

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

На правах рукописи



Мансур Али Махмуд

**МОДЕЛЬ, МЕТОД И АЛГОРИТМЫ DATA MINING ДЛЯ
ИНТЕЛЛЕКТУАЛЬНОЙ ОБРАБОТКИ И АНАЛИЗА
ТЕКСТОВ НА ЕСТЕСТВЕННОМ ЯЗЫКЕ**

Специальность – 1.2.1. Искусственный интеллект и машинное обучение
(технические науки)

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель –
Кравченко Юрий Алексеевич
доктор технических наук, доцент

Таганрог – 2025

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ.....	4
ГЛАВА 1. АНАЛИТИЧЕСКИЙ ОБЗОР СОВРЕМЕННЫХ МЕТОДОВ АНАЛИЗА И ПРЕДСТАВЛЕНИЯ ТЕКСТОВЫХ ДОКУМЕНТОВ	15
1.1. Методы интеллектуального анализа текстовых данных.....	15
1.1.1. Понятие Data Mining.....	15
1.1.2. Аналитический обзор задач и методов Data mining	17
1.1.3. Этапы процесса интеллектуального анализа данных.....	24
1.1.4. Интеллектуальный анализ текста.....	25
1.2. Обзор методов представления текста для машинной обработки.....	29
1.2.1. Методы представления текста на основе векторной модели	29
1.2.2. Методы тематического моделирования.....	31
1.2.3. Методы векторизации текстов на основе методов встраивания слов ...	34
1.2.4. Методы векторизации текстов на основе построения концептов.....	37
1.3. Общее сравнение методов представления текста.....	43
1.4. Постановка задачи исследования	45
1.5. Выводы по разделу.....	49
ГЛАВА 2. МОДЕЛЬ И МОДИФИЦИРОВАННЫЙ МЕТОД ВЕКТОРИЗАЦИЯ ТЕКСТА НА ОСНОВЕ МЕТОДОВ DATA MINING	53
2.1. Разработка метода векторизации текста на основе методов Data mining	53
2.1.1. Функциональная схема предлагаемого метода.....	54
2.1.2 Применение фильтрации терминов на этапе построения словаря эталонных концептов	63
2.2. Временная сложность разработанного метода <i>BoWC</i>	66
2.3. Выводы по разделу.....	67
ГЛАВА 3. АЛГОРИТМ ПОСТРОЕНИЯ КОНЦЕПТОВ ПРИ РЕШЕНИИ ЗАДАЧИ КЛАСТЕРИЗАЦИИ С ИСПОЛЬЗОВАНИЕМ КЛЮЧЕВЫХ ФРАЗ	69

3.1. Решение задачи многозначности слова в словаре эталонных концептов	69
3.2. Алгоритм извлечения ключевых фраз на основе применения парсера.....	71
3.3. Алгоритм построения концептов на основе извлечения ключевых фраз	74
3.4. Модификация функции взвешивания концептов метода BoWC	77
3.5. Выводы по разделу.....	80
ГЛАВА 4. РАЗРАБОТКА ПРОГРАММНОГО ПРИЛОЖЕНИЯ И ПРОВЕДЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ.....	81
4.1. Разработка компонентной архитектуры программного Веб-приложения...	81
4.2. Задание настроек вычислительных экспериментов	85
4.3. Результаты эксперимента по анализу эффективности метода <i>BoWC</i> и настройка его параметров	92
4.4. Результаты эксперимента по анализу эффективности метода <i>BoWC</i> с применением фильтрации терминов.....	102
4.5. Результаты эксперимента по использованию словаря эталонных концептов на основе ключевых фраз	106
4.6. Анализ эффективности алгоритма извлечения ключевых фраз на основе парсера.....	112
4.7. Анализ результатов вычислительного эксперимента по исследованию характеристик интерпретируемости векторов	120
4.8. Выводы по разделу.....	128
ЗАКЛЮЧЕНИЕ	130
СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ	134
СПИСОК ЛИТЕРАТУРЫ	135
ПРИЛОЖЕНИЕ №1.....	151
ПРИЛОЖЕНИЕ №2.....	155

ВВЕДЕНИЕ

Актуальность темы исследования и степень её разработанности. В эпоху цифрового прогресса и массового применения технологии искусственного интеллекта (*ИИ*) экспоненциальный рост объёмов текстовых данных представляет собой серьезную проблему. Значительный объем неструктурированных текстов на естественном языке, генерируемых в различных областях, требует создания эффективных методов их обработки и анализа для извлечения действенных закономерностей, позволяющих повысить эффективность алгоритмов машинного обучения в данной области.

