *К. Ф. Сафин.* О комбинированном алгоритме обнаружения заимствований в текстовых документах / К. Ф. Сафин, Ю. В. Чехович // Тру-

ды Института системного программирования РАН. 2022. т. 34, № 1. с. 151—160

Объем и структура работы. Диссертация состоит из введения, 5 глав, и заключения. Полный объём диссертации составляет 83 страницы, включая 15 рисунков и 5 таблиц. Список литературы содержит 123 наименования.

Краткое содержание работы по главам. В первой главе приводятся основные разновидности постановок задачи поиска внутренних текстовых заимствований. Также приводится состояние проблемы на текущий момент времени и обзор существующих методов решения.

Во второй главе приводится общий подход к решению поставленной задачи, предлагаемый в данной работе. Обосновывается выбор способа построения векторных представлений сегментов текста, а также экспериментально показывается состоятельность данного метода.

В третьей главе описывается метод поиска некорректно заимствованных сегментов текста. Предлагаемый метод экспериментально сравнивается с похожими методами поиска текстовых заимствований.

В четвертой главе приводится модификация метода поиска заимствованных сегментов с использованием вспомогательных моделей векторизации текста. Описывается вычислительный эксперимент, в рамках которого данный метод сравнивается со статистическими подходами к решению данной задачи.

В пятой главе описывается метод для решения упрощенной задачи установления факта наличия заимствований в анализируемом тексте. Предлагается метод основанный на предыдущих алгоритмах, предназначенный для решения задачи бинарной классификации. Показывается, что предлагаемый метод позволяет с высокой точностью отбирать тексты без наличия некорректных заимствований. Описываются детали реализации программного комплекса.

Глава 1. Обзор литературы

1.1 Интерпретации задачи

Проблема некорректных текстовых заимствований существует достаточно долго [1]. Авторство некоторых классических художественных произведений до сих пор подвергается сомнению. Однако чаще всего данная проблема возникает в сфере образования и науки. Письменные работы в формате рефератов, курсовых работ, диссертаций и прочего давно стали своего рода классикой для оценки знаний обучающихся. А развитие научной области невозможно без написания статей и учебников, так как это, наверное, единственный на сегодняшний день способ зафиксировать некоторый результат научной деятельности.

При этом, при выполнении письменной работы или написании научной статьи, некоторые авторы не всегда приводят цитату, а неправомерно используют фрагменты текстов из трудов чужого авторства. Широкое распространение сети Интернет, к сожалению, сделало такую возможность крайне доступной. Но считать такую сборную работу правомерной все-таки нельзя. Если работу выполняет обучающийся, то работа должна показать степень усвоения материала. Если же работа является научным результатом, то такое действие можно расценивать как присвоение чужих результатов и нарушение авторских прав. И в том и в другом случае неправомерные текстовые заимствования недопустимы.

Необходимо иметь инструмент по обнаружению некорректных текстовых заимствований. Учитывая объемы работ, о ручной проверке речи быть не может: на сегодняшний день физически невозможно сравнить одну рукопись со всеми источниками, выложенными в открытом доступе. Существуют системы для проверки работ на предмет заимствований [5]. Как правило, такие системы сравнивают анализируемый текст с открытыми источниками и своими закрытыми базами документов.

Существует два глобальных подхода к задаче поиска заимствований в тексте: поиск внешних заимствований (от англ. external plagiarism detection) и поиск внутренних заимствований (от англ. intrinsic plagiarism detection). Поиск внешних заимствований представляет собой поиск по внешней коллекции документов, которые могли быть использованы в качестве источника заимство-

вания. Такой подход в том или ином виде сводится к попарному сравнению исследуемого документа с каждым документом из коллекции.

Коллекция текстовых документов, по которой происходит поиск внешних заимствований, как правило, большая, а значит и поиск по ней является тяжелой вычислительной задачей. Как правило, тексты представляют в виде перекрывающихся словесных n-грамм (т.н. шинглов), которые впоследствии сравнивают с n-граммами анализируемого документа [4]. Промышленные инструменты, работающие на таком принципе сравнения документов показывают высокую точность при поиске заимствований в текстовых документах [5]. Такой метод работает только в случае дословного заимствования фрагмента текста. Однако существуют методы обфускации (маскирования) заимствованных фрагментов, например, перефразирование или перевод текстового фрагмента из документа на другом языке. Конечно, системы поиска заимствований умеют находить и перефразирования [6], и переводные заимствования [7], однако это требует дополнительных расходов. Во-первых, требуется больше времени и вычислительных ресурсов на проверку одного документа, а во-вторых, необходимо постоянно расширять текстовую коллекцию потенциальных источников.

Поиск внутренних заимствований же, наоборот, не использует внешнюю коллекцию потенциальных источников, а анализирует текст изолированно [8]. При поиске анализируются различные стилистические, синтаксические, орфографические особенности текста.

Поиск внутренних заимствований обычно рассматривается как полноценный инструмент обнаружения текстовых заимствований. То есть, в результате работы алгоритма должны быть указаны конкретные фрагменты текста, которые были заимствованы [9]. Анализируемый текст при таком подходе, как правило, разбивается на отдельные сегменты. Например, текст делится на предложения [10], или определяется некоторая ширина шага, в соответствии с которой текст разделяется на сегменты одинаковой длины [11]. Полученные сегменты сравниваются со всем текстом и делается вывод о заимствовании для каждого сегмента. Для сравнения сегментов используются различные признаки, например, частота символьных n-грамм, из которых состоит текст [12; 13], или грамматические [14] и синтаксические признаки [15]. Иногда используются векторные представления, полученные с помощью нейронных сетей [16]. Довольно часто решается более общая задача диаризации авторов, в рамках которой нужно определить авторство для каждого фрагмента текста [17; 18]. Методы поиска

внутренних заимствований, в силу ограничения на анализ только исследуемого текста, не отличаются высокими показателями точности [19].

Поиск некорректных заимствований в тексте без привлечения внешней коллекции — довольно общая постановка задачи, поэтому существует множество различных задач, подходящих под данное описание:

- Кластеризация по авторству [26]. Имея текстовый документ, необходимо выделить в нем сегменты и сгруппировать эти сегменты согласно авторству.
- Обнаружение факта, что документ написан несколькими авторами [27]. Нужно сделать вывод, является ли исследуемый текст оригинальной работой одного автора или же нескольких.
- Нахождение нарушений стиля [28]. В анализируемом тексте необходимо найти позиции, на которых происходит изменение авторского стиля.
- Определения числа авторов [29]. Задача очень похожа на обнаружение факта, что документ написан несколькими авторами, но дополнительно необходимо установить, скольким авторам принадлежит исследуемый текст.
- Проверка авторства [30]. Имея два текста (или их фрагменты), нужно установить, принадлежат ли они одному автору или разным.

Более сложные постановки задач ведут к худшим результатам алгоритмов. Под более сложными постановками подразумевается то, насколько больше информации необходимо предоставить в процессе решения задачи. Например, предсказать факт наличия заимствований в тексте немного проще, чем найти конкретные фрагменты некорректных заимствований.

1.2 Функция стиля и статистический подход

Статистический подход к решению такого рода задач во многом опирается на классические методы обработки естественного языка [31]. Очень часто, в рамках статистического подхода, вводят понятие стилистической функции φ (функции стиля, стилометрии) [32]. Данная функция является попыткой формализации стиля письма отдельно взятого человека. Такая функция (если она существует) должна принимать примерно одинаковые значения на фрагментах

текста одного и того же автора и сильно отличающиеся значения на текстах другого автора. Если говорить более формально, функция должна задавать отображение из множества текстов в некоторое пространство статистик этих текстов $\mathbb{S}$. Для простоты можно предположить, что пространство статистик является множеством рациональных чисел, хотя это совсем не обязательно. И тогда значения такой функции должны быть инварианты относительно текстов одного автора и отличаться от значений на текстах другого автора:

$$\begin{gathered} \varphi(x) : D \to \mathbb{S} \\ \varphi(D_A) \cap \varphi(D_B) = \varnothing, \end{gathered} \tag{1.1}$$

где под D_A, D_B понимаются множества текстов двух независимых авторов. Подразумевается, что авторы независимы и их тексты не пересекаются. Конечно, в реальной жизни это не всегда так. Однако и понятие стилистической функции (1.1) само по себе является довольно неформальным.

Поиском такой функции ученые занимаются уже более века. Было замечено [33], что кривая зависимости частот использования слов от их длины является уникальной для отдельно взятого автора. Данное наблюдение заложило фундамент для целого направления авторской идентификации (authorship attribution) [20]. Логично, что следующие исследования были посвящены поиску схожей функции. Например, предлагалось подсчитывать среднюю длину слов в тексте [34] или, что очень похоже, использовать среднее число слов в предложении в качестве уникальной характеристики авторского стиля [35]. Несложно догадаться, что такие простые статистики не являются индивидуальными показателями для разных авторов, что было доказано и на практике [36].

Логичным развитием описанного выше подхода является анализ самих употребляемых слов, а не только посчет их количества. В области обработки естественного языка принято выделять отдельно категорию стоп-слов (в англоязычной литературе иногда можно встретить также название «функциональные слова», functional words [37]). Стоп-слова — это слова, не несущие практически никакой смысловой нагрузки в тексте. Они используются в качестве элемента связности. К таким словам, например, относят союзы, предлоги, междометия. Стоит отметить, что понятие стоп-слов не сильно формализовано и набор стоп-слов может отличаться в зависимости от контекста. Также, стоп-слова сильно привязаны к области использования. То есть одно и то же слово может считаться стоп-словом в рамках одной тематики и не являться таковым

в рамках другой. Частота употребления стоп-слов по отношению к остальным словам сильно зависит от стиля письма автора и может сохраняться даже при смене тематики [38].

Было показано [39], что такая характеристика, как использование стоп-слов, может быть эффективной в задаче различения авторов, так как каждый автор имеет специфичный шаблон использования этих слов. Данное утверждение было подтверждено различными авторами в своих статьях [36; 40—42]. Важным этапом развития данного подхода было применение метода главных компонент (англ. principal component analysis, PCA [43]) к набору частот использования слов [44]. Основная особенность метода главных компонент заключается в том, что он позволяет снизить размерность исходного векторного пространства при этом теряя минимальное количество информации. Таким образом среди всех употребляемых автором слов можно выделить те, частоты которых являются отличительной чертой данного автора [45].

Также стоит упомянуть методы, не использующие частоты распределения конкретных слов. К ним можно отнести, например, методы, анализирующие общую структуру текста и легкость восприятия этого текста. Сложность текста довольно трудно формализуемая величина, однако среди попыток выразить ее в виде формулы можно выделить индекс удобочитаемости Флеша [46]. Данный индекс отображает легкость восприятия текста человеком исходя из таких показателей как: длины предложений, слов, удельного количества наиболее частотных (или редких) слов и так далее. Иногда данный показатель используется для поиска некорректных заимствований. Гипотеза состоит в том, что большая вариативность индекса внутри текста может говорить о том, что текст на самом деле состоит из фрагментов, написанных разными авторами [47; 48]. Однако данный подход не получил широкого распространения ввиду невысокого качества работы и сильной зависимости индекса от конкретного языка.

1.3 Решение рассматриваемой задачи с применением методов машинного обучения

С развитием компьютерных алгоритмов и методов машинного обучения, частотные характеристики перестали быть самостоятельным способом опреде-

ления авторства или поиска текстовых заимствований. Они стали использоваться в качестве признаков, которые подаются на вход некоторому алгоритму. Если говорить формально, то при использовании некоторого семейства параметрических алгоритмов, в качестве стилистической функции уже выступает композиция функций. То есть статистические признаки текста, такие как, например, частота употребления слова, теперь используются не в качестве индикатора оригинальности текста, а в качестве способа построения векторного представления этого текста.

К примеру, в качестве модели машинного обучения был использован многослойный перцептрон в задаче анализа авторства [39]. В качестве архитектуры была выбрана трехслойная полносвязная сеть. Или один из популярных методов классического машинного обучения — метод опорных векторов (англ. support vector machines, SVM [49]) был применен в задаче распознавания стилистических особенностей разных авторов [50]. Использование высокопарметрических алгоритмов, как следствие, повысило качество решения рассматриваемых задач.

При этом, частотные характеристики текста из самостоятельного инструмента анализа авторства становятся признаками, используемыми методами машинного обучения.

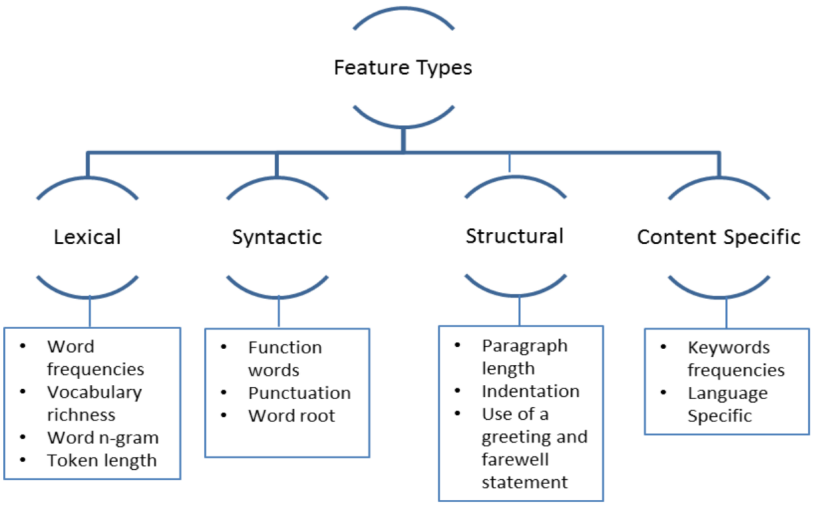


Рисунок 1.1 — Категоризация текстовых признаков из [51]

Существует условное деление текстовых признаков на категории [42; 52]. Наиболее релевантная категоризация признаков приведена на рис. 1.1 [51].

Выделяют четыре крупных группы текстовых признаков:

– Лексические

    - Частоты использования слов
    - Количество уникальных слов
    - Словесные или символьные n-граммы
    - Длины слов
- Синтаксические
    - Использование стоп-слов
    - Пунктуация
    - Части речи
- Структурные
    - Длины предложений и параграфов
    - Отступы
- Зависящие от содержания текста
    - Частоты использования ключевых слов или специфичных символов
    - Признаки, специфичные для конкретного языка

Перечисленные примеры признаков в той или иной степени могут определять стиль письма. Лексические признаки построены на основе базовых единиц текста. Обычно, базовой единицей текста, несущей смысл, принято считать слова, и в перечисленных раннее примерах работ авторы используют частоты использования слов в качестве характеристики стиля письма отдельно взятого автора [53]. Однако в области обработки естественного языка такой единицей чаще считают так называемые n-граммы [54]. Под символьной (или словесной) n-граммой понимается фиксированная последовательность символов (или слов) длины n. Словесные n-граммы также иногда называют шинглами. Частоты использования таких n-грамм довольно успешно используют в качестве признакового описания текстов [55—59]. К примеру, одним из самых первых и самых распространенных подходов является использование символьных или словесных n-грамм для определения авторства [12]. Для каждой рассматриваемой n-граммы подсчитывается частота ее встречаемости в рассматриваемом тексте [13]. Эмпирическое наблюдение заключается в том, что распределение используемых n-грамм индивидуально для каждого человека. Это можно объяснить например тем, человек склонен употреблять слова из своего активного словарного запаса, который в какой-то степени индивидуален. Недостатки такого метода очевидны. Во-первых, для того, чтобы получить наиболее точную аппроксимацию распределения частот для конкретного автора, нужно иметь коллекцию

документов этого автора, причем довольно большую [60; 61]. Во-вторых, хотя лексикон каждого человека и индивидуален, набор слов в конкретном языке, хоть и очень большой, но все-таки ограничен.

К лексическим признакам относятся также различные статистики, посчитанные для используемых автором слов: например, длины слов [32; 42] или количество уникальных слов в тексте. Последнюю величину часто называют размером словаря, где под словарем понимается именно набор уникальных слов, используемых автором.

Синтаксические признаки учитывают особенности письма уровнем выше. К таким признакам можно отнести использование стоп-слов (например, в относительном количестве) [58; 62], разнообразие и количество знаков препинания и частей речи [20; 63]. Стоит отметить, что не все признаки легко можно подсчитать. Например, чтобы получить информацию об употреблении различных частей речи, необходимо сначала произвести синтаксический разбор предложения, что является довольно нетривиальной процедурой.

Структурные и текстозависимые признаки являются самыми высокоуровневыми признаками. К структурным признакам можно отнести способ общей структуризации текста, а именно разбиение на абзацы или [42], например, длины предложений [32; 42]. Тоже стоит отметить, что структуризация текста может теряться в процессе анализа. Так, при анализе pdf-документов, происходит выделение текстового слоя, в процессе чего частично может теряться информация о переносах и отступах. Однако иногда структурные признаки могут быть единственными доступными. К примеру, в работе [24] решается задача поиска некорректных заимствований среди рукописных сочинений. Так как распознавание рукописного текста – очень сложная и не всегда решаемая задача, авторы предлагают использовать временной ряд, составленный из длин слов в качестве представления рукописного текста.

Признаки, зависящие от текста, как правило сильно зависят от решаемой задачи. К ним может относится использование специфичных ключевых слов или символов [42]. Также могут использоваться особенности языка. Например, для русского языка можно подсчитывать частоту использования причастных и деепричастных оборотов, которые не так сильно распространены, например, в английском языке.