Проблема необходимости повышения эффективности процессов обработки и анализа текстов на естественном языке подчеркивает значимость методов *Data Mining* при решении задач классификации и кластеризации для структурирования текстовых данных [1, 2]. При решении задачи классификации (обучение с учителем) данные распределяются в предопределенные классы на основе учёта заданных признаков.

Кластеризация, напротив, является задачей обучения без учителя и позволяет распределить документы на основе оценки сходства признаков по кластерам, количество которых изначально неизвестно. Эта задача имеет решающее значение для выявления внутренних структур в данных, таких как определение шаблонов в словах и фразах, встречающихся в документах, и группировку похожих слов по темам для создания концептуальных кластеров [3]. Ее эффективность оценивается путём измерения однородности кластеров (англ. *Clusters Homogeneity*), которая относится к степени схожести между членами конкретного кластера в выборке [4].

Векторное представление (векторизация текстов) является одной из основных моделей пространства решений при классификации и кластеризации текстовых документов в системах искусственного интеллекта. Методы векторизации текстов включают в себя множество информационных процессов, которые облегчают преобразование текстов в цифровые векторы [5, 6]. Векторное преобразование

документов позволяет использовать различные математические операции и алгоритмы машинного обучения для выявления закономерностей, связей и тенденций в них, тем самым способствуя развитию приложений искусственного интеллекта в различных областях науки и техники [7, 8].

В данном исследовании основное внимание уделяется двум основным свойствам, которые влияют на дискриминационную способность векторов признаков документов, а именно размерности и интерпретируемости [9]. Векторы с высокой размерностью включают в себя большее количество признаков, что повышает эффективность интеллектуального анализа текстов, но приводит к нехватке вычислительных ресурсов, занимает больше памяти и отрицательно влияет на масштабируемость. Интерпретируемость векторов признаков позволяет обнаружить ошибки, что увеличивает доверие пользователей к таким моделям, так как их применение повышает качество результатов классификации и кластеризации текстовых документов.

Традиционные модели представления текстовых документов, такие как «мешка слов» (BoW, Bag-of-Words), и TF-IDF (term frequency invers document frequency), достигли определённых результатов во многих задачах классификации и кластеризации документов благодаря своей простоте, эффективности и сравнительно высокой точности [10, 11]. Кроме того, векторы, построенные на основе этих моделей, являются интерпретируемыми. Тем не менее, у этих моделей есть два основных недостатка. Во-первых, обычно в них присутствует разреженность данных и высокая размерность [12]. Во-вторых, эти модели не учитывают семантические отношения между словами. Эти недостатки ограничивают способность моделей интеллектуального анализа текста фиксировать истинное сходство между документами [13, 14].

В отличие от этого, методы векторизации текстов на основе нейронных сетей, такие как встраивание слов (англ. word embedding) [15, 16] и встраивание документов (англ. document embedding) [17, 18], считаются мощными средствами представления семантических отношений между словами и документами в пространстве высокой размерности [19–21]. Эти методы обладают способностью

обнаруживать скрытые закономерности и гибкостью в обработке исходных текстов, создавая плотные и низкоразмерные представления для слов, предложений и документов. Однако, использование их для кодирования целых документов не является достаточно эффективным, так как требуется либо усреднение векторов слов внутри документа, либо их конкатенирование или суммирование [22, 12, 23]. Процесс конкатенирования векторов не решает проблему большой размерности, а процессы усреднения и суммирования игнорируют часть информации, содержащейся в документе. В свете всего этого векторы, построенные этими методами, неинтерпретируемые [24].