1.4 Использование вспомогательных моделей векторизации текстов

С развитием мощностей вычислительных машин, стало возможным использование моделей машинного обучения с большим числом параметров. Речь идет, в первую очередь, о различных архитектурах нейронных сетей.

Самым простым подходом является использование заранее обученных нейронных сетей, используемых для векторизации слов или фрагментов текста. Такие сети осуществляют отображение из пространства слов (или текстов) в некоторое векторное пространство фиксированной размерности. Причем получаемые вектора удовлетворяют гипотезе дистрибутивности [64]. Она заключается в том, что «лингвистические единицы, встречающиеся в схожих контекстах, имеют близкие значения» [65]. Это значит, что слова, встречающиеся в одном контексте будут иметь близкие вектора, согласно некоторой функции схожести. И наоборот, слова, которые редко или никогда не встречаются вместе, будут иметь далекие друг от друга векторы. Как правило, в качестве меры сравнения выбирается косинусная мера, и тогда близость и удаленность векторов равносильна их коллинеарности и ортогональности соответственно.

Таким образом, такие сети позволяют легко получить векторное представление рассматриваемого текста, которое можно использовать как вход для используемой модели. Среди самых популярных подходов можно выделить, например, архитектуру word2vec, которая является одной из первых подобных моделей [66; 67]. В основе модели лежит двухслойная нейронная сеть, которая обучается путем реконструирования контекста слов.

По схожему принципу работает модель GloVe [68]. Отличие заключается в том, что для настройки параметров используется матрица совстречаемости слов в используемом корпусе документов. Данная матрица высокой размерности подвергается процедуре факторизации [69]. Векторные представления слов более низкой размерности, чем матрица совстречаемости, получаются путем минимизации ошибки реконструкции методом наименьших квадратов [70].

Развитием данного подхода является модель fastText [71; 72]. Ключевым ее отличием является то, что данная модель оперирует на уровне символьных n-грамм. То есть рассматриваемое слово представляется в виде перекрывающихся последовательностей символов. Данное нововведение позволило решить

сразу две проблемы. Во-первых, контекст однокоренных слов теперь учитывается совместно, что ведет к улучшению качества. Во-вторых, проблема OOV слов (Out Of Vocabulary, слова, которые модель не встречала в обучающем корпусе) решается автоматически, так как неизвестные слова представляются в виде набора уже известных n-грамм.

Данные модели, хоть и настраиваются с учетом контекста каждого слова, все же являются контекстно независимыми. Это значит, что вектор каждого слова зафиксирован, после окончания процедуры обучения модели. То есть контекст, в котором находится слово не влияет на получаемое векторное представление. Более поздние модели пытаются преодолеть этот недостаток, пытаясь учесть контекст слова при использовании.

Одним из первых подходов к построению контекстно зависимой модели векторизации слов является модель ELMO [73]. Модель представляет из себя рекуррентную нейронную сеть, параметры которой настраиваются для решения некоторых вспомогательных задач (например, классификация текстов, выделение именованных сущностей и подобное). В процессе решения таких задач происходит в том числе векторизация текста. Суть заключается в том, что при решении вспомогательных задач, модель учится в том числе сохранять в векторном представлении информацию о входящей информации с учетом контекста. Этот модуль векторизации затем и используется для решения других задач. Такой подход, когда модель учится решать один тип задач, а затем часть этой модели используется для решения других задач популярен и называется передачей модели обучения (от англ. transfer learning) [74; 75].

Самой популярной на данный момент моделью обработки естественного языка, которая используется в том числе для векторизации, является модель BERT [76]. Данная модель также является ярким примером передачи модели обучения. Отличие от предыдущего метода заключается в более сложном архитектурном устройстве данной модели, которое включает в себя, например, использование механизма внимания [77]. Механизм внимания, по сути является способом учета контекста при построении векторного представления текста. В работе [78], к примеру, предобученная модель BERT используется в качестве алгоритма векторизации, а для классификации используется случайный лес [79].

1.5 Архитектуры нейросетевых моделей

Более сложные нейросетевые подходы подразумевают построение архитектуры и настройку ее параметров.

К примеру, в [80] используется полносвязная сеть с одним скрытым слоем для определения числа авторов рассматриваемого документа. При этом классифицируемый документ векторизуется при помощи tf-idf подхода. В работе [81] авторы используют модель LSTM [82] для решения задачи обнаружения изменения авторского стиля. Подход опирается на извлечение текстовых признаков, таких как средняя длина предложения в словах, средняя длина слова, а также используются предварительно обученные вектора слов модели fastText. Для задачи авторы используют двухслойную двунаправленную LSTM модель. В качестве сегментов текста выбраны абзацы текста. Если изменения стиля между абзацами не обнаружено, текущий абзац приписывается автору предыдущего абзаца.

Подход [83] использует сиамские нейронные сети [84] для расчета степени схожести параграфов. Сиамская Сеть имеет слой векторизации, основанный на модели GloVe, двунаправленный слой LSTM, слой измерения расстояния и полносвязный слой с сигмоидной функцией активации для вычисления итоговой метки класса.

Подход, разработанный в [85], основан на использовании модели BERT и стилистических признаков текста, ранее предложенных в [86]. Векторы формируются на уровне предложений текста, а затем эти векторы предложений объединяются на уровне абзацев. Текстовые признаки извлекаются на уровне абзаца. Чтобы определить изменения стиля между двумя абзацами выполняется бинарная классификация с помощью ансамбля моделей [87].

В статье [88] используется сверточная нейронная сеть, работающая на уровне букв.

1.6 Выводы к главе

В данной главе приведён обзор текущего состояния задачи поиска некорректных текстовых заимствований и ее востребованности в научном сообществе.

Приведены три семейства подходов, используемых при решении различных вариаций данной задачи. Первое семейство , использующее различные статистические показатели текста, является самым ранним, и на данный момент такие методы можно считать устаревшими. Это объясняется тем, что на смену пришли методы классического машинного обучения, где различные статистические показатели используются в качестве признаков. Однако использование более сложных моделей над этими показателями текста позволяет извлечь из них гораздо больше информации. Последнее семейство нейросетевых подходов появилось относительно недавно, в силу увеличения вычислительных мощностей. Нейросетевые алгоритмы потенциально имеют больший потенциал, однако они имеют и более высокие требования. Основное заключается в том, что для настройки таких алгоритмов требуются большие корпуса размеченных данных, что в рассматриваемой задаче, к сожалению, большая редкость. Не менее значимое требование заключается в требуемых вычислительных мощностях. Если речь идёт о высоконагруженной системе проверке студенческих работ, данное условие становится критичным.

Стоит отметить то, что задача поиска некорректных заимствований без использования коллекции потенциальных источников существует достаточно давно. Если брать в расчёт вопросы авторства художественной литературы, то можно сказать, что данный вопрос интересует учёных уже минимум пару веков. Однако точных и универсальных методов ее решения на данный момент до сих пор не существует. Все предлагаемые методы носят скорее исследовательский характер, так как не отличаются высоким качеством работы.

Стоит также отметить, что любое решение данной задачи, как и любое решение задачи из области анализа естественных языков, сильно привязано к целевому языку использования предлагаемого метода. Это связано как с различиями в словарях, используемых в разных языках, так и с другими особенностями. Например, грамматические признаки, показывающие хорошую работу в одном языке, могут совсем не работать в другом языке или же вообще не существовать в принципе. Работ, посвящённых обнаружению внутренних заим-

ствований именно на русском языке гораздо меньше, чем на английском. Существующие же подходы для английского или любого другого языка нужно либо адаптировать, либо сильно дорабатывать.

Глава 2. Метод поиска некорректных текстовых заимствований без использования внешних источников

Как было рассмотрено выше, поиск внутренних заимствований может использоваться в качестве самостоятельной системы поиска некорректных текстовых заимствований. Это означает, что коллекции потенциальных источников заимствований нет в принципе и в распоряжении системы есть только исследуемый текст. В большинстве случаев, при анализе текста на заимствования, необходимо явно указать фрагменты текста, которые были некорректно переиспользованы. При такой постановке задаче, в том или ином виде используется понятие стилистической функции. То есть вывод о заимствовании или оригинальности того или иного фрагмента делается на основе анализа данного фрагмента в рамках всего текста.

Как уже было сказано, почти всегда требуется явно указать заимствованные фрагменты. Но текст, как правило, представляет из себя цельную структуру, не разбитую на отдельные фрагменты. Конечно, в тексте есть предложения, абзацы и секции, но они представляют из себя часть авторского стиля, поэтому не всегда уместно использовать их в качестве естественного разбиения текста на отдельные фрагменты.

Таким образом, в процессе решения задачи поиска некорректных заимствований без использования коллекции потенциальных источников, возникает вспомогательная подзадача сегментирования текста. Решение этой подзадачи сильно зависит от постановки исходной задачи. В данном случае, требования, выдвигаемые к сегментам текста немного противоречат друг другу:

- С одной стороны, сегменты должны быть достаточно малы, чтобы отдельно взятый сегмент содержал в себе либо текст оригинального автора, либо некорректно заимствованный текст.
- С другой стороны, сегменты должны быть достаточно велики, чтобы можно было собрать статистически достоверные признаки данного сегмента.

Кроме того, полученные сегменты необходимо некоторым образом векторизовать, чтобы их можно было анализировать, используя методы машинного обучения. При этом от качества полученных векторов будет зависеть общее качество работы всего алгоритма.

2.1 Векторизация текстов

Одним из самых популярных и в то же время эффективных подходов при обработке текстов естественного языка является анализ частот употребления слов в рассматриваемом тексте. Данный метод показывает неплохие результаты в качестве инструмента анализа текста на предмет авторства. Однако с приходом популярности методов машинного обучения этот подход из самостоятельного инструмента превратился во вспомогательный метод построения признакового описания текста.

2.1.1 Метод мешка слов

Одним из самых простых методов векторизации текста с помощью анализа частот слов, входящих в этот текст, является подход под названием «мешок слов» (от англ. bag-of-words) [64]. В данном подходе текстовый документ d воспринимается как набор (мешок) слов w_i, без учета их порядка и грамматики:

$$d = \bigcup_{i=1}^{|d|} w_i,$$

где под $|d|$ понимается количество всех слов в документе d. Для текста d строится так называемый словарь слов v_d — множество уникальных слов данного текста:

$$v_d = \bigcup_{i=1}^{V_d} w_i,$$

где символом V_d обозначено количество уникальных слов в рассматриваемом тексте. Затем для каждого слова w_i из словаря подсчитывается число вхождений данного слова в текст $n_{w_i}^d$:

$$n_{w_i}^d = \sum_{j=1}^{|d|} \mathbb{I}[w_i = w_j], \tag{2.1}$$

где через $\mathbb{I}$ обозначена индикаторная функция.

Из подсчитанных значений (2.1) составляется вектор $\mathbf{x}_d$ — признаковое описание рассматриваемого документа d:

$$\mathbf{x}_d^{bow} = \begin{bmatrix} n_{w_1}^d \\ \vdots \\ n_{w_{V_d}}^d \end{bmatrix}. \tag{2.2}$$

Однако вектор признакового описания текста сам по себе смысла не несет. Если же рассматривается несколько текстов, то для каждого из них необходимо получить векторное представление. При построении вектора описанным выше способом, полученные вектора текстов будут принадлежать разным векторным пространствам, так как у текстов почти наверняка разные словари и, соответственно, разные размерности векторов. Для того чтобы избежать этой проблемы, строится объединенный словарь рассматриваемого корпуса текстов $D = \{d_i\}_{i=1}^N$:

$$v_D = \bigcup_D v_{d_i}.$$

При использовании объединенного словаря, получаемые векторные представления текстов (2.2) будут находится в одном и том же векторном пространстве. Стоит отметить также, что получаемые вектора будут в некоторой степени разрежены в том смысле, что они будут иметь нули на позициях слов, которые есть в объединенном словаре, но не присутствуют в рассматриваемом тексте.

При этом, можно достаточно просто увеличить репрезентативную мощность данного подхода. При описании концепции мешка слов, в рассмотрение шли только отдельные слова. Однако можно добавить в рассмотрение словесные n-граммы. При этом достаточно просто включить их в словарь, весь остальной алгоритм останется прежним. Популярность использования n-грамм (как правило, это 2- и 3-граммы) объясняется тем, что в естественном языке довольно многие слова употребляются совместно с другими. Самым простым примером могут служить фразеологизмы или устойчивые выражения. Включение n-грамм в рассмотрение улучшает векторизацию текста, так как это позволяет учитывать структурные единицы языка, например, устойчивые словосочетания.

2.1.2 Метод с использованием статистики tf-idf

Метод мешка слов неплохо работает для простого представления текста в виде некоторого вектора. Одним из главных требований, неформально выдвигаемых к векторным представлениям объектов, является их репрезентативность [89]. Один из главных недостатков этого метода же состоит в том, что все слова трактуются как равнозначные. Хотя это довольно очевидно, что некоторые слова играют более значимую роль в представлении текста, чем другие. Самым простым примером могут быть ключевые слова, характерные для какой-либо области знаний. По ключевым словам можно, например, различить тексты разных тематик. Если же рассматривать тексты одной тематики, то логично предположить, что разные авторы склонны использовать немного разные термины для описания одного и того же факта. С другой стороны, факт того, что какое-либо слово больше других употребляется в текстах не говорит о том, что оно несет больше информации , чем остальные слова. Это лишь говорит о том, что это довольно популярное слово, к которым можно отнести, например, стоп-слова.

Более развитым подходом, который призван устранить этот недостаток метода мешка слов, является метод построения векторных представлений текстов с использованием статистик, рассчитываемых для каждого слова [90]. Данный подход векторизует каждый отдельно взятый текст с учетом корпуса документов D, в рамках которого этот текст рассматривается [21].

Логика, лежащая в основе данного метода заключается в следующем. Слова, которые часто встречаются в текстах корпуса D, не могут нести много репрезентативной информации о каком-то конкретном тексте d_i, поэтому такие слова должны иметь меньший вклад в итоговый вектор. И наоборот, слово, которое встречается в тексте d_i, но при этом редко встречается (или вообще отсутствует) в остальных текстах корпуса $D \setminus \{d_i\}$, хорошо представляет данный текст. Такой метод оценивания «важности» слов в рамках текста относительно корпуса документов называется tf-idf (от англ. tf — term frequency, idf — inverse document frequency).

Говоря более формальным языком, tf-idf статистика для слова в документе представлена в виде произведения двух независимых друг от друга статистик.

Первая статистика, tf (term frequency — частота слова), отражает то, насколько часто данное слово w_j употребляется в данном тексте d_i. Можно предложить много разных способов подсчета данного показателя. Самый простой, в виде подсчета числа вхождений слова w_j в d_i, как раз и используется в методе мешка слов. Однако, как правило, число употреблений нормируют, чтобы привести все статистики к единому масштабу. Чаще всего производится нормировка на общее число слов в тексте. В таком случае, формула для расчета статистики слова w_j в тексте d_i выглядит следующим образом:

$$\mathrm{tf}(w_j, d_i) = \frac{n_{w_j}^{d_i}}{\sum\limits_j n_{w_j}^{d_i}}. \tag{2.3}$$

Вторая статистика, idf (inverse document frequency — обратная частота документа), является инверсией частоты, с которой рассматриваемое слово встречается в документах коллекции D [91]. Так как это величина, обратная к частоте встречания слова в текстах корпуса, то чем ближе она к нулю, тем чаще встречается слово и наоборот, чем больше эта величина, тем слово специфичнее для текстов, в которых оно присутствует. Для этой статистики также существуют различные методы расчета. Одним из популярных методов является отношение общего числа документов корпуса к количеству документов, которые содержат рассматриваемое слово w_j. Для того, чтобы статистика не принимала очень большие значения для редких слов (для них число в знаменателе будет близко к нулю), от полученного отношения обычно берется логарифм:

$$\mathrm{idf}(w_j, D) = 1 + \log \frac{|D|}{|\{d_i \in D | w_j \in d_i\}|}, \tag{2.4}$$

где под $|D|$ понимается число документов в рассматриваемой коллекции, а под $|\{d_i \in D | w_j \in d_i\}|$ — число документов, содержащих рассматриваемое слово w_j. Добавление единицы к итоговой статистике необходимо для того, чтобы слова, которые встречаются абсолютно во всех документах, не получили нулевую величину.

Итоговая величина tf-idf для слова w_j в документе d_i в рамках корпуса документов D рассчитывается как произведение описанных множителей:

$$\begin{aligned} \text{tf-idf}(w_j, d_i, D) &= \mathrm{tf}(w_j, d_i) \cdot \mathrm{idf}(w_j, D) = \\ &= \frac{n_{w_j}^{d_i}}{\sum\limits_j n_{w_j}^{d_i}} \cdot \left(1 + \log \frac{|D|}{|\{d_i \in D | w_j \in d_i\}|}\right). \end{aligned} \tag{2.5}$$

Векторное представление текста с учетом tf-idf статистик для слов строится аналогично вектору мешка слов (2.2), только частота использования каждого слова заменяется на статистику:

$$\mathbf{x}_{d_i}^{tf-idf} = \begin{bmatrix} \text{tf-idf}(w_1, d_i, D) \\ \vdots \\ \text{tf-idf}(w_{V_d}, d_i, D) \end{bmatrix}. \tag{2.6}$$

Описанные методы довольно активно используются в решении различных задач и реализованы в прикладных пакетах программирования. Одной из самых популярных реализаций является реализация в рамках библиотеки scikit-learn [92].

2.2 Поиск смены авторского стиля

2.2.1 Сегментирование текста

Задача сегментирования текста является вспомогательной задачей в области обработки естественного языка [93]. Под сегментированием текста понимается процесс разбиения текстового документа d на составляющие его сегменты:

$$d = \bigcup_{i=1}^{|d|} s_i.$$

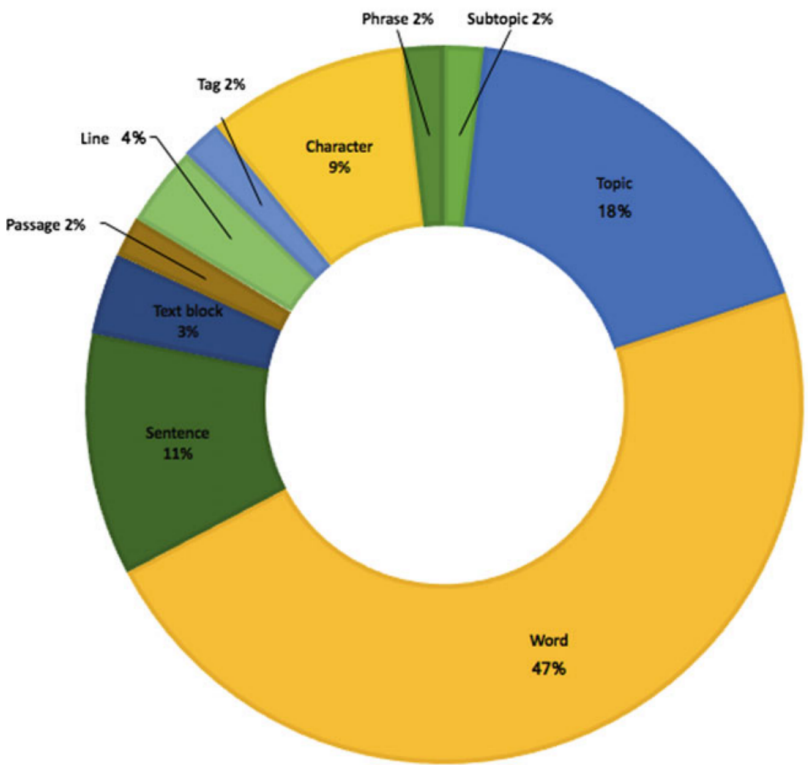
Понятие сегмента сильно зависит от решаемой задачи. Глобально, сегмент — некоторая базовая и самодостаточная единица текстового документа в рамках рассматриваемой задачи. Чаще всего за сегменты в различных задачах выбирают:

– Разделы текста [94; 95]. Как правило это необходимо в задачах извлечения фактов (новостей, мнений и т.п.).
– Предложения [96; 97]. Отличительная особенность предложений в том, что они сохраняют грамматическую целостность. Это важно, например, в задачах машинного перевода [98].

– Слова [99; 100]. Чаще всего используется в задачах, где не важны грамматические отношения между словами. Например, задача классификации текстов часто решается на уровне анализа употребляемых в тексте слов.
– Символы [101; 102]. Например, часто используется в языках, где символы могут представлять целые слова, как правило, это иероглифические языки.

В [93] приводится статистика популярности различных стратегий сегментации текста в работах, связанных с анализом естественного языка, за последние 10 лет. Представленная на рисунке 2.1 диаграмма отображает эту статистику.

Рисунок 2.1 — Частота использования различных стратегий сегментации из [93]



В рамках рассматриваемой задачи важно сохранить стиль письма автора при сегментировании текста. Поэтому разделение на слова и на отдельные символы не подходит. Разделы текста, такие как секции или параграфы, могут быть слишком большими и содержать фрагменты текстов разного авторства. В процессе сегментирования же важно добиться того, чтобы каждый сегмент принадлежал одному автору. Самым подходящим подходом в таком случае является разбиение текста по предложениям [103]. Однако к недостаткам такого метода можно отнести слишком короткие предложения, которые практически

не содержат информацию об авторском стиле. В таком случае можно применять разбиение текста на сегменты одинаковой длины с фиксированным шагом или объединять несколько предложений в один сегмент [17].

2.2.2 Векторизация сегментов

Для использования методов машинного обучения, полученные сегменты текста необходимо представить в виде векторов. При этом, учитывая специфику задачи, крайне желательно, чтобы вектора сегментов строились с учетом их представленности во всем тексте. Похожая логика лежит в основе векторизации текстов с использованием tf-idf статистик (2.6) с тем отличием, что вектор в таком подходе строится для текста в рамках корпуса документов.

Предлагается адаптировать данный подход применимо к подзадаче векторизации сегментов. Изменение будет заключаться в том, что в роли корпуса документов теперь будет выступать набор сегментов текста, а в роли векторизуемого текста – рассматриваемый сегмент. Тогда формула для расчета tf статистики (2.3) для слова w_j в сегменте s_i будет выглядеть следующим образом:

$$\mathrm{tf}_{seg}(w_j, s_i) = \frac{n_{w_j}^{s_i}}{\sum\limits_j n_{w_j}^{s_i}}, \tag{2.7}$$

где за $n_{w_j}^{s_i}$ обозначено число вхождений слова w_j в сегмент s_i. А статистика idf для слова w_j в документе d (2.4) видоизменится следующим образом:

$$\mathrm{idf}_{seg}(w_j, d) = 1 + \log \frac{|d|}{|\{s_i \in d | w_j \in s_i\}|}, \tag{2.8}$$

где через $|d|$ обозначено количество сегментов текста. Аналогично, итоговая статистика tf-idf для слова w_j в документе d получается путем перемножения двух величин (2.5):

$$\begin{aligned} \text{tf-idf}_{seg}(w_j, s_i, d) &= \mathrm{tf}_{seg}(w_j, s_i) \cdot \mathrm{idf}_{seg}(w_j, d) = \\ &= \frac{n_{w_j}^{s_i}}{\sum\limits_j n_{w_j}^{s_i}} \cdot \left(1 + \log \frac{|d|}{|\{s_i \in d | w_j \in s_i\}|}\right). \end{aligned} \tag{2.9}$$

Так же, как и в случае векторизации текстов, построенная статистика может использоваться для векторизации сегментов текста:

$$\mathbf{x}_{s_i}^{tf-idf} = \begin{bmatrix} \text{tf-idf}_{seg}(w_1, s_i, d) \\ \vdots \\ \text{tf-idf}_{seg}(w_{S_d}, s_i, d) \end{bmatrix}. \tag{2.10}$$

При этом также слова будут иметь веса соответственно их важности. Распространенные слова, которые встречаются во всем тексте, получат меньший коэффициент, а более редкие слова, которые встречаются в малом количестве сегментов, получат больший коэффициент.

2.2.3 Построение ряда статистик

Для выявления некорректно заимствованных сегментов текста, необходимо построить для каждого из них некоторую статистику, которая будет отображать степень принадлежности сегмента стилю письма основного автора текста.

Так как сегменты чужого авторства, по предположению, должны стилистически отличаться от всего остального текста, то логично сравнить каждый сегмент со всем текстом. Причем построенные вектора сегментов (2.10) уже несут некоторую информацию о том, насколько данный сегмент стилистически похож на весь текст.

Так как векторы сегментов находятся в пространстве единой размерности, то надо оценить, насколько вектора удалены друг от друга. Один из распространенных подходов в таком случае — рассчитать попарные расстояния между векторами. Чаще всего для этих целей используют косинусную меру близости, так как в высокоразмерных пространствах удобнее использовать ограниченную функцию [104]. Но также используются и другие метрики, например, L1 [17].

Однако в процессе векторизации сегментов (2.10), не происходит векторизации самого текста. Для сравнения вектора рассматриваемого сегмента со всем текстом предлагается использовать среднее расстояние до всех сегментов текста. Либо брать средний вектор сегментов в качестве векторного представления всего текста.

2.2.4 Поиск выбросов

Как уже было сказано, сегменты, отличающиеся от остального текста, потенциально могут быть некорректно заимствованными. С учетом описанных ранее процедур векторизации и расчета статистик для каждого сегмента, это значит, что статистика рассматриваемого сегмента должна сильно отличаться от остальных. То есть возникает задача поиска выбросов в ряде статистик.

Самым простым, но эффективным подходом обнаружения выбросов в ряде статистик является фильтрация по некоторому пороговому значению [105]. Статистики, превышающие некоторое пороговое значение считаются выбросами, а соответствующие сегменты — некорректно заимствованными.

2.3 Базовый эксперимент

Для того, чтобы убедиться в том, что предложенный подход по обнаружению выбросов в ряде статистик действительно позволяет обнаруживать некорректно используемые фрагменты текста чужого авторства, был проведен базовый эксперимент.

Эксперимент был проведен на подвыборке корпуса текстов, подготовленных в рамках конкурса PAN CLEF [106]. PAN — это серия научных мероприятий и совместных задач по цифровой криминалистике и стилометрии текста. Начиная с 2007 года регулярно проводятся конференции и устраиваются конкурсы, посвященные в том числе обнаружению некорректных текстовых заимствований [26—30; 106].

Эксперимент заключается в реализации простейшего метода построения ряда статистик и его визуализации для оценки корреляции между заимствованными сегментами и значения статистики, подсчитанной для этих сегментов.

2.3.1 Подход

В качестве процедуры сегментации рассматриваемого текста был выбран процесс разделения текста по предложениям. В целом, стратегия разделения текста по предложениям является не самой лучшей. Основным недостатком такого подхода являются слишком короткие предложения, на которых подсчет статистики будет сильно отличаться от остальных. Однако такой подход приемлем в рамках предварительной оценки работоспособности алгоритма.

Как уже было сказано, для того, чтобы провести анализ текста, нужно предложить способ построения его векторного представления. Для проведения базового эксперимента была выбрана упрощенная статистика (2.9), которая учитывает только рассматриваемый документ (tf часть произведения). Каждому слову w_j ставится в соответствие число:

$$\text{fr\_class}_{w_j} = \log \frac{n_{\max}}{n_{w_j}}, \tag{2.11}$$

где $n_{\max}$ — число вхождений наиболее часто употребимого слова в тексте, n_w — частота вхождений слова w в этом предложении. Из полученных значений формируется итоговый вектор аналогично (2.10).

Рассчитанные признаки нормировались с использованием среднего значения и среднеквадратичного отклонения.

Обозначим за $m^j = \overline{x^j}$ среднее значение j-го признака для рассматриваемого документа, за r^j — среднеквадратичное отклонение. Тогда нормализованный признак j для сегмента i рассчитывается по формуле

$$s_i^j = \frac{x^j - m^j}{r^j}. \tag{2.12}$$

Для каждого предложения s_i строился вектор признаков $\mathbf{s}_i$ и затем подсчитывалось отклонение от усредненного по всему тексту вектора $\mathbf{s}_{avr}$ в L1-метрике:

$$stat(\mathbf{s}_i) = ||\mathbf{s}_i - \mathbf{s}_{avr}|| = \sum_{j=1}^{l} |s_i^j - s_{avr}^j|. \tag{2.13}$$

2.3.2 Результаты и примеры

Эксперимент проводился на нескольких текстах коллекции PAN-2011.

На рисунке 2.2 показан пример работы алгоритма на одном из текстов коллекции. На графике показано отклонение признакового вектора каждого предложения от усредненного вектора. Красными линиями выделены предложения, помеченные экспертом как заимствованные.

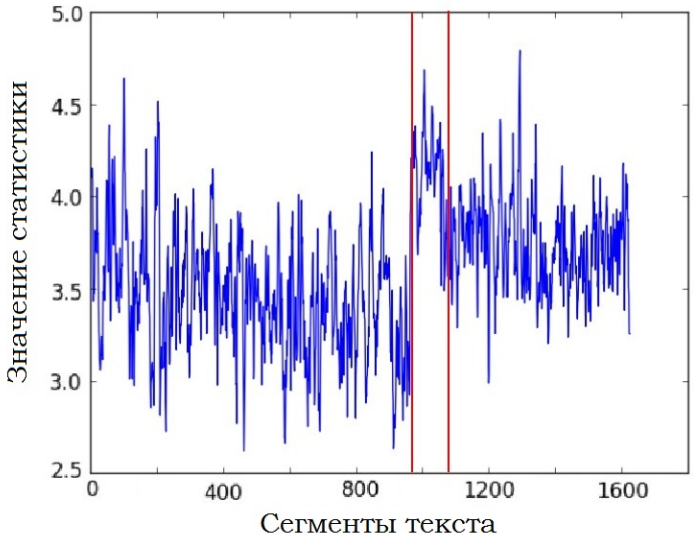


Рисунок 2.2 — Отклонение признакового вектора от среднего

Видно, что заимствованные фрагменты имеют характерные выбросы из области средних значений отклонения. Однако некоторые предложения, не являющиеся заимствованными, также сильно отличаются от усредненного признакового вектора. На основании этого можно сделать вывод, что использование только данного признака недостаточно для решения поставленной задачи. Для более точного детектирования заимствованных сегментов, необходимо проводить дополнительные операции над полученным рядом статистик.

2.4 Выводы к главе

В данной главе приведены основные методы анализа естественных языков, используемые в настоящей работе. Также дано общее описание подхода,

предлагаемого для поиска некорректных текстовых заимствований без использования внешних источников.

Общая схема подхода состоит из нескольких связанных подзадач.

Сначала исследуемый текст подвергается процедуре сегментации, в результате которой он разделяется на отдельные сегменты. Если рассматривать текст с точки человека, то структурной единицей текста является предложение. Однако разбиение текста на предложения может быть не лучшей тактикой, в силу неравномерного распределения предложений по длине. Слишком короткие предложения могут создать трудности при сборе статистик из-за своего недостаточного размера. Поэтому предлагаются более подходящие методы для сегментирования текста. Среди них можно выделить разбиение по параграфам, разбиение с фиксированным шагом, разбиение по нескольким предложениям. Стоит отметить, что выбор конкретного способа сегментирования текста сильно зависит от условий использования итогового алгоритма. Так, при выделении текстового слоя некоторые инструменты могут терять структурную информацию о тексте — разбиение на предложения и параграфы. Тогда использование этой информации для сегментирования не представляется возможным.

Полученные сегменты текста необходимо каким-то образом векторизовать. Для этого предлагается использовать подход, основанный на представлении текста с помощью tf-idf статистик для входящих в него слов. Данный подход строит векторное представление текста с учетом некоторого корпуса текстовых документов, в рамках которого этот текст рассматривается. Предлагается использовать данный подход для векторизации сегментов рассматриваемого текста. То есть, tf-idf статистики слов собираются для слов сегмента, в рамках всего документа. Аналогично случаю векторизации текстов, слова, часто используемые во всем тексте и в конкретном сегменте получат маленький вес и наоборот, слова, которые специфичны для данного сегмента, но редко встречаются во всем тексте, получат более высокий вес.

Из построенных векторов сегментов необходимо построить ряд статистик, получаемый расчётом статистики для каждого сегмента. Довольно логично, что в рамках расчёта статистики надо сравнивать рассматриваемый сегмент со всем текстом. Поэтому в качестве такого показателя используется популярный метод сравнения векторов по некоторой метрике. То есть вектор каждого сегмента сравнивается с усреднённым вектором всех сегментов, который, по предположению, является представлением всего текста.

Также проведен базовый эксперимент, в рамках которого показано, что простой подход, основанный на расчете частот встречаемости слов в тексте, выделяет заимствованные фрагменты достаточно хорошо.

Глава 3. Поиск внутренних заимствований как самостоятельная система исследования текста на оригинальность

Задача поиска некорректных текстовых заимствований без использования коллекции внешних документов имеет множество различных формулировок. Наиболее общая постановка заключается в том, чтобы в исследуемом документе явно указать фрагменты, которые были некорректно заимствованы. При этом разбиение текста на сегменты не задано, что дополнительно усложняет задачу.

Таким образом, для решения такой задачи необходимо сначала задать некоторое разделение исходного текстового документа на сегменты. И среди полученных сегментов указать те, которые были некорректно использованы.

3.1 Постановка задачи

В рамках предложенного метода поиска заимствованных фрагментов 2.2 и при некотором заданном разбиении текста на сегменты, формально задача звучит следующим образом.

Пусть D — коллекция текстовых документов, d — текстовый документ, s_i — сегмент текста: $d = \bigcup s_i$, $d \in D$. Среди сегментов текста s_i необходимо выделить те, значение статистики которых $stat(\mathbf{s}_i)$ превосходит некоторый заданный порог значений $statThreshold$:

$$\text{outliers} = \{s : stat(\mathbf{s}_i) > statThreshold\}. \tag{3.1}$$

3.2 Критерии качества

В экспериментах используются критерии качества, применявшиеся в конкурсе PAN-2011 [106] по поиску внутренних заимстований. Предложенные критерии являются адаптацией известных критериев точности и полноты [107] для рассматриваемой задачи.

Обозначим за пару (s, d) последовательность символов, помеченную экспертом как заимствование в документе d. $S = \bigcup\limits_i s_i$ — совокупность всех заимствованных сегментов. За пару (r, d) обозначим последовательность, помеченную алгоритмом как заимствованную. Аналогично $R = \bigcup r_i$ — совокупность всех сегментов, которые алгоритм классифицировал как заимствованные. Рассмотрим меры качества *Precision* и *Recall*:

$$\begin{aligned} \mathrm{Prec}(S,R) &= \frac{1}{|R|} \sum_{r_j \in R} \frac{|\bigcup\limits_{s_i \in S} (s_i \cap r_j)|}{|r_j|}, \\ \mathrm{Rec}(S,R) &= \frac{1}{|S|} \sum_{s_i \in S} \frac{|\bigcup\limits_{r_j \in R} (s_i \cap r_j)|}{|s_i|}. \end{aligned} \tag{3.2}$$

Данные величины отражают точность (доля правильного распознавания заимствований по отношению ко всем выделенным сегментам) и полноту (доля правильного распознавания заимствований по отношению ко всем заимствованиям в тексте) работы алгоритма.