В последнее время появились методы на основе трансформеров, которые создают контекстные векторные представления и достигают лучших результатов в задачах классификации, кластеризации, сходства текста и поиска [25, 26]. Однако их эффективность при решении этих задач с относительно длинными документами невысока. Это связано с их ограниченными возможностями обработки длинных документов, что требует либо усечения текста и, следовательно, потери информации [27], либо необходимости модифицировать модель [26, 28]. В этом случае модель допускает более длинные текстовые последовательности, но с повышенными вычислительными затратами.

Таким образом, существующие методы векторизации текстов не позволяют обеспечить семантические представления документов (векторов) с малыми размерностями, которые можно интерпретировать без негативного влияния на эффективность алгоритмов классификации и кластеризации.

В данном исследовании применено комплексное решение для построения низкоразмерных, *интерпретируемых* векторных представлений текстов на основе концептов [29, 3]. *Концепт* – это набор слов или фраз, имеющих общее семантическое значение. В этом подходе для представления текста вместо слов используются концепты, где каждый элемент вектора соответствует одному концепту, что позволяет снизить размерность пространства решений и сохранить интерпретируемость. Это требует разработки эффективных методов и алгоритмов обработки и анализа текстов на естественном языке, которые способны построить

концепты и определить их соответствие анализируемым документам, таким образом, чтобы сохранить дискриминационную способность полученных векторов.

Для реализации предложенных автором модели, метода и алгоритмов в данном исследовании разработано программное приложение, позволяющее автоматизировать процесс обработки и анализа текстов в условиях большой размерности и увеличить информационный объем качественно структурированного текста в информационном пространстве [30, 31]. Разработанное приложение принимает на вход набор текстов из различных предметных областей, например, таких как корпоративные данные, и затем обрабатывает и анализирует эти тексты для извлечения ключевых фраз и определения основных концептов, а также классифицирует их по категориям в соответствии с деловой активностью. Приложение также подчеркивает значимость характеристики интерпретируемости векторов, созданных предложенным методом при поиске целевой аудитории в рекомендательной системе, реализующей технологию «*Look-a-like*». Технология *Look-a-like* – это инструменты для поиска объектов (людей, компаний, устройств и др.), схожих с существующей целевой аудиторией по характеристикам, поведению или интересам, чтобы расширить охват и повысить эффективность маркетинговых действий [32]. Это существенно улучшает пользовательский опыт и повышает эффективность доступа к релевантной информации необходимой для принятия решений, а также обеспечивает фильтрацию нежелательного контента.

Таким образом, **актуальной научной задачей** для развития отрасли искусственного интеллекта и машинного обучения является разработка *моделей, методов и алгоритмов Data Mining для интеллектуальной обработки и анализа текстов на естественном языке*, позволяющих снизить частоту ошибок при классификации и кластеризации текстов.

Ряд работ посвящен развитию методов анализа текста и методов представления текста для решения задач классификации и кластеризации. Источниками для исследования диссертации послужили работы отечественных и

зарубежных ученых по основам текстового анализа, взвешивания терминов и извлечения информации: Р. Муни [33], Х. Шютце и К. Д. Мэннинг [34], К. С. Джонс, М. А. Хёрст [35], А. Панченко [36], И. Д. Иванович [37–40] и А. Кутузов [41]. Работы Н. Красвелл и Б. Митра [42, 19–21], Д. М. Блей [43], Д. Уэстон, Д. Юрафски [44], П. Сердюков и И. Титов [45] вносят значительный вклад в представление текста и векторизацию, и моделей семантического поиска.

Целью диссертационной работы является повышение эффективности моделей, методов и алгоритмов классификации и кластеризации текстов. Под эффективностью понимается минимизация частоты ошибок классификации и кластеризации текстов при условии снижения размерности векторного пространства признаков с сохранением его интерпретируемости.