Вычисляется *F1-мера* как среднее гармоническое между *Precision* и *Recall* (3.2):

$$F1(S,R) = \frac{\mathrm{Prec}(S,R) \cdot \mathrm{Rec}(S,R)}{\mathrm{Prec}(S,R) + \mathrm{Rec}(S,R)}. \tag{3.3}$$

Вычисляется величина гранулярности:

$$\mathrm{gran}(S,R) = \frac{1}{|S_R|} \sum_{s_i \in S_R} |R_{s_i}|, \tag{3.4}$$

где S_R — множество заимствованных сегментов, обнаруженных алгоритмом, R_s — сегменты, отмеченные алгоритмом, которые детектируют данный сегмент заимствований s:

$$\begin{gathered} S_R = \{s | s \in S \wedge \exists r \in R : r \text{ детектирует } s\}, \\ R_S = \{r | r \in R \wedge r \text{ детектирует } s\}, \\ r \text{ детектирует } s : \text{если } r \cap s \neq \varnothing. \end{gathered} \tag{3.5}$$

Таким образом, гранулярность показывает то, насколько мелко алгоритм разбивает заимствованные сегменты текста. Если заимствованные сегменты разделяются алгоритмом на много мелких, то гранулярность будет иметь высокие значения.

По описанным величинам (3.3), (3.4) вычисляется итоговая мера качества *pladget*:

$$\text{pladget}(S,R) = \frac{F1(S,R)}{\log(1 + \text{gran}(S,R))}. \tag{3.6}$$

3.3 Общий подход

Для обнаружения заимствований исходный текст d разбивается на сегменты s_i:

$$d = \cup s_i. \tag{3.7}$$

Для каждого сегмента вычисляется вектор признаков $\mathbf{s}_i$ и строится статистика $stat(\mathbf{s}_i)$. Затем происходит детектирование выбросов среди значений статистики на основании ее отклонения от среднего значения $stat_{avr}(d) = \frac{1}{N}\sum_{i=1}^{N} stat(\mathbf{s}_i)$, где N — число сегментов в тексте. Если отклонение превышает заданный порог $statThreshold$, то сегмент считается заимствованным:

$$|stat(\mathbf{s}_i)) - stat_{avr}(d)| > statThreshold. \tag{3.8}$$

Корпус документов D разбивается на обучающую и тестовую выборки: $D = D_{test} \cup D_{train}$. При обучении параметры алгоритма $\boldsymbol{\omega}$ настраиваются таким образом, чтобы улучшить меры качества работы алгоритма. При фиксированном способе разбиения текста мера *Granularity* не изменяется, т. к. она зависит от мелкости разбиения. Тогда для увеличения итоговой меры качества *Pladget* достаточно улучшить *F1-меру*:

$$\hat{\boldsymbol{\omega}} = \arg\max_{\boldsymbol{\omega} \in \boldsymbol{\Omega}} F1(S, R). \tag{3.9}$$

Т. е. требуется вектор параметров $\boldsymbol{\omega} = (l_{segm}, n, statThreshold)$, максимизирующий *F1-меру*. Введены следующие обозначения: l_{segm} — минимальная длина сегмента, n — ширина окна сглаживания, $statThreshold$ — порог выброса.

3.3.1 Описание алгоритма

Предлагаемый алгоритм работает с частотными признаками, предоставляющими описание текста. В качестве такого признака выбран признак частоты встречаемости слов, описанный в формуле (2.11).

Исходный текст подвергается предобработке: удаляются служебные символы, все буквы переводятся в нижний регистр. Также из текста удаляются стоп-слова.

3.3.2 Сегментирование текста.

Текст разбивается на предложения. Затем формируется разбиение текста на сегменты s_i: если длина очередного предложения меньше минимальной длины сегмента l_{segm}, к этому предложению добавляется следующее за ним — процесс повторяется, пока длина сегмента s_i не превысит заданную минимальную длину. Минимальная длина сегмента l_{segm} является настраиваемым параметром алгоритма.

3.3.3 Построение статистики и детектирование аномалий.

Для каждого сегмента s_i текста строится вектор признаков. Затем строится статистика $stat(\mathbf{s}_i)$ на основе отклонения вектора признаков от усредненного по всему тексту вектора (2.13).

Полученная статистика сглаживается методом скользящего среднего: новые значения статистики $stat'(\mathbf{t}_i)$ вычисляются по формуле

$$stat'(\mathbf{s}_i) = \frac{1}{2n+1} \sum_{k=i-n}^{i+n} stat(\mathbf{s}_i), \tag{3.10}$$

где n — ширина сглаживания, которая также является настраиваемым параметром. Значения в крайних точках вычисляются по формуле (N — число сегментов)

$$\begin{aligned} stat'(\mathbf{s}_i) &= \frac{1}{i+n+1}\sum_{k=0}^{i+n} stat(\mathbf{s}_i), \\ stat'(\mathbf{s}_i) &= \frac{1}{i+n+1}\sum_{k=i-n}^{N} stat(\mathbf{s}_i). \end{aligned} \tag{3.11}$$

Полученные значения статистики $stat'(\mathbf{s}_i)$ исследуются на выбросы. Если в ряде статистики присутствует аномалия, превышающая заданный порог $statThreshold$, то сегмент s_i, отвечающий этому выбросу, помечается как заимствованный. Предлагаемый алгоритм можно описать в виде псевдокода (1).

Минимальная длина сегмента, ширина окна сглаживания и порог выброса настраиваются на обучающей выборке путем максимизации *F1-меры*.

Algorithm 1: Алгоритм обнаружения некорректно заимствованных сегментов текста

```
Input: Text document
vectorsList ← [ ];
segmentsList ← splitAndMerge(text);
for segment in segmentsList do
    segmentVector = vectorize(segment);
    vectorsList.append(vector);
outliersList ← [ ];
for vector in vectorsList do
    stat = calcStat(vector);
    if stat > statThreshold then
        outliersList.append(segment);
```

Алгоритм настраивался на частях 1–5 корпуса PAN-2011 (около 70% от всего объема) путем максимизации *F1-меры*. Тестирование проводилось на остальной части корпуса. Оптимальные параметры после настройки: $\hat{l}_{segm} = 450$, $\hat{n} = 8$, $\hat{\delta}_{susp} = 0{,}37$.

3.4 Вычислительный эксперимент

3.4.1 Описание данных

В эксперименте используется корпус текстовых документов конкурса PAN-2011 [106]. В текстах присутствуют сегменты настоящих, имитированных и искусственных заимствований. Каждый сегмент текста соответственно полностью взят из другого источника, либо заимствованный текст переписан человеком другими словами, либо специально обученный алгоритм строит текст, стараясь повторить стиль автора.

Выборка состоит из 4753 текстов, к каждому из текстов прилагается файл с экспертной разметкой заимствованных сегментов.

Для анализа корпуса была собрана подвыборка корпуса, состоящая из 30 документов, которые были просмотрены вручную. Анализ показал, что большая часть документов содержит в себе заимствования, сильно отличающиеся от остального текста по тематике и набору используемых слов. К примеру, в текст по экономике вставляется фрагмент, вырезанный из художественного текста.

Тексты корпуса были исследованы на то, сколько различных сегментов заимствований присутствует в тексте. На гистограмме (3.1) показано распределение этого показателя. Видно, что в большинстве, документы содержат не очень много сегментов чужого авторства. Тексты в среднем содержат от 1 до 7 сегментов заимствований. В большинстве текстов доля заимствований не превышает 4-5%, что усложняет задачу поиска этих заимствований.

Не менее важно то, какая доля заимствований содержится в каждом тексте, то есть было рассчитано отношение длины заимствованных сегментов к длине текста в символах. На гистограмме 3.1 приведено распределение количества текстов по доле заимствований.

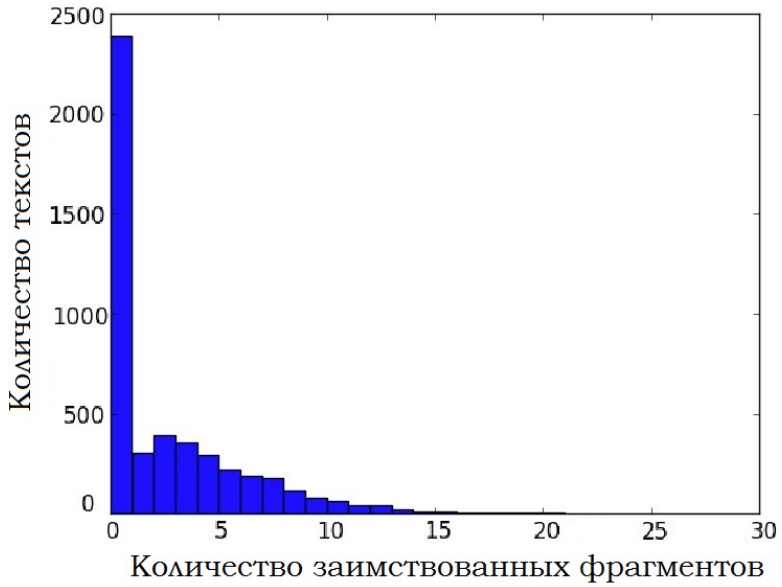


Рисунок 3.1 — Распределение текстов по количеству заимствованных сегментов.

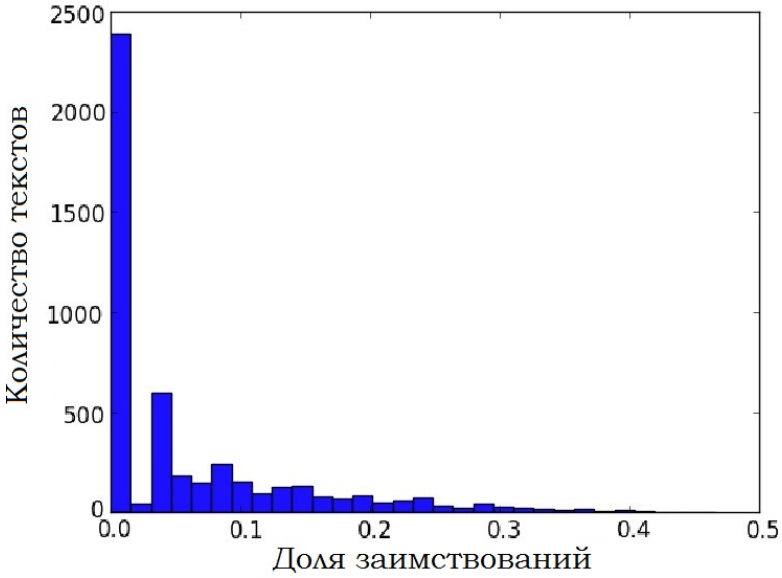


Рисунок 3.2 — Распределение текстов по доле заимствований.

3.4.2 Результаты эксперимента и примеры работы

На рисунках 3.3, 3.4 приведены примеры работы алгоритма. Красные участки обозначают заимствованные фрагменты, зеленым обозначен порог значения статистики, выше которого значения считаются выбросами.

На корпусе PAN-2011 алгоритм показал сравнимые результаты с победителем конкурса — алгоритмом Oberreuter [108]. Также было проведено сравнение

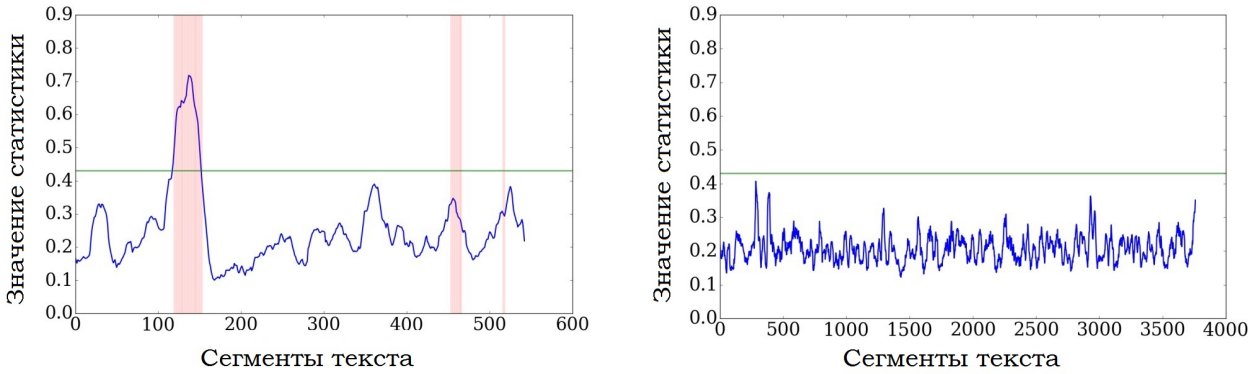


Рисунок 3.3 — Результаты на обучающей выборке

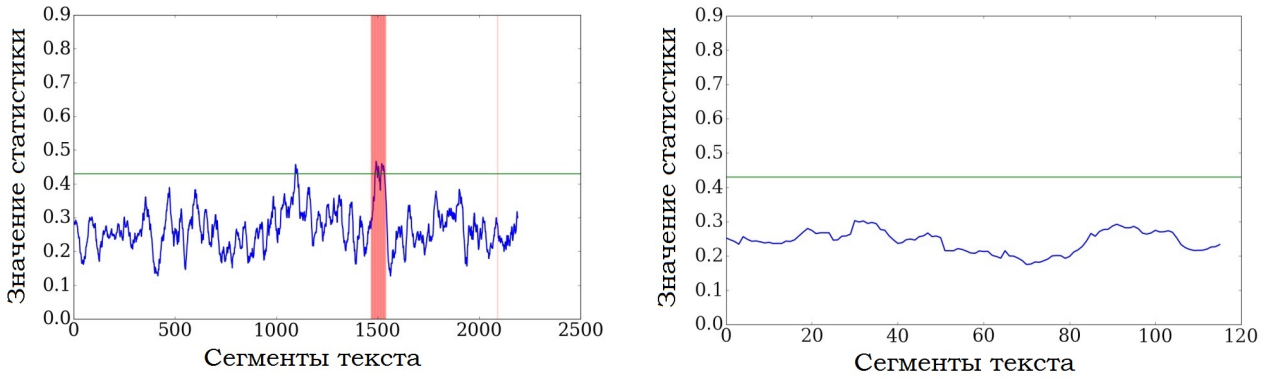


Рисунок 3.4 — Результаты на тестовой выборке

с алгоритмом [109], занявшим второе место на конкурсе. Качество работы предлагаемого алгоритма значительно превышает качество работы алгоритма [109]. Результаты работы и сравнение с двумя алгоритмами, приведены в таблице 1.

	Precision	Recall	F1	Granularity	Plagdet
Предлагаемый алгоритм	0.27	0.28	0.28	1.04	0.28
Oberreuter et al. [108]	0.34	0.31	0.33	1	0.33
Kestemont et al. [109]	0.11	0.43	0.17	1.03	0.17

Таблица 1 — Сравнение качества алгоритмов на корпусе PAN

3.5 Анализ ошибок

Результаты работы предлагаемого алгоритма зависят от длины документа. При анализе небольших по объему текстов сглаживание приводит к суще-

ственной потере информации об аномальных значениях статистики. При малой ширине сглаживания шумовые выбросы вызывают ложное срабатывание алгоритма.

3.6 Выводы к главе

В главе предложен алгоритм для обнаружения некорректно заимствованных сегментов текста без использования коллекции потенциальных источников заимстований.

Предлагаемый алгоритм использует распределение частот слов внутри текста для нахождения заимствованных сегментов. Сегментирование текста осуществляется по группам предложений. Для каждого сегмента строится статистика. Затем ряд статистики для всего текста сглаживается методом скользящего среднего. Полученные значения исследуются на отклонение от среднего значения для выявления заимствованных сегментов.

Алгоритм был настроен и протестирован на корпусе PAN-2011. Алгоритм Oberreuter [108], модификацией которого является предлагаемый алгоритм, показал на этом же корпусе результаты в 0,32 по *F1-мере*. Таким образом, описанный алгоритм показал сравнимые результаты при работе с тем же корпусом документов.

Глава 4. Поиск внутренних заимствований с использованием вспомогательных моделей векторизации текстов

Задача анализа текстов на предмет наличия неправомерных заимствований сильно востребована и, как следствие, имеет множество различных постановок [26—30]. Однако, как и во многих задачах анализа естественного языка, для получения качественных результатов необходимо иметь довольно большой корпус размеченных документов.

Для улучшения качества решения задачи часто прибегают к методам обучения без учителя (от англ. Unsupervised learning) [110]. Построение хорошей модели векторизации текста во многом определяет качество решения итоговой задачи. Метод обучения без учителя в данном случае подразумевает построение модели векторизации текстов без использования какой-либо информации о некорректных заимствованиях.

Предлагается использовать вспомогательную модель векторизации текста, параметры которой были настроены на большом корпусе текстовых документов. Предполагается, что такая модель может улучшить качество решения задачи поиска текстовых заимстований.

Для проверки данного подхода был реализован алгоритм по нахождению границ смены авторского стиля. Данная задача является вариацией задачи, описанной в разделе 3.1. Отличие заключается в том, надо указать не заимствованные фрагменты текста, а позиции текста, на которых происходит смена авторского стиля. Предлагаемый подход проверялся в рамках конкурса [28].

4.1 Критерии качества

В рамках конкурса [28] были предложены немного улучшенные метрики качества для оценивания качества работы алгоритмов поиска внутренних заимствований.

WindowDiff. Метрика, которая отражает общее качество сегментации текста. Она возвращает ошибку (между 0 и 1, где 0 соответствует идеальному

предсказанию) по предсказанию границ смены стиля. Границы, которые немного не совпадают с истинными штрафуются не так сильно, как лишние найденные границы или совсем пропущенные:

$$WindowDiff(ref, hyp) = \frac{1}{N-k}\sum_{i=1}^{N-k}(|b(ref_i, ref_{i+k}) - b(hyp_i, hyp_{i+k})| > 0),$$

где $b(i,j)$ обозначает количество границ между позициями i и j в тексте, а N обозначает число предложений в тексте, ref и hyp обозначают эталонное и предложенное сегментирование.

WinPR. Более общей метрикой, по сравнению с WindowDiff является метрика WinPR. Она обобщает популярные метрики полноты и точности (WinP и WinR соответственно), и таким образом дает более детальную картину качества предсказания.

$$True\ Positives = TP = \sum_{i=1-k}^{N}\min(R_{i,i+k}, C_{i,i+k}),$$

$$True\ Negatives = TN = -k(k-1) + \sum_{i=1-k}^{N}(k - \max(R_{i,i+k}, C_{i,i+k})),$$

$$False\ Positives = FP = \sum_{i=1-k}^{N}\max(0, C_{i,i+k} - R_{i,i+k}),$$

$$False\ Negatives = FN = \sum_{i=1-k}^{N}\max(0, R_{i,i+k} - C_{i,i+k}),$$

где R и C обозначают количество границ в эталонной разметке и предложенной алгоритмом соответственно, в окне i, с максимумом в k; N — количество объектов и k обозначает размер окна. На рисунке 4.1 схематично изображены разные случаи ошибок между эталонной и предложенной разметками.

Соответственно WinP, WinR, WinF рассчитываются как:

$$WinP = \frac{TP}{TP+FP},$$
$$WinR = \frac{TP}{TP+FN},$$
$$WinF = \frac{2 \cdot WinP \cdot WinR}{WinP + WinR}.$$

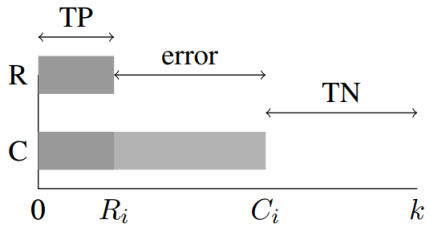


Рисунок 4.1 — Вычисление ошибки между эталонной и предсказанной разметками

4.2 Описание алгоритма

4.2.1 Модель векторизации сегментов текста

В качестве вспомогательной модели векторизации была выбрана модель Skip-Thought Vectors [111]. Данная модель представляет из себя нейросеть, параметры которой настроены для того, чтобы возвращать репрезентативный вектор входного предложения на естественном языке без использования какой-либо дополнительной информации. Причем вектора строятся с учетом предыдущего и последующего предложений, т.е. модель учитывает контекст и порядок следования предложений в тексте.

Используемая модель построена по распространенной схеме кодирования-декодирования (от англ. encoder-decoder scheme) [112]. Суть этой схемы состоит в том, что входной объект, в данном случае текст, сначала проходит стадию кодирования, в результате которой данный объект получает некоторый вектор фиксированной размерности. Затем полученный вектор проходит стадию декодирования, суть которой заключается в том, чтобы попытаться как можно более точно восстановить исходный объект. Параметры модели же настраиваются таким образом, чтобы минимизировать итоговую ошибку реконструкции объекта по его векторному представлению. Причем кодировщик и декодировщик представляют из себя рекуррентные нейронные сети (от англ. RNN – recurrent neural network) [113; 114], которые могут обрабатывать входные последовательности переменной длины.

Принципиальная схема модели Skip-Thought выглядит следующим образом. Пусть за $w_i^1, \ldots, w_i^N$ обозначены слова в предложении s_i, где N – длина

данного предложения в словах. На каждом шаге кодировщик генерирует вектор $\mathbf{h}_i^t$, который может быть интерпретирован как векторное представление текущей входной последовательности слов $w_i^1, \ldots, w_i^t$. А последний полученный вектор последовательности $\mathbf{h}_i^N := \mathbf{s}_i$ является векторным представлением всей последовательности s_i:

$$\begin{aligned} \mathbf{z}_t &= \sigma(\mathbf{W}_{zx}\mathbf{x}_t + \mathbf{W}_{zh}\mathbf{h}_{t-1}), \\ \mathbf{r}_t &= \sigma(\mathbf{W}_{rx}\mathbf{x}_t + \mathbf{W}_{rh}\mathbf{h}_{t-1}), \\ \tilde{\mathbf{h}}_t &= \tanh(\mathbf{W}_x\mathbf{x}_t + \mathbf{W}_h(\mathbf{r}_t \circ \mathbf{h}_{t-1})), \\ \mathbf{h}_t &= (1 - \mathbf{z}_t) \circ \mathbf{h}_{t-1} + \mathbf{z}_t \circ \tilde{\mathbf{h}}_t, \end{aligned} \tag{4.1}$$

где за $(\mathbf{W}_{zx}, \mathbf{W}_{zh}, \mathbf{W}_{rx}, \mathbf{W}_{rh}, \mathbf{W}_x, \mathbf{W}_h)$ — обозначены параметры кодировщика LSTM, $\mathbf{x}_t$ — векторное представление слова w_t, $(\circ)$ обозначает покомпонентное умножение. Под $\sigma()$ и $\tanh()$ понимаются так называемые нелинейные функции активации [115; 116]:

$$\begin{aligned} \sigma(x) &= \frac{1}{1 + e^{-x}}, \\ \tanh(x) &= \frac{e^x - e^{-x}}{e^x + e^{-x}}. \end{aligned}$$

Принципиальная схема работы декодировщика устроена аналогично.

В результате процесса векторизации предложения моделью [111] получается вектор размерности 4800. Причем он является конкатенацией двух независимых векторов, размерностей 2400 каждый. Первый вектор (uni-skip vector) является результатом векторизации, когда слова подаются в модель в прямом порядке (то есть в том порядке, в каком они представлены в предложении). Второй вектор (bi-skip vector) является результатом векторизации, когда слова подаются в модель сначала в прямом (первые 1200 компонент вектора bi-skip), а потом в обратном порядке (последние 1200 компонент вектора bi-skip).

Оптимальный размер используемого вектора подбирается в процессе подбора гиперпараметров данной задачи. Делается выбор между следующими вариантами:

– Весь 4800-мерный вектор
– 2400-мерный вектор uni-skip
– 2400-мерный вектор bi-skip

4.2.2 Сегментирование и построение статистик

В качестве процедуры сегментирования, описанной в разделе 2.2.1, в данном эксперименте выбран процесс разбиения на предложения при помощи стандартного инструмента анализа текстов [117]. Логику этоого решения можно объяснить тем, что модель векторизации [111] была настроена на работу именно с предложениями, поэтому в данном случае короткие и длинные предложения должны обрабатываться одинаково корректно.

Подсчет статистики для вектора $\mathbf{s}_i$ рассматриваемого сегмента s_i в документе d (раздел 2.2.3) происходит путем расчета среднего расстояния до каждого из остальных сегментов. В качестве функции расстояния используется косинусная мера близости [118]:

$$stat(s_i) = \frac{1}{|d|} \sum_{j \neq i} \cos(\mathbf{s}_i, \mathbf{s}_j), \tag{4.2}$$

где под $|d|$ понимается количество предложений в тексте.

Для обнаружения границ смены стиля статистика каждого сегмента сравнивается с пороговым значением. Если статистика превосходит это значение, то данный сегмент является выбросом, а начало сегмента считается границей смены стиля:

$$stat(s_i) > statThreshold \Rightarrow s_i \; is \; outlier. \tag{4.3}$$

Общую схему алгоритма можно представить в виде псевдокода 2.

4.3 Вычислительный эксперимент

4.3.1 Подбор гиперпараметров

Гиперпараметры алгоритма, порог статистики $statThreshold$ и размерность векторов skip thought, настраивались на валидационной выборке путем

Algorithm 2: Алгоритм нахождения границ смены авторского стиля

```
Input: Text document
vectorsList ← [ ];
segmentsList ← splitIntoSentences(text);
for segment in segmentsList do
    segmentVector = getSkipThoughtVector(segment);
    vectorsList.append(vector);
bordersList ← [ ];
for vector in vectorsList do
    stat = calcStat(vector);
    if stat > statThreshold then
        border = getSegmentStart(vector);
        bordersList.append(border);
```

максимизации целевой метрики $WinF$. Для каждого варианта размерности вектора был осуществлен подбор значения $statThreshold$ путем перебора по заданному промежутку значений. На рисунках 4.2, 4.3 представлены значения метрик качества для различных размерностей при варьировании порога статистики $statThreshold$. Вариант с использованием uni-skip векторов показал более высокие результаты, поэтому для них был проведен более детальный перебор значений $statThreshold$.

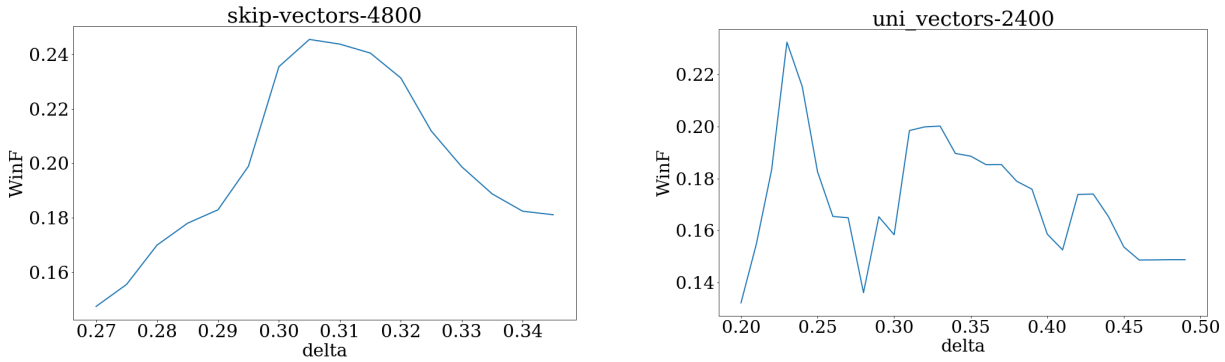


Рисунок 4.2 — Подбор оптимальных значений порога для модели skip (слева) и uni (справа) векторов.

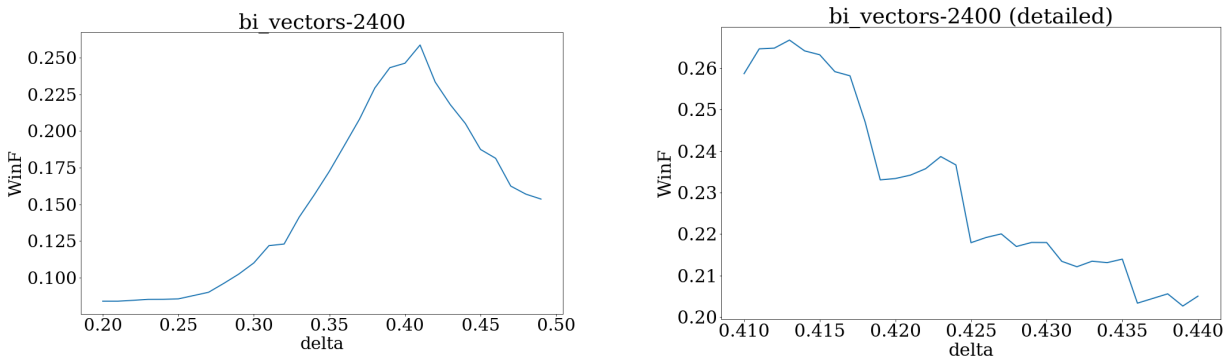


Рисунок 4.3 — Грубый (слева) и точный (справа) подбор оптимальных значений порога для модели bi векторов.

4.3.2 Результаты и примеры работы

Пример работы алгоритма на одном из текстов из тестовой подвыборки приведен на рисунке 4.4.

На графике представлен ряд значений статистики для каждого из сегментов текста. Также для наглядности приведена визуализация матрицы попарных расстояний между всеми векторами сегментов текста. Синими точками обозначены истинные границы смены авторского стиля. Можно отметить, что сегменты чужого авторства имеют значительно большее расстояние до остальных сегментов текста.

На рисунке 4.5 выделены сегменты текста, которые, по результатам работы алгоритма, признаны заимствованными. Горизонтальная линия — порог значения статистики, преодоление которого говорит о том, что соответствующий сегмент является некорректно заимствованным. Вертикальными линиями обозначены такие случаи.

В [119] приведены результаты сравнения предлагаемого алгоритма, а также алгоритмов [120] и [121]. Два последних алгоритма основаны на статистиком анализе частот встречаемости слов и символов, а также некоторых текстовых признаках, при этом они не используют вспомогательные модели векторизации текстов.

Качество работы алгоритма в сравнении с другими подходами представлено в таблице 2.

Целью данной работы является разработка системы предварительного анализа текста на предмет некорректных заимствований. Поэтому достаточно

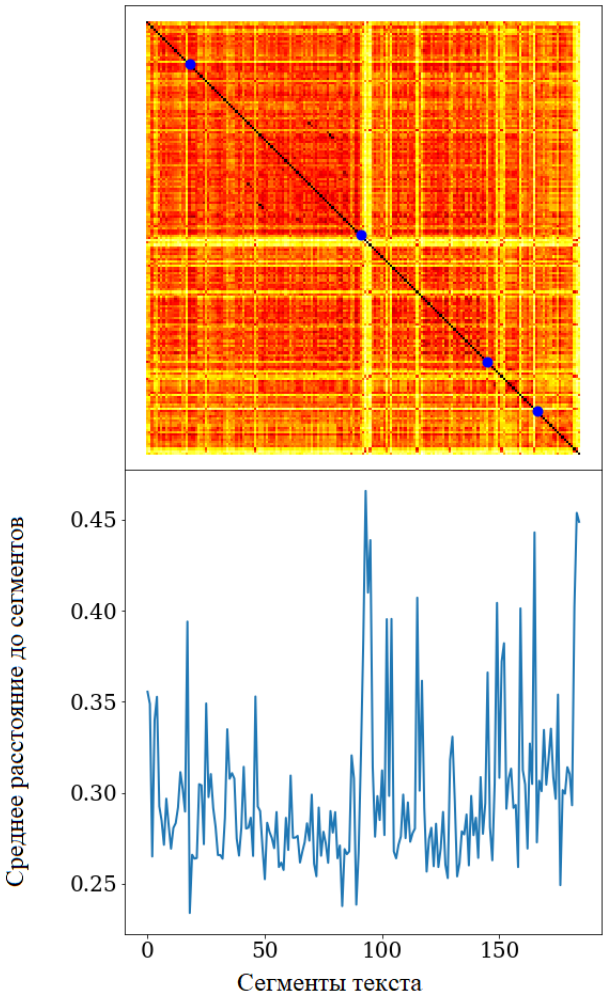


Рисунок 4.4 — Пример работы алгоритма

	WinP	WinR	WinF	WindowDiff
Предлагаемый алгоритм	0.371	0.543	0.277	0.529
Karaś, Śpiewak, Sobecki [121]	0.315	0.586	0.323	0.546
Khan [120]	0.399	0.487	0.289	0.480

Таблица 2 — Сравнение качества алгоритмов

важно, как алгоритм обрабатывает высокооригинальные тексты (т.е. без некорректных заимствований). В таблице 3 приведены метрики качества для подвыборки текстов, не содержащих вставок чужого авторства.

Стоит отметить, что предлагаемый алогоритм очень хорошо ведет себя с высокооригинальными текстами в том плане, что в них не происходят ложные

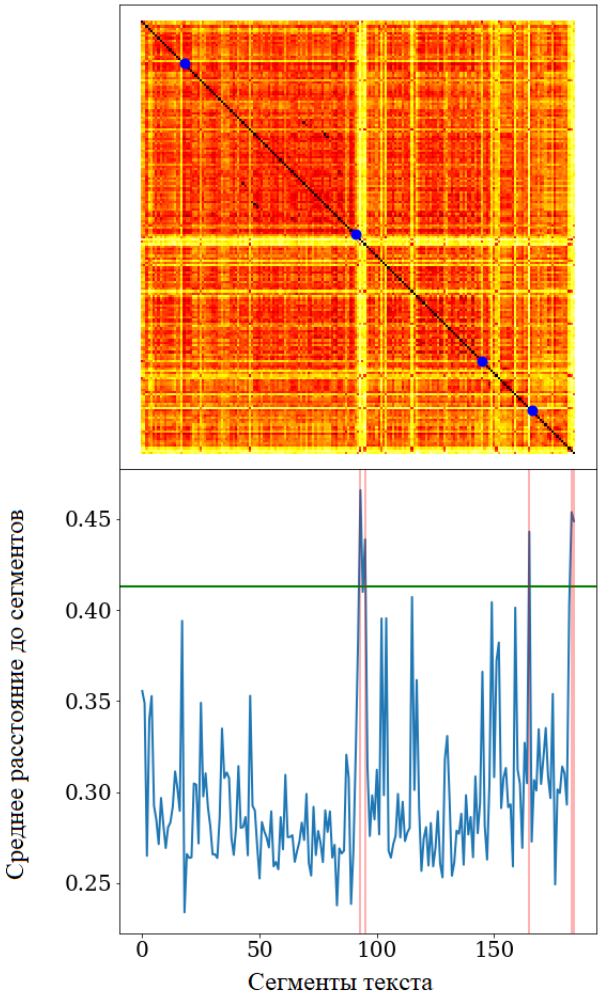


Рисунок 4.5 — Пример обнаружения границ смены стиля

	Количество авторов	WinF	WindowDiff
Предлагаемый алгоритм	1	1.	0.0
Karaś, Śpiewak, Sobecki [121]	1	0.059	0.145
Khan [120]	1	1.	0.0

Таблица 3 — Сравнение качества алгоритмов на подвыборке высокооригинальных текстов

срабатывания. Это значит, что алгоритм с высокой точностью отбирает тексты без смены авторского стиля.