Для достижения поставленной цели были решены следующие основные задачи:

1. Проведён аналитический обзор современных методов Data mining и методов векторизации текстов на естественном языке.
2. Построена модель векторизации текстов с использованием алгоритмов извлечения ключевых фраз и алгоритмов кластеризации.
3. Разработан модифицированный метод генерации векторных представлений документов на основе алгоритмов обработки и анализа текстов в системах искусственного интеллекта.
4. Разработан алгоритм извлечения и фильтрации ключевых фраз на основе парсера.
5. Разработан алгоритм построения концептов на основе алгоритмов извлечения ключевых фраз и кластеризации.
6. Разработан программное приложение для проведения вычислительного эксперимента и подтверждения достоверности и эффективности полученных основных результатов.

Объект исследования – тексты на естественном языке.

Предмет исследования – модели, методы и алгоритмы обработки и анализа текстов для решения задач классификации и кластеризации текстовых документов.

Методология и методы диссертационного исследования. При выполнении диссертационной работы использовались методы интеллектуального анализа данных, методы обработки и анализа текстов на естественном языке, методы системного анализа, теории информационных систем, формальной логики, методы искусственного интеллекта и машинного обучения, а также методы объектно-ориентированного программирования.

Тематика работы соответствует п. 4 «Разработка методов, алгоритмов и создание систем искусственного интеллекта и машинного обучения для обработки и анализа текстов на естественном языке, для изображений, речи, биомедицины и других специальных видов данных», п. 5 «Методы и технологии поиска, приобретения и использования знаний и закономерностей, в том числе – эмпирических, в системах искусственного интеллекта. Исследования в области совместного применения методов машинного обучения и классического математического моделирования. Методы и средства использования экспертиз знаний» паспорта специальности 1.2.1. Искусственный интеллект и машинное обучение (технические науки).

Научная новизна и соответствие научной специальности:

1. Построена математическая модель векторизации текстов на основе концептов, **отличающаяся** применением новых правил построения эталонных концептов и новых функций определения их весов, **позволяющая** снизить размерность векторного пространства и улучшить дискриминационную способность результирующих векторов признаков (пункт 4 паспорта специальности 1.2.1, страницы 57-64 диссертации).

2. Разработан модифицированный метод генерации векторных представлений документов на основе построенной модели векторизации, **отличающийся** применением интерпретируемых признаков при векторизации, **позволяющий** снизить частоту ошибок алгоритмов классификации и кластеризации текстовых документов (пункт 4 паспорта специальности 1.2.1, страницы 53-67 диссертации).

3. Разработан алгоритм извлечения и фильтрации ключевых фраз на основе частоты их появления, **отличающийся** применением функции парсера для

разметки частей речи, что **позволяет** извлекать ключевые фразы с правильной грамматической структурой (пункт 5 паспорта специальности 1.2.1, страницы 69-74 диссертации).

4. Разработан алгоритм построения концептов из семантически близких фраз, **отличающийся** решением задачи кластеризации фраз с учетом контекстуальной семантической близости, что **позволяет** повысить однородность кластеров, представляющих концепты (пункт 5 паспорта специальности 1.2.1, страницы 74-80 диссертации).

Теоретическая значимость работы. Полученные научные результаты развивают аппарат искусственного интеллекта и машинного обучения в области решения важной научной проблемы увеличения информационного объема семантически обработанных текстов в информационном пространстве; разработка методов и алгоритмов машинного обучения для обработки и анализа текстов на естественном языке, в том числе, методов векторизации, классификации и кластеризации текстов; исследования и разработки средств представления текстов.

Практическая значимость работы заключается в создании программного приложения, позволяющего использовать разработанные модель, метод и алгоритмы обработки и анализа текстов на естественном языке в системах искусственного интеллекта для минимизации частоты появления ошибок при решении задач классификации и кластеризации с учётом условий снижения размерности векторного пространства и сохранения его интерпретируемости.

Положения, выносимые на защиту:

1. Математическая модель векторизации текстов на основе применения новых правил построения эталонных концептов и новых функций определения их весов **позволяет** снизить размерность векторного пространства и улучшить дискриминационную способность результирующих векторов признаков;

2. Модифицированный метод генерации векторных представлений текстов на основе построенной модели векторизации **позволяет** снизить частоту ошибок алгоритмов классификации и кластеризации текстовых документов;

3. Алгоритм извлечения и фильтрации ключевых фраз на основе применения функций парсера для разметки частей речи **позволяет** извлекать ключевые фразы с правильной грамматической структурой;

4. Алгоритм построения концептов из семантически близких фраз **позволяет** повысить однородность кластеров, представляющих концепты.