Хотя все алгоритмы показывают примерно одинаковые результаты по качеству, стоит отметить, что алгоритмы [121] и [120] основаны на частотных при-

знаках и не используют вспомогательные векторные модели. Предлагаемый же алгоритм использует нейросетевую модель [111], которая требует гораздо больше вычислительных ресурсов. Это видно при сравнении времени работы всех предлагаемых алгоритмов на тестовой выборке в таблице 4.

	Время работы
Предлагаемый алгоритм	00:20:25
Karaś, Śpiewak, Sobecki [121]	00:01:19
Khan [120]	00:02:23

Таблица 4 — Сравнение времени работы алгоритмов

Архитектурно более сложная модель предсказуемо требует больше вычислительных ресурсов. В контексте данной работы это может являться большим недостатком, так как основная цель заключается в том, чтобы построить систему предварительной оценки документов на предмет некорректных заимствований. Такая система должна быть нетребовательной к вычислительным ресурсам и работать быстро.

4.4 Выводы к главе

В главе представлен метод по обнаружению некорректно заимствованных сегментов текста с использованием вспомогательной модели векторизации текста. В качестве алгоритма векторизации была выбрана модель [111], представляющая из себя рекуррентную нейронную сеть формата кодировщик-декодировщик. Параметры этой модели были настроены на внешнем корпусе в формате обучения без учителя.

Алгоритм обнаружения использует вспомогательную модель для векторизации сегментов текста, в качестве которых выбраны предложения. В качестве статистик используется расстояние от вектора каждого сегмента до усредненного вектора всех сегментов. В полученном ряде статистик ищутся выбросы, которые свидетельствуют о смене авторского стиля на границе соответствующих сегментов.

Предложенный алгоритм сравнен с двумя более простыми подходами, основанными на анализе частот слов в тексте и некоторых текстовых признаках.

Качество всех трех алгоритмов на тестовой выборке примерно одинаковое, при этом предлагаемый алгоритм с высокой степенью точности отбирает тексты, не содержащие смен авторского стиля, что является основной целью данной работы. Однако время работы алгоритма, использующего вспомогательную модель векторизации текста существенно выше других, что является большим недостатком данного подхода.

Глава 5. Система фильтрации высокооригинальных текстов на основе стилистического анализа

Как было сказано раннее, у задачи поиска некорректных текстовых заимствований существует множество различных постановок. Каждая из них формулируется для решения некоторой частной проблемы, поэтому необходимо сначала определить проблематику темы, чтобы выбрать корректную постановку задачи.

Данная работа рассматривает прежде всего случай проверки работ учащихся, которые должны подвергаться автоматической проверке на факт некорректных заимствований. То есть, если рассматривать отдельно взятую работу, результатом проверки этой работы должен являться вывод о том, содержит ли данная работа некорректные заимствования.

Согласно данной формулировке проблемы, становится ясно, что решаемая задача является задачей обнаружения факта мультиавторства [27]. При этом, задача может быть расширена до задачи нахождения границ нарушения стиля [28]. В базовой постановке задача обнаружения факта мультиавторства представляет собой задачу бинарной классификации текстов. То есть каждому рассматриваемому текстовому документу d_i необходимо присвоить метку класса.

5.1 Постановка задачи

Говоря более формально, нужно построить алгоритм $\mathcal{A}$, осуществляющий отображение простанства текстовых документов в пространство меток класса:

$$\mathcal{A} : D \to Y,$$

где $D = \{d_i\}_{i=1}^{N}$ – рассматриваемое пространство текстовых документов, $Y = \{y_0, y_1\}$ – пространство меток класса. В данном случае пространство меток класса соответствует двум классам:

– Класс 0 — класс высокооригинальных документов,

– Класс 1 — класс документов с заимствованиями.

Под высокооригинальным документом понимается документ, содержащий малое количество заимствований из любых других текстов (или не содержащий их вовсе). Соответственно, под документом с заимствованиями понимается текст, содержащий большое число вставок из других текстов.

5.2 Критерии качества

Основная цель предлагаемого алгоритма — отфильтровывать высокооригинальные документы, не пропуская при этом документы с заимствованиями. Поэтому, при оценке качества алгоритма важны, прежде всего, следующие показатели:

– полнота класса документов с заимствованиями,

– количество отфильтрованных высокооригинальных документов.

Под полнотой класса (recall) понимается следующая величина:

$$\text{Recall} = \frac{TP}{TP + FN},$$

где TP, FN – true-positive и false-negative объекты. То есть это тексты, которые верно отнесены алгоритмом к нужному классу и, наоборот, тексты, которые неверно отнесены алгоритмом к противоположному классу, соответственно. Таким образом, полнота класса — это доля верно найденных алгоритмом документов из всех документов этого класса.

Второй показатель (количество высокооригинальных документов) важен, так как можно привести пример крайнего случая, когда все документы относятся к классу документов с заимствованиями. Тогда полнота класса документов с заимствованиями будет равна 1, но ни одного документа отфильтровано не будет, и нагрузка на систему поиска внешних заимствований не уменьшится.

Количество отфильтрованных документов можно рассматривать как полноту класса высокооригинальных документов. То есть, нам важна полнота обоих классов, однако полнота класса документов с заимствованиями важнее. Можно ввести вспомогательную метрику, которую будем использовать при настройке гиперпараметров алгоритма:

$$\text{Recall}_\beta = \beta \cdot \text{Recall}_1 + \text{Recall}_0, \tag{5.1}$$

где Recall_1 и Recall_0 – полнота класса документов с заимствованиями и высокооригинальных документов соответственно. Для того, чтобы полнота класса 1 была в приоритете, весовой коэффициент берется β больше единицы.

5.3 Описание алгоритма

Общую логику работы предлагаемого алгоритма можно представить в виде псевдокода (3).

Algorithm 3: Алгоритм определения факта заимствования в тексте

```
Input: Text document
statsList ← [ ];
text ← preprocess(text);
segmentsList ← getSegments(text);
for segment in segmentsList do
    segmentVector = vectorize(segment);
    stat ← calcStat(vector);
    statsList.append(stat);
outliersCount ← 0;
for stat in statsList do
    if stat > statThreshold then
        outliersCount+ = 1;
if outliersCount > outliersThreshold then
    return 'text is not original';
else
    return 'text is original';
```

Алгоритм состоит из следующих основных этапов:

- предобработка текста,
- сегментация текста,
- векторизация сегментов,
- расчет статистик для сегментов,

– обнаружение выбросов в ряде статистик.

5.3.1 Предобработка текста

Сначала текст проходит процедуру предобработки. Используются стандартные техники обработки естественного языка: удаление редких символов, удаление стоп-слов (слов, не несущих смысловую нагрузку), приведение слов к начальной форме. Конкретные техники предобработки и их параметры подбираются при настройке алгоритма на конкретном корпусе документов.

5.3.2 Сегментация текста

Под сегментацией текста понимается процедура разбиения текста d на сегменты s_j:

$$d = \bigcup_{j=1}^{m} s_j.$$

При этом сегменты могут быть пересекающимися и, наоборот, иметь нулевое пересечение. Для каждого сегмента затем будет рассчитываться некоторая статистика, поэтому сегментирование должно удовлетворять некоторым условиям.

Разбиение на сегменты должно быть достаточно мелким, чтобы можно было детектировать выброс в ряде статистик, и значение статистики на некотором сегменте сильно отличалось от остальных значений. С другой стороны, размер отдельного сегмента должен быть достаточно велик, чтобы можно было посчитать адекватную статистику.

Самые популярные стратегии сегментации, которые применимы в данном случае:

– разбиение по параграфам,

– разбиение окном с фиксированным шагом.

В процессе настройки гиперпараметров алгоритма, выбирается стратегия сегментации, а также ее аргументы (ширина окна и размер шага).

5.3.3 Векторизация сегментов

Каждый сегмент текста подвергается процедуре векторизации. Для построения векторного представления сегмента используются частоты словесных и символьных n-грамм.

Под символьной или словесной n-граммой понимается последовательность из n символов или слов в тексте.

Каждой n-грамме w в тексте d ставится в соответствие число:

$$freq_w = \frac{cnt(w)}{\sum\limits_{w' \in d} cnt(w')} \cdot \log\left(\frac{m}{seg(w)}\right), \tag{5.2}$$

где $cnt(w)$ – число вхождений w в текст d, m – число сегментов в тексте, $seg(w)$ – число сегментов, содержащих w.

Вектор сегмента формируется из рассчитанных величин (5.2) для всех уникальных n-грамм в текста. Если n-грамма есть в тексте, но отсутствует в сегменте, то значение (5.2) равно 0.

Тип рассматриваемых n-грамм (символьные или словесные) выбирается при настройке гиперпараметров алгоритма на конкретном корпусе документов.

5.3.4 Подсчет статистик и нахождение аномалий

Для рассматриваемого текста строится ряд статистик путем подсчета некоторой статистики для каждого сегмента текста. В качестве статистики выбрано расстояние от вектора сегмента s_i до усредненного вектора всех сегментов $\overline{s}$:

$$\overline{s} = \frac{1}{m}\sum_{j=1}^{m} s_j.$$

Тип расстояния между векторами также выбирается при настройке алгоритма. Выбор происходит из следующих вариантов:

– евклидово расстояние,

– косинусное расстояние.

Полученный ряд статистик сглаживается скользящим средним фиксированной ширины.

В полученном ряде статистик выполняется поиск выбросов. Под выбросом подразумевается значение статистики, которое превышает некоторый заданный порог (который подбирается при настройке гиперпараметров). По количеству выбросов в тексте принимается решение об оригинальности документа.

5.4 Вычислительный эксперимент

Для проверки качества предложенного алгоритма было проведено два вычислительных эксперимента. Первый эксперимент был проведен на корпусе англоязычных документов, подготовленных в рамках конкурса по обнаружению текстовых заимствований PAN-2020. Для проведения второго эксперимента был использован корпус русскоязычных текстов Paraplag [122], специально составленный для проверки алгоритмов поиска текстовых заимствований.

В рамках каждого эксперимента производилась настройка гиперпараметров описанного алгоритма. Для настройки, данные разбивались на обучающую и тестовую выборки, в размерах 70% и 30% от всего корпуса соответственно. Настройка гиперпараметров производилась с помощью кросс-валидации на трех разбиениях обучающей выборки. В качестве целевой метрики использовалась предложенная метрика (5.1).

5.4.1 Описание данных

В первой части вычислительного эксперимента используется корпус текстов, подготовленных и размеченных в рамках конкурса PAN-2020 [123]. Корпус содержит документы на английском языке. Каждый документ может содержать от 0 до 10 вставок текста другого авторства.

Корпус состоит из двух частей. Первая часть представляет из себя узкоспециализированный набор документов — все документы в ней посвящены теме технологий. Вторая часть корпуса является набором текстов различной

тематики (путешествия, философия, экономика, история и т.д.). Это сделано для того, чтобы была возможность протестировать предлагаемые алгоритмы на устойчивость к смене тематики при работе с документами. Количество документов с заимствованиями примерно равно количеству высокооригинальных документов.

Для второй части эксперимента был использован русскоязычный корпус тексов, содержащий документы с заимствованиями Paraplag [122]. Корпус представляет из себя набор текстов (эссе), в которые авторы намеренно добавляли заимствования из других документов. В качестве высокооригинальных документов представлены источники этих заимствований (статьи из энциклопедий). Доля документов с заимствованиями относительно всего корпуса около 15%.

5.5 Результаты эксперимента

Для оценки итогового качества полученного алгоритма, была рассчитана полнота каждого из целевых классов на тестовой выборке. Результаты экспериментов приведены в таблице (5).

Название корпуса	Язык	Доля класса 1 в корпусе	Полнота класса 0	Полнота класса 1
PAN-2020	английский	50%	10%	94%
Paraplag	русский	15%	32%	97%

Таблица 5 — Результаты экспериментов

Видно, что на обоих корпусах полнота класса 1 близка к 100%. Это значит, что малая часть документов, которым необходима детальная проверка с помощью системы поиска внешних заимствований, будут неправомерно отфильтрована. При этом, часть высокооригинальных документов будет правильно отсеяна, что снизит нагрузку на систему.

Примеры работы алгоритма на конкретных текстах приведены на графиках (5.1) и (5.2). Синими линиями представлены значения рядов статистик для каждого текста, красной горизонтальной — верхний порог статистики, выше которого она считается выбросом.

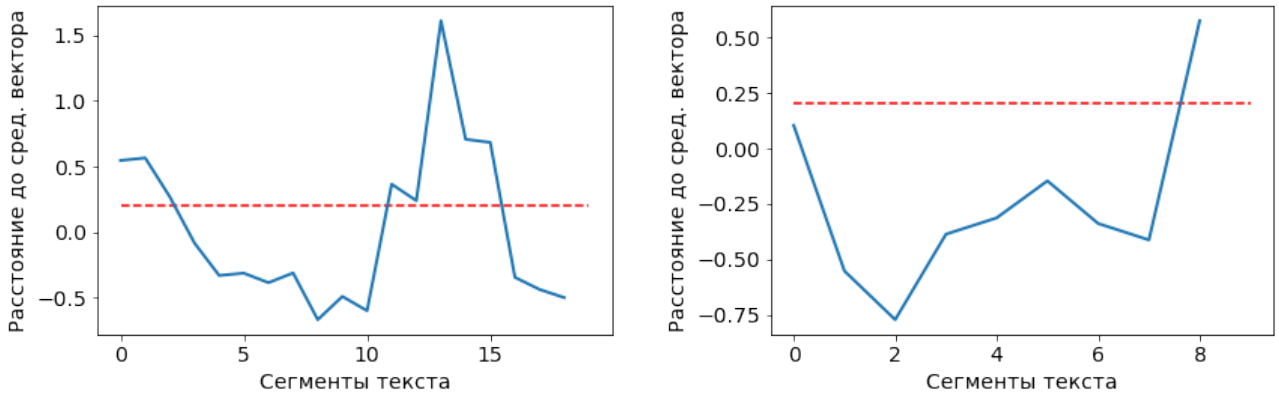


Рисунок 5.1 — Пример работы на тексте с заимствованиями (слева) и на высокооригинальном тексте (справа). Английский корпус документов.

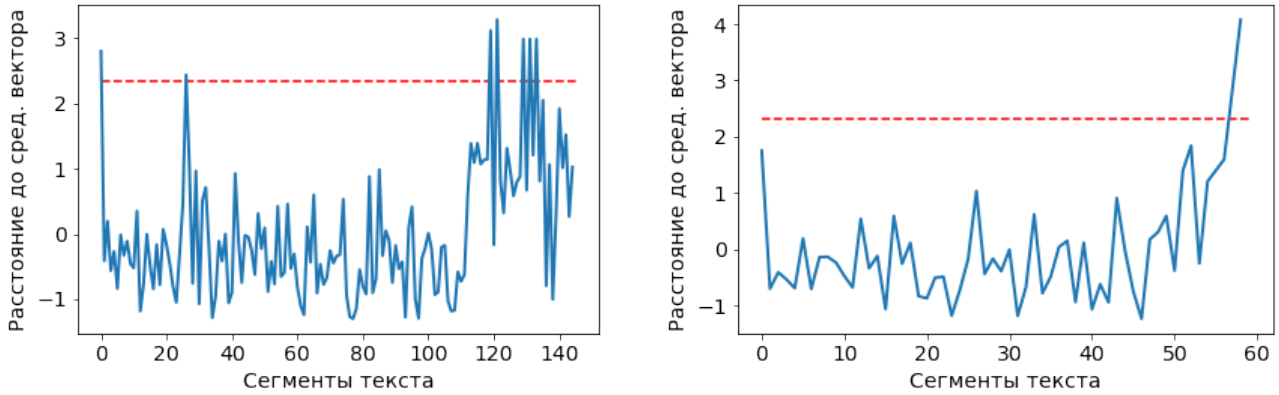


Рисунок 5.2 — Пример работы на тексте с заимствованиями (слева) и на высокооригинальном тексте (справа). Русский корпус документов.

5.6 Детали реализации программного комплекса

На основе предлагаемого алгоритма фильтрации высокооригинальных текстов был реализован программный комплекс. Комплекс предназначен для внедрения в систему выявления некорректных текстовых заимствований с использованием внешних текстовых коллекций. Цель внедрения программного комплекса заключается в снижении нагрузки на основную систему в моменты пиковых нагрузок. Снижение нагрузки происходит путём отбора документов, не требующих детальной проверки. При этом все документы проходят базовую проверку при помощи алгоритма шинглов [5]. Документы же, требующие детальной проверки, проходят полную проверку с использованием ресурсоемких методов (поиск перефразирований, переводных заимстований и т.д.).

Общая схема программного комплекса, внедрённого в промышленную систему приведён на диаграмме 5.3. Включение дополнительной фильтрации может осуществляться вручную по мере необходимости либо автоматически, при повышенной нагрузке на основную систем (выше определённого количества документов в минуту). Подразумевается, что при приемлемой нагрузке фильтрация документов не осуществляется.

5.6.1 Формат входных данных и предобработка

Модуль фильтрации высокооригинальных документов работает с текстовым слоем текстового документа без форматирования и дополнительных медиа файлов (картинок, диаграм и подобного) — так называемым текстовым слоем документа. Извлечение текстового слоя из документа происходит на стороне промышленной системы выявления заимствований. При этом при извлечении текстовых слоев из документов с форматированием (.doc, .docx, .pdf и подобные) остаются артефакты обработки, такие как: лишние переносы слов, пустые строки и подобное. В связи с этим был добавлен модуль предобработки текста, цель которого заключается в устранении артефактов извлечения текстового слоя.

Модуль предобработки удаляет лишние пробелы, символы переноса строк и неотображаемые символы. Также удаляются символы, не нужные для анализа текста. Как правило это все неалфавитные символы, включая цифры, так как они не несут содержательного смысла, а лишь увеличивают размер словаря. Исключением для удаления являются спецсимволы, характерные для конкретной области знаний.

5.6.2 Модуль фильтрации

Логика работы алгоритма отбора высокооригинальных документов, не нуждающихся в детальной проверке, описана в разделе 5.3. Результатом работы модуля фильтрации является бинарный ответ, свидетельствующий о необ-

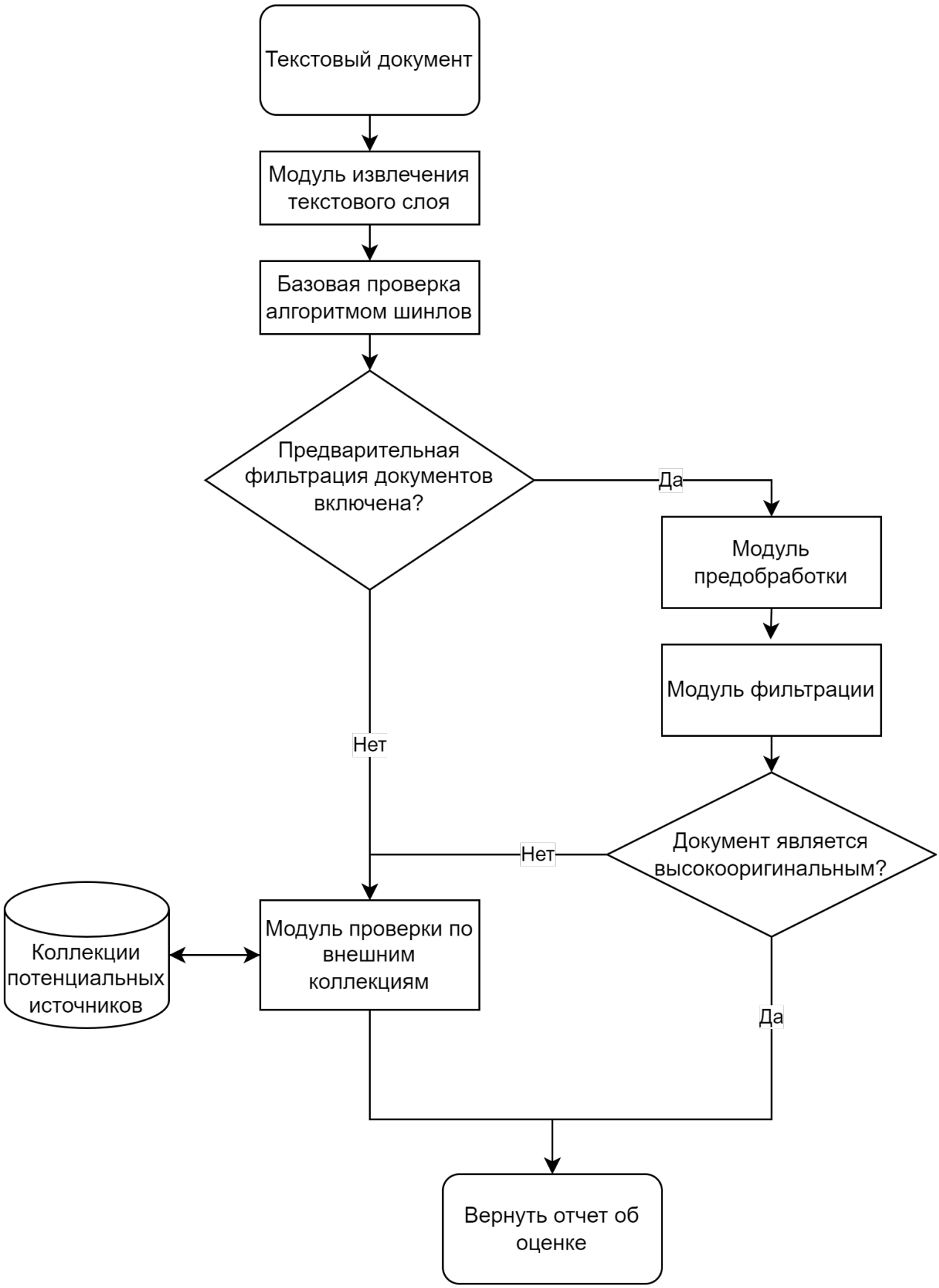


Рисунок 5.3 — Принципиальная схема программного комплекса

ходимости проеведения детальной проверки проверяемого документа. В случае положительного ответа, документ направляется в систему проверки на деталь-

ную проверку. В случае отрицательного ответа, полноценная проверка для рассматриваемого документа опускается.

5.7 Выводы к главе

В главе предложен алгоритм по обнаружению факта заимствований в тексте. При этом алгоритм анализирует текст изолированно, не используя внешнюю коллекцию возможных источников заимствований. Для установления факта заимствования, текст сегментируется, и для каждого сегмента рассчитывается статистика, основанная на частотах распределения n-грамм. В полученном ряде статистик происходит поиск выбросов, и по их количеству делается вывод о степени оригинальности текста.

Предложенный алгоритм был настроен и протестирован на корпусах английских и русских текстов. В обоих случаях алгоритм корректно отбирает высокооригинальные документы, оставляя при этом документы с заимствованиями для дальнейшей проверки. Таким образом, алгоритм удовлетворяет выдвинутым к нему требованиям по качеству и может быть использован в качестве первичного отбора документов при использовании высоконагруженной системы поиска заимствований по внешнему корпусу.

Как было сказано, основная цель предложенного алгоритма заключается в фильтрации высокооригинальных документов, для которых не требуется детальная проверка. Конечно, в идеальном случае все поступающие на проверку документы должны проходить полноценную проверку. Однако реальность такова, что при высокой нагрузке, система поиска внешних заимствований может сильно задерживать ответ или пропускать документы. В такой постановке кажется логичным пожертвовать малым (частью документов, с низкой долей заимствований), чтобы сохранить работоспособность системы.

Предложенный алгоритм как раз и осуществляет такую логику. При сравнительно небольшом количестве пропущенных документов с заимствованиями (около 3% для русского корпуса), удалось сократить поток документов для обработки почти на треть.

Также описана принципиальная схема разработанного программного комплекса, который внедряется в промышленную систему поиска текстовых заимствований.

Заключение

Основные результаты работы заключаются в следующем.

В главе 1 рассмотрены различные постановки решаемой задачи, ее актуальность и востребованность в современном научном сообществе. Также приведен краткий исторический экскурс, начиная от возникновения первых подходов, их развития до состояния на сегодняшний день. Первые подходы были основаны на статистическом анализе распределения слов и других лингвистических единиц. Позже эти методы стали использоваться в качестве построения признакового пространства для алгоритмов машинного обучения. В последенее же время большую популярность приобретают алгоритмы, использующие различные архитектуры нейронных сетей.

В главе 2 приведено описание основного метода, который предлагается для решения задачи поиска некорректных текстовых заимствований без использования внешних источников. Основной идеей является использование метода веторизации сегментов текста с помощью статистик tf-idf, подсчитанных для этих сегментов аналогично тому, как происходит векторизация текстов в рамках некоторого корпуса документов. Также приведен базовый эксперимент, подтверждающий жизнеспособность предлагаемого метода.

В главе 3 описывается система поиска некорректных заимствований с явным указанием подозрительных участков текста. Таким образом такая система является аналогом системы поиска с привлечением внешней коллекции потенциальных источников. Приводится эксперимент, в котором тестируется предлагаемая система поиска. Хотя результаты работы неплохие, их качества недостаточно для того, чтобы использовать такую систему в качестве альтернативы полноценной системе поиска по внешней коллекции источников.

В главе 4 также описывается система поиска заимствований с указанием подозрительных участков текста. Ее отличие заключается в том, что она использует вспомогательную модель векторизации предложений. Экспериментально показывается, что качество работы такой системы сравнимо с аналогами, основанными на статистическом анализе, однако время работы гораздо выше, что делает такую систему малопригодной для использования в промышленной сфере.

В главе 5 приводится описание алгоритма отбора документов, не содержащих некорректных текстовых заимствований. Так как постановка задачи не подразумевает указания конкретных сегментов текста, задача сводится к задаче бинарной классификации. Представлено решение этой задачи, основанное на предыдущих алгоритмах. Экспериментально показано, что предлагаемый алгоритм позволяет отфильтровывать до 30% высокооригинальных документов без потерь в классе документов с заимствованиями.

Список литературы

1. *Никитов, А. В.* Плагиат в работах студентов и аспирантов: проблема и методы противодействия / А. В. Никитов, О. А. Орчаков, Ю. В. Чехович //. — 2012.
2. *Stein, B.* Plagiarism analysis, authorship identification, and near-duplicate detection PAN'07 / B. Stein, M. Koppel, E. Stamatatos // SIGIR Forum. — 2007. — Vol. 41, no. 2. — P. 68—71. — URL: https://doi.org/10.1145/1328964.1328976.
3. *Chekhovich, Y. V.* Analysis of duplicated publications in Russian journals / Y. V. Chekhovich, A. V. Khazov // Journal of Informetrics. — 2022. — Vol. 16, no. 1. — P. 101246. — URL: https://www.sciencedirect.com/science/article/pii/S1751157721001176.
4. *Зеленков, И. В.* Сравнительный анализ методов определения нечетких дубликатов для Web-документов / И. В. Зеленков, И. В. Сегалович // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: Тр. 9-й Всеросс. научн. конф. RCDL. — Переславль-Залесский: Университет г. Переславля. — 2007.
5. Система распознавания интеллектуальных заимствований «Антиплагиат» / Ю. Журавлев [и др.] // Математические методы распознавания образов: 12-я Всероссийская конференция: Сборник докладов. — 2005.
6. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection / R. Socher [et al.] // NIPS. — 2011.
7. *Кузнецова, Р. В.* Методы обнаружения переводных заимствований в больших текстовых коллекциях / Р. В. Кузнецова, О. Ю. Бахтеев, Ю. В. Чехович // Информатика и её применения. — 2021. — т. 15, № 1. — с. 30—41.
8. *Е. М. Ешилбашян.* Поиск заимствований в армянских текстах путем внутреннего стилометрического анализа / Е. М. Ешилбашян, А. А. Асатрян, Ц. Г. Гукасян // Труды ИСП РАН. — 2021. — т. 33, № 1. — с. 209—224.
9. *Eissen, S. M. z.* Intrinsic Plagiarism Detection / S. M. z. Eissen, B. Stein // Advances in Information Retrieval. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2006. — P. 565—569.

10. *Muhr, M.* External and Intrinsic Plagiarism Detection Using Vector Space Models / M. Muhr, M. Zechner, R. Kern // CEUR Workshop Proceedings. — 2009. — Jan. — Vol. 502.

11. Outlier-Based Approaches for Intrinsic and External Plagiarism Detection / G. Oberreuter [et al.] // KES. — 2011.

12. *Stamatatos, E.* Intrinsic Plagiarism Detection Using Character n-gram Profiles / E. Stamatatos //. — 2009.

13. *Bensalem, I.* Intrinsic Plagiarism Detection using N-gram Classes / I. Bensalem, P. Rosso, S. Chikhi //. — 01/2014.

14. *Tschuggnall, M.* Countering Plagiarism by Exposing Irregularities in Authors' Grammar / M. Tschuggnall, G. Specht // Proceedings - 2013 European Intelligence and Security Informatics Conference, EISIC 2013. — 2013. — Aug. — P. 15—22.

15. *Романов, А. С.* Методика проверки однородности текста и выявления плагиата на основе метода опорных векторов и фильтра быстрой корреляции / А. С. Романов, Р. В. Мещеряков, З. И. Резанова // Доклады Томского государственного университета систем управления и радиоэлектроники. — 2014.

16. *Safin, K.* Style Breach Detection with Neural Sentence Embeddings / K. Safin, R. Kuznetsova // Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Dublin, Ireland, September 11-14, 2017. Vol. 1866 / ed. by L. Cappellato [et al.]. — CEUR-WS.org, 2017. — (CEUR Workshop Proceedings).

17. Methods for Intrinsic Plagiarism Detection and Author Diarization / M. P. Kuznetsov [et al.] // Working Notes of CLEF 2016 - Conference and Labs of the Evaluation forum, Évora, Portugal, 5-8 September, 2016. Vol. 1609 / ed. by K. Balog [et al.]. — CEUR-WS.org, 2016. — P. 912—919. — (CEUR Workshop Proceedings). — URL: http://ceur-ws.org/Vol-1609/16090912.pdf.

18. *Gillam, L.* Quite Simple Approaches for Authorship Attribution, Intrinsic Plagiarism Detection and Sexual Predator Identification / L. Gillam, A. Vartapetiance. — 2012.

19. Overview of the 3rd International Competition on Plagiarism Detection. / M. Potthast [et al.] //. — 01/2011.

20. *Stamatatos, E.* A survey of modern authorship attribution methods / E. Stamatatos // J. Assoc. Inf. Sci. Technol. — 2009. — Vol. 60, no. 3. — P. 538—556. — URL: https://doi.org/10.1002/asi.21001.

21. *Jones, K. S.* A statistical interpretation of term specificity and its application in retrieval / K. S. Jones // Journal of Documentation. — 1972. — Vol. 28. — P. 11—21.

22. *К. Ф. Сафин.* Определение заимствований в тексте без указания источника / К. Ф. Сафин, М. П. Кузнецов, М. В. Кузнецова // Информ. и её примен. — 2017. — т. 11, № 3.

23. *Safin, K.* Detecting a Change of Style using Text Statistics: Notebook for PAN at CLEF 2018 / K. Safin, A. Ogaltsov // Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, Avignon, France, September 10-14, 2018. Vol. 2125 / ed. by L. Cappellato [et al.]. — CEUR-WS.org, 2018. — (CEUR Workshop Proceedings).

24. Near-duplicate handwritten document detection without text recognition / O. Bakhteev [et al.] // Computational Linguistics and Intellectual Technologies. — 2021.

25. *К. Ф. Сафин.* О комбинированном алгоритме обнаружения заимствований в текстовых документах / К. Ф. Сафин, Ю. В. Чехович // Труды Института системного программирования РАН. — 2022. — т. 34, № 1. — с. 151—160.

26. Clustering by Authorship Within and Across Documents / E. Stamatatos [et al.] // CLEF. — 2016.

27. Overview of the Author Identification Task at PAN-2018: Cross-domain Authorship Attribution and Style Change Detection / M. Kestemont [et al.] // CLEF. — 2018.

28. Overview of the Author Identification Task at PAN-2017: Style Breach Detection and Author Clustering / M. Tschuggnall [et al.] // CLEF. — 2017.

29. Overview of the Style Change Detection Task at PAN 2019 / E. Zangerle [et al.] // CLEF. — 2019.

30. Overview of PAN 2021: Authorship Verification, Profiling Hate Speech Spreaders on Twitter, and Style Change Detection: Extended Abstract / J. Bevendorff [et al.] //. — 03/2021. — P. 567—573.

31. *Jurafsky, D.* Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition / D. Jurafsky, J. H. Martin. — 1st. — USA : Prentice Hall PTR, 2000.

32. *Holmes, D. I.* The Evolution of Stylometry in Humanities Scholarship / D. I. Holmes // Literary and Linguistic Computing. — 1998. — Vol. 13. — P. 111—117.

33. *Mendenhall, T. C.* The Characteristic Curves of Composition / T. C. Mendenhall. — 1887. — URL: https://doi.org/10.1126/science.ns-9.214s.237.

34. *Fucks, W.* On Mathematical Analysis of Style / W. Fucks // Biometrika. — 1952. — Vol. 39, no. 1/2. — P. 122—129. — URL: http://www.jstor.org/stable/2332470 (visited on 04/12/2022).

35. *Yule, G. U.* On sentence-length as a statistical characteristic of style in prose: With application to two cases of disputed authorship / G. U. Yule // Biometrika. — 1939. — Vol. 30. — P. 363—390.

36. *Koppel, M.* Authorship verification as a one-class classification problem / M. Koppel, J. Schler //. — 01/2004.

37. *Tony, C.* Language Inference from Function Words / C. Tony, I. Witten. — 1995. — Feb.

38. *Khamis, S.* Inference and Disputed Authorship: The Federalist / S. Khamis, F. Mosteller, D. L. Wallace // Journal of the American Statistical Association. — 1966. — Vol. 34. — P. 277.

39. *Tweedie, F. J.* Neural network applications in stylometry: The *Federalist Papers* / F. J. Tweedie, S. Singh, D. I. Holmes // Comput. Humanit. — 1996. — Vol. 30, no. 1. — P. 1—10. — URL: https://doi.org/10.1007/BF00054024.

40. *Juola, P.* A Controlled-corpus Experiment in Authorship Identification by Cross-entropy / P. Juola, H. Baayen // Literary and Linguistics Computing. — 2005. — Jan. — Vol. 20.