Степень достоверности результатов. Достоверность научных результатов работы подтверждается непротиворечивостью и согласованностью с известными фактами и исследованиями в рассматриваемой области, высокой степенью сходимости теоретических результатов с данными экспериментов, и определяется применением теоретических и методологических основ разработок ведущих ученых в области создания интеллектуальных систем, корректным и обоснованным использованием математического аппарата, экспериментальными исследованиями разработанных моделей и методов.

Личный вклад автора. Все выносимые на защиту результаты и положения, составляющие основное содержание диссертационной работы, разработаны и получены лично автором или при его непосредственном участии. В работах, опубликованных в соавторстве, соискателю принадлежит определяющая роль в развитии информационных процессов моделей и методов обработки и анализа текстов на естественном языке.

Реализация и внедрение результатов работы. Теоретические и практические результаты работы внедрены в информационные процессы ИТ-компании ООО «Ит-Эффект» (г. Москва). Полученные в работе научные результаты позволили повысить эффективность решения задач классификации, кластеризации и извлечения ключевых фраз в рекомендательной системе, реализующей технологию «look-alike» (поиск целевой аудитории для эффективного масштабирования деловой активности предприятия). Результаты работы также используются в учебном процессе института компьютерных технологий и информационной безопасности Южного федерального университета.

Апробация результатов диссертации. Основные положения и отдельные результаты исследования докладывались и обсуждались на следующих

конференциях: VI International Conference on Information Technologies in Engineering Education (Inforino 2022), (Россия, Москва, апрель 2022); VI Всероссийская научно-техническая конференция «Фундаментальные и прикладные аспекты компьютерных технологий и информационной безопасности», (Россия, Таганрог 2020); XVIII, XIX и XX Всероссийская конференция молодых ученых аспирантов и студентов «Информационные технологии, системный анализ и управление ИТСАУ» (Россия, Таганрог, 2019-2022); II научно-методическая конференция НПР «Современные компьютерные технологии» (Россия, Таганрог, 2021-2022); XII международная научно-техническая конференция «технологии разработки информационных систем (ТРИС-2022)» (Россия, Таганрог 2022); «5th International Scientific Convention UCIENCIA» (Куба, сентябрь 2023); International Russian Automation Conference RusAutoCon, (Россия, Сочи, 2023).

Публикации. По теме диссертации опубликовано **17** научных работ, из которых: **3** статьи опубликованы в издании из перечня рекомендованных ВАК (К2), в т.ч. **1** статья опубликована без соавторов; **2** статьи – в изданиях из международных баз данных Scopus и/или Web of Science. Получены **2** свидетельства об государственной регистрации программ для ЭВМ. В трудах всероссийских и международных конгрессов и конференций опубликовано **9** работ.

Структура и объем работы. Диссертация состоит из введения, **4** разделов, заключения, списка литературы, содержащего **146** наименований, и **2** приложений. Основная часть работы содержит **150** страниц, включая **31** рисунок и **12** таблиц.

Во введении сформулирована цель работы, обоснована актуальность темы диссертации, описаны основные научные положения, выносимые на защиту, научная новизна, теоретическая и практическая ценность, апробация диссертационной работы, реализация и внедрение, а также структура диссертации.

В первой главе представлен аналитический обзор научных исследований в области обработки и анализа текстов на основе методов искусственного интеллекта и машинного обучения. Также рассмотрены существующие методы представления текста. Основное внимание уделено методам векторизации текстов.

Проанализированы публикации, напрямую связанные с темой диссертации. Даны формализованные постановки основных задач исследования и сделаны выводы по разделу в целом.

Во второй главе решается задача генерации векторных представления текстов. Разработаны модель и метод векторизации документов с использованием технологии, основанные на методах Data mining для построения словаря эталонных концептов, а также на методах оценки семантической близости для сопоставления слов документа с эталонными концептами и построения векторов. Метод создает семантические признаки, характеризующие каждый документ, позволяя получать низкоразмерные, более информативные векторные представления, способные эффективно решать задачи классификации и кластеризации документов любой длины без необходимости вырезания или потери какой-либо части важной информации документа. В главе описаны рабочие этапы метода, его математическая модель, отличия и преимущества по сравнению с другими методами.

Третья глава диссертации посвящена описанию разработке двух алгоритмов. Первый – это алгоритм построения концептов на основе ключевых фраз (n -грамм) вместо униграмм. Принцип работы алгоритма основан на использовании ключевых фраз для решения проблемы неоднозначности слова в словаре эталонных концептов. Второй алгоритм – это алгоритм извлечения и фильтрации ключевых фраз на основе применения функций парсера для разметки частей речи. Приведено описание функций алгоритма, используемых при оценке и определении весов релевантных ключевых фраз. Алгоритм позволяет отфильтровать ключевые фразы с неправильной грамматической структурой. Применение этого алгоритма при построении словаря эталонных концептов способствует уменьшению зашумленности концептов и повышению их однородности.

В четвертой главе рассмотрен практический пример использования результатов работы. Описана разработка программного приложения и проведения серии вычислительных экспериментов для сравнения качества и эффективности разработанных модели, алгоритмов и метода. В среднем предложенный автором

метод векторизации текстов снижает частоту ошибок алгоритмов классификации и кластеризации документов на 0,2 – 1 % и 2 – 6 % соответственно, а также позволяет уменьшить число признаков по сравнению с конкурирующими методами. Временная сложность предложенного метода векторизации текстов в худшем случае составляет $O(n^2)$. Разработанные алгоритмы и метод превосходят большинство базовых методов по минимальному количеству признаков и максимальной точности классификации и кластеризации документов.

В заключении изложены итоги выполненного исследования, рекомендации, перспективы дальнейшей разработки темы.

В приложениях приведены свидетельства об официальной регистрации программ для ЭВМ и копии актов внедрения.

ГЛАВА 1. АНАЛИТИЧЕСКИЙ ОБЗОР СОВРЕМЕННЫХ МЕТОДОВ АНАЛИЗА И ПРЕДСТАВЛЕНИЯ ТЕКСТОВЫХ ДОКУМЕНТОВ

В данной главе выполняется аналитический обзор научных исследований в области представления и анализа текстовых документов на основе современных методов интеллектуального анализа данных. Глава начинается с описания основных задач и методов анализа данных, что является основой для последующего исследования. Анализ текста (англ. Text mining), как подраздел анализа данных, имеет огромный потенциал для извлечения ценной информации. Учитывая, что текст является одной из самых распространенных и часто встречающихся форм данных, понимание эффективного анализа и извлечения информации из него является критически важным. Одной из ключевых проблем в текстовом анализе является представление текстовых данных в формате, который может быть обработан алгоритмами машинного обучения. С этой целью обсуждаются различные подходы к представлению текста, называемые векторизацией текста, включая модель векторного пространства, семантическое представление и языковые модели [46]. Предоставляя сравнительный анализ этих методов, автор раскрывает их преимущества, недостатки и пригодность для различных задач анализа текста. В конце главы дана постановка задачи и выбраны основные направления исследований.

1.1. Методы интеллектуального анализа текстовых данных

1.1.1. Понятие Data Mining

Технология «Data mining» является междисциплинарной областью исследований, возникшей и развивающейся на базе теорий вероятности и прикладной статистики, теорий информации, распознавания образов, искусственного интеллекта, теорий баз данных, хранилищ данных, высокопроизводительных вычислений, визуализации данных машинного обучения и др [47, 48].

Понятие «Data Mining», появившееся в 1978 году, приобрело высокую популярность в современной трактовке примерно с первой половины 1990-х годов. Термин «Data Mining» часто переводится как добыча данных, извлечение информации, раскопка данных, интеллектуальный анализ данных, средства поиска закономерностей между информационными элементами, извлечение знаний, анализ шаблонов и другие менее распространённые синонимы [49].