41. *Matthews, R.* Neural Computation in Stylometry I: An Application to the Works of Shakespeare and Fletcher / R. Matthews, T. Merriam // Literary and Linguistic Computing. — 1993. — Vol. 8. — P. 203—209.

42. A framework for authorship identification of online messages: Writing-style features and classification techniques / R. Zheng [et al.] // J. Assoc. Inf. Sci. Technol. — 2006. — Vol. 57, no. 3. — P. 378—393. — URL: https://doi.org/10.1002/asi.20316.

43. *Pearson, K.* LIII. On lines and planes of closest fit to systems of points in space / K. Pearson // Philosophical Magazine Series 1. — 1901. — Vol. 2. — P. 559—572.

44. *Burrows, J. F.* Word-Patterns and Story-Shapes: The Statistical Analysis of Narrative Style / J. F. Burrows // Literary and Linguistic Computing. — 1987. — Vol. 2. — P. 61—70.

45. *Biber, D.* Dimensions of Register Variation: A Cross-Linguistic Comparison / D. Biber //. — 1995.

46. *Flesch, R. F.* A new readability yardstick. / R. F. Flesch // The Journal of applied psychology. — 1948. — Vol. 32 3. — P. 221—33.

47. *Eissen, S. M. zu.* Plagiarism Detection Without Reference Collections / S. M. zu Eissen, B. Stein, M. Kulig // GfKl. — 2006.

48. *Torres, M.* The Cloze Procedure as a Test of Plagiarism: The Influence of Text Readability / M. Torres, M. Roig // The Journal of Psychology. — 2005. — Vol. 139. — P. 221—232.

49. *Cortes, C.* Support-Vector Networks / C. Cortes, V. Vapnik // Mach. Learn. — USA, 1995. — Vol. 20, no. 3. — P. 273—297. — URL: https://doi.org/10.1023/A:1022627411411.

50. Authorship Attribution with Support Vector Machines / J. Diederich [et al.] // Appl. Intell. — 2003. — Vol. 19, no. 1/2. — P. 109—123. — URL: https://doi.org/10.1023/A:1023824908771.

51. An integrated approach for intrinsic plagiarism detection / M. Alsallal [et al.] // Future Generation Computer Systems. — 2017. — Dec. — Vol. 96.

52. *Stein, B.* Intrinsic Plagiarism Analysis / B. Stein, N. Lipka, P. Prettenhofer // Lang. Resour. Eval. — Berlin, Heidelberg, 2011. — Mar. — Vol. 45, no. 1. — P. 63—82. — URL: https://doi.org/10.1007/s10579-010-9115-y.

53. *Mosteller, F.* Inference in an Authorship Problem / F. Mosteller, D. L. Wallace // Journal of the American Statistical Association. — 1963. — Vol. 58, no. 302. — P. 275—309. — URL: http://www.jstor.org/stable/2283270 (visited on 05/09/2022).

54. Syntactic Clustering of the Web / A. Z. Broder [et al.] // Comput. Networks. — 1997. — Vol. 29. — P. 1157—1166.

55. *Sanderson, C.* On Authorship Attribution via Markov Chains and Sequence Kernels / C. Sanderson, S. Günter //. Vol. 3. — 01/2006. — P. 437—440.

56. *Kjell, B.* Discrimination of Authorship Using Visualization / B. Kjell, W. A. Woods, O. Frieder // Inf. Process. Manag. — 1994. — Vol. 30. — P. 141—150.

57. *Juola, P.* Authorship attribution / P. Juola // Foundations and Trends® in Information Retrieval. — 2008. — Mar. — Vol. 1. — P. 233—334.

58. *Koppel, M.* Computational methods in authorship attribution / M. Koppel, J. Schler, S. E. Argamon // J. Assoc. Inf. Sci. Technol. — 2009. — Vol. 60. — P. 9—26.

59. *Ц. Г. Гукасян.* Векторные модели на основе символьных н-грамм для морфологического анализа текстов / Ц. Г. Гукасян // Труды ИСП РАН. — 2020. — т. 32, № 2. — с. 7—14.

60. *Zhao, Y.* Searching with Style: Authorship Attribution in Classic Literature / Y. Zhao, J. Zobel // Proceedings of the Thirtieth Australasian Conference on Computer Science - Volume 62. — Ballarat, Victoria, Australia : Australian Computer Society, Inc., 2007. — P. 59—68. — (ACSC '07).

61. *Luyckx, K.* Authorship Attribution and Verification with Many Authors and Limited Data / K. Luyckx, W. Daelemans // COLING 2008, 22nd International Conference on Computational Linguistics, Proceedings of the Conference, 18-22 August 2008, Manchester, UK / ed. by D. Scott, H. Uszkoreit. — 2008. — P. 513—520. — URL: https://aclanthology.org/C08-1065/.

62. *Argamon, S.* Style Mining of Electronic Messages for Multiple Authorship Discrimination: First Results / S. Argamon, M. Šarić, S. S. Stein //. — Washington, D.C. : Association for Computing Machinery, 2003. — P. 475—480. — (KDD '03). — URL: https://doi.org/10.1145/956750.956805.

63. *Stamatatos, E.* Plagiarism Detection Based on Structural Information / E. Stamatatos // Proceedings of the 20th ACM International Conference on Information and Knowledge Management. — Glasgow, Scotland, UK : Association for Computing Machinery, 2011. — P. 1221—1230. — (CIKM '11). — URL: https://doi.org/10.1145/2063576.2063754.

64. *Harris, Z. S.* Distributional Structure / Z. S. Harris // WORD. — 1954. — Vol. 10. — P. 146—162.

65. *Sahlgren, M.* The Distributional Hypothesis / M. Sahlgren // The Italian Journal of Linguistics. — 2008. — Vol. 20. — P. 33—54.

66. Distributed Representations of Words and Phrases and their Compositionality / T. Mikolov [et al.] // Advances in Neural Information Processing Systems. — 2013. — Oct. — Vol. 26.

67. Efficient Estimation of Word Representations in Vector Space / T. Mikolov [et al.] // Proceedings of Workshop at ICLR. — 2013. — Jan. — Vol. 2013.

68. *Pennington, J.* GloVe: Global Vectors for Word Representation / J. Pennington, R. Socher, C. D. Manning // EMNLP. — 2014.

69. Word Embedding Revisited: A New Representation Learning and Explicit Matrix Factorization Perspective / Y. Li [et al.] // IJCAI. — 2015.

70. *Линник, Ю.* Метод наименьших квадратов и основы теории обработки наблюдений / Ю. Линник. — М.: Физматлит, 1958.

71. Enriching Word Vectors with Subword Information / P. Bojanowski [et al.] // Transactions of the Association for Computational Linguistics. — 2017. — Vol. 5. — P. 135—146.

72. Bag of Tricks for Efficient Text Classification / A. Joulin [et al.] // Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. — Association for Computational Linguistics, 04/2017. — P. 427—431.

73. Deep Contextualized Word Representations / M. E. Peters [et al.] // Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). — New Orleans, Louisiana : Association for Computational Linguistics, 06/2018. — P. 2227—2237. — URL: https://aclanthology.org/N18-1202.

74. *Pan, S. J.* A Survey on Transfer Learning / S. J. Pan, Q. Yang // IEEE Transactions on Knowledge and Data Engineering. — 2010. — Vol. 22. — P. 1345—1359.

75. Handbook Of Research On Machine Learning Applications and Trends: Algorithms, Methods and Techniques - 2 Volumes / E. S. Olivas [et al.]. — Hershey, PA : Information Science Reference - Imprint of: IGI Publishing, 2009.

76. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / J. Devlin [et al.] // ArXiv. — 2019. — Vol. abs/1810.04805.

77. Attention is All You Need / A. Vaswani [et al.] //. — 2017. — URL: https://arxiv.org/pdf/1706.03762.pdf.

78. *Iyer, A.* Style Change Detection Using BERT / A. Iyer, S. Vosoughi // CLEF. — 2020.

79. *Ho, T. K.* Random decision forests / T. K. Ho // Proceedings of 3rd International Conference on Document Analysis and Recognition. Vol. 1. — 1995. — 278—282 vol.1.

80. *Zuo, C.* Style Change Detection with Feed-forward Neural Networks / C. Zuo, Y. Zhao, R. Banerjee // CLEF. — 2019.

81. *Deibel, R.* Style Change Detection on Real-World Data using an LSTM-powered Attribution Algorithm—Notebook for PAN at CLEF 2021 / R. Deibel, D. Löfflad // CLEF 2021 Labs and Workshops, Notebook Papers / ed. by G. Faggioli [et al.]. — CEUR-WS.org, 09/2021. — URL: http://ceur-ws.org/Vol-2936/paper-163.pdf.

82. *Hochreiter, S.* Long Short-term Memory / S. Hochreiter, J. Schmidhuber // Neural computation. — 1997. — Dec. — Vol. 9. — P. 1735—80.

83. *Nath, S.* Style change detection using Siamese neural networks (Notebook for PAN at CLEF 2021) / S. Nath //. — 09/2021.

84. *Chicco, D.* Siamese Neural Networks: An Overview / D. Chicco // Artificial Neural Networks - Third Edition. Vol. 2190 / ed. by H. M. Cartwright. — Springer, 2021. — P. 73—94. — (Methods in Molecular Biology). — URL: https://doi.org/10.1007/978-1-0716-0826-5%5C_3.

85. *Strøm, E.* Multi-label Style Change Detection by Solving a Binary Classification Problem—Notebook for PAN at CLEF 2021 / E. Strøm // CLEF 2021 Labs and Workshops, Notebook Papers / ed. by G. Faggioli [et al.]. — CEUR-WS.org, 09/2021. — URL: http://ceur-ws.org/Vol-2936/paper-191.pdf.

86. An Ensemble-Rich Multi-Aspect Approach Towards Robust Style Change Detection: Notebook for PAN at CLEF 2018 / D. Zlatkova [et al.] // CLEF. — 2018.

87. *Rokach, L.* Pattern Classification Using Ensemble Methods / L. Rokach. — USA : World Scientific Publishing Co., Inc., 2010.

88. *Schaetti, N.* Character-based Convolutional Neural Network and ResNet18 for Twitter Author Profiling: Notebook for PAN at CLEF 2018 / N. Schaetti // CLEF. — 2018.

89. *Bengio, Y.* Representation Learning: A Review and New Perspectives / Y. Bengio, A. Courville, P. Vincent // IEEE transactions on pattern analysis and machine intelligence. — 2013. — Aug. — Vol. 35. — P. 1798—1828.

90. *Rajaraman, A.* Mining of Massive Datasets / A. Rajaraman, J. Leskovec, J. Ullman. — 01/2014.

91. *Jones, K. S.* Idf term weighting and ir research lessons / K. S. Jones // Journal of Documentation. — 2004. — Vol. 60. — P. 521—523.

92. Scikit-learn: Machine Learning in Python / F. Pedregosa [et al.] // Journal of Machine Learning Research. — 2011. — Vol. 12. — P. 2825—2830.

93. *Pak, I.* Text Segmentation Techniques: A Critical Review / I. Pak, P. L. Teh //. — 2018.

94. *Osman, D.* Opinion Search in Web Logs. / D. Osman, J. Yearwood //. Vol. 63. — 03/2007. — P. 133—139.

95. *Flejter, D.* Unsupervised Methods of Topical Text Segmentation for Polish / D. Flejter, K. Wieloch, W. Abramowicz // ACL 2007. — 2007.

96. ClassStruggle: a clustering based text segmentation / S. Lamprier [et al.] //. — 01/2007. — P. 600—604.

97. Aspect-based sentence segmentation for sentiment summarization / J. Zhu [et al.]. — 2009. — Jan.

98. *Bahdanau, D.* Neural Machine Translation by Jointly Learning to Align and Translate / D. Bahdanau, K. Cho, Y. Bengio // ArXiv. — 2014. — Sept. — Vol. 1409.

99. Word segmentation of handwritten text using supervised classification techniques / Y. Sun [et al.] // Applied Soft Computing. — 2007. — Jan. — Vol. 7. — P. 71—88.

100. Comprehensive Information Based Semantic Orientation Identification / Yunwu [et al.] // 2007 International Conference on Natural Language Processing and Knowledge Engineering. — 2007. — P. 274—279.

101. *Ma, G.* Word Segmentation of Overlapping Ambiguous Strings During Chinese Reading / G. Ma, X. Li, K. Rayner // Journal of experimental psychology. Human perception and performance. — 2014. — Jan. — Vol. 40.

102. A new watershed model based system for character segmentation in degraded text lines / A. S. Kavitha [et al.] // Aeu-international Journal of Electronics and Communications. — 2017. — Vol. 71. — P. 45—52.

103. *Palmer, D. D.* Text Preprocessing / D. D. Palmer // Handbook of Natural Language Processing, Second Edition / ed. by N. Indurkhya, F. J. Damerau. — Chapman, Hall/CRC, 2010. — P. 9—30. — URL: http://www.crcnetbase.com/doi/abs/10.1201/9781420085938-c2.

104. *Bellman, R.* Dynamic Programming / R. Bellman. — Dover Publications, 1957.

105. *Yang, J.* Outlier Detection: How to Threshold Outlier Scores? / J. Yang, S. Rahardja, P. Fränti // Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing. — Sanya, China : Association for Computing Machinery, 2019. — (AIIPCC '19). — URL: https://doi.org/10.1145/3371425.3371427.

106. An Evaluation Framework for Plagiarism Detection. / M. Potthast [et al.] //. Vol. 2. — 01/2010. — P. 997—1005.

107. *Powers, D. M. W.* Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation / D. M. W. Powers // ArXiv. — 2020. — Vol. abs/2010.16061.

108. Approaches for Intrinsic and External Plagiarism Detection—Notebook for PAN at CLEF 2011 / G. Oberreuter [et al.] // Notebook Papers of CLEF 2011 Labs and Workshops, 19-22 September, Amsterdam, The Netherlands / ed. by V. Petras, P. Forner, P. Clough. — CEUR-WS.org, 09/2011. — URL: http://ceur-ws.org/Vol-1177.

109. *Kestemont, M.* Intrinsic Plagiarism Detection Using Character Trigram Distance Scores—Notebook for PAN at CLEF 2011 / M. Kestemont, K. Luyckx, W. Daelemans // Notebook Papers of CLEF 2011 Labs and Workshops, 19-22 September, Amsterdam, The Netherlands / ed. by V. Petras, P. Forner, P. Clough. — CEUR-WS.org, 09/2011. — URL: http://ceur-ws.org/Vol-1177.

110. *Hinton, G. E.* Unsupervised learning : foundations of neural computation / G. E. Hinton, T. J. Sejnowski //. — 1999.

111. Skip-Thought Vectors / R. Kiros [et al.] // arXiv preprint arXiv:1506.06726. — 2015.

112. *Sutskever, I.* Sequence to Sequence Learning with Neural Networks / I. Sutskever, O. Vinyals, Q. V. Le // NIPS. — 2014.

113. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation / K. Cho [et al.] // EMNLP. — 2014.

114. Comparative Study of CNN and RNN for Natural Language Processing / W. Yin [et al.] // ArXiv. — 2017. — Vol. abs/1702.01923.

115. *Gustineli, M.* A survey on recently proposed activation functions for Deep Learning / M. Gustineli. — 2022. — URL: https://arxiv.org/abs/2204.02921.

116. Growing Cosine Unit: A Novel Oscillatory Activation Function That Can Speedup Training and Reduce Parameters in Convolutional Neural Networks / M. M. Noel [et al.] // ArXiv. — 2021. — Vol. abs/2108.12943.

117. *Bird, S.* Natural language processing with Python: analyzing text with the natural language toolkit / S. Bird, E. Klein, E. Loper. — " O'Reilly Media, Inc.", 2009.

118. *Rahutomo, F.* Semantic Cosine Similarity / F. Rahutomo, T. Kitasuka, M. Aritsugi //. — 10/2012.

119. Overview of the Author Identification Task at PAN 2017: Style Breach Detection and Author Clustering / M. Tschuggnall [et al.] // Working Notes Papers of the CLEF 2017 Evaluation Labs. Vol. 1866 / ed. by L. Cappellato [et al.]. — 09/2017. — (CEUR Workshop Proceedings). — URL: http://ceur-ws.org/Vol-1866/.

120. *Khan, J.* Style Breach Detection: An Unsupervised Detection Model—Notebook for PAN at CLEF 2017 / J. Khan // CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers, 11-14 September, Dublin, Ireland / ed. by L. Cappellato [et al.]. — CEUR-WS.org, 09/2017. — URL: http://ceur-ws.org/Vol-1866/.

121. *Karaś, D.* OPI-JSA at CLEF 2017: Author Clustering and Style Breach Detection—Notebook for PAN at CLEF 2017 / D. Karaś, M. Śpiewak, P. Sobecki // CLEF 2017 Evaluation Labs and Workshop – Working Notes Papers, 11-14 September, Dublin, Ireland / ed. by L. Cappellato [et al.]. — CEUR-WS.org, 09/2017. — URL: http://ceur-ws.org/Vol-1866/.

122. *I.V.Sochenkov.* The paraplag: russian dataset for paraphrased plagiarism detection / I.V.Sochenkov, D. Zubarev, I. Smirnov // Computational Linguistics and Intellectual Technologies. — 2017. — Vol. Papers from the Annual International Conference “Dialogue” 2017.

123. PAN20 Authorship Analysis: Style Change Detection / E. Zangerle [et al.]. — 02/2020. — URL: https://doi.org/10.5281/zenodo.3660984.