Поскольку Data Mining междисциплинарная область, для этого термина было дано несколько определений, включая следующее: Data Mining – это процесс обнаружения в сырых данных ранее неизвестных, нетривиальных, практически полезных и доступных интерпретаций знаний, необходимых для принятия решений в различных сферах человеческой деятельности. Также определяется как «процесс выделения из данных неявной и неструктурированной информации и представления ее в виде, пригодном для использования» [47, 50].

Стоит отметить, что термин «Data mining» является неправильным, поскольку целью является извлечение шаблонов и знаний из больших объемов данных, а не извлечение (добыча) самих данных. Чтобы определение интеллектуального анализа данных было четким, необходимо провести различие между следующими концепциями, которые хотя и используются в качестве синонимов, но между ними существуют принципиальные различия. Термин «данные» происходит от слова data – факт, а информация (information) означает разъяснение, изложение, т.е. сведения или сообщение. Согласно [51], данные – это набор оперативных и объективных фактов, описывающих объекты, события, явления, процессы и т.д. Информация – это выделенная, упорядоченная и обработанная в соответствии с контекстом часть данных, наделенная системой отношений между данными и определенным смыслом (данные о данных, или данные плюс метаданные). Знание – это сложная сетевая иерархия элементов информации с выявленными зависимостями и/или существенными связями между фактами, событиями, явлениями и процессами и т.д [52].

Следовательно, суть и цель технологии Data Mining можно охарактеризовать так: это технология, которая предназначена для поиска в больших объемах данных

неочевидных, объективных и полезных на практике закономерностей. Неочевидных – это значит, что найденные закономерности не обнаруживаются стандартными методами обработки информации или экспертным путем. Объективных – это значит, что обнаруженные закономерности будут полностью соответствовать действительности, в отличие от экспертного мнения, которое всегда является субъективным. Практически полезных – это значит, что выводы имеют конкретное значение, которому можно найти практическое применение. Знания – совокупность информации, которая образует целостное описание, соответствующее некоторому уровню осведомленности об описываемом вопросе, предмете, проблеме и т.д.

В данном пункте автор привел определения основных терминов необходимых для понимания предметной области диссертационного исследования. В следующем пункте проведён аналитический обзор основных задач и методов технологии Data mining.

1.1.2. Аналитический обзор задач и методов Data mining

К задачам интеллектуального анализа данных относятся описательные и прогностические задачи.

Прогностические задачи делают предсказание неизвестных значений данных с использованием известных значений. К ним относятся прогнозирование, классификация, регрессия и т.д. Описательные задачи идентифицируют шаблоны или отношения в данных и исследуют свойства данных [53]. К ним относятся кластеризация, правило ассоциации, обнаружение последовательности и т.д. Ниже рассмотрены наиболее распространенные задачи Data mining, при этом большое внимание уделяется задачам классификации и кластеризации, поскольку они будут использованы в данном исследовании для анализа текста.

Классификация (англ. Classification) – это деривация модели, которая определяет класс объекта на основе его атрибутов [54, 47]. Набор объектов задается как обучающий набор, в котором каждый объект представлен вектором атрибутов вместе со своим классом. Анализируя связь между атрибутами и классом объектов

в обучающем наборе, можно построить классификационную модель. Эта задача является примером контролируемого обучения, потому что классы предопределены до изучения целевых данных. В результате решения задачи классификации обнаруживаются признаки, которые характеризуют группы объектов исследуемого набора данных – классы; по этим признакам новый объект можно отнести к тому или иному классу.

Методы классификации многочисленны и разнообразны, наиболее известными из них являются: k-ближайших соседей (англ. k-nearest neighbor), деревья решений, генетические алгоритмы, метод опорных векторов (SVM) и нейронные сети [55, 56]. Ниже подробно описаны классификаторы, которые имеют отношение к методам и алгоритмам, разработанным в этом исследовании.

Метод опорных векторов (англ. Support Vector Machine SVM) использует процедуру обучения с учителем для задач классификации и регрессии [56, 57]. Он работает путем определения лучшей гиперплоскости для разделения набора данных, где гиперплоскость представляет собой подпространство, разделяющее данные на различные классы. SVM популярен из-за своей высокой точности, скорости вычислений и способности решать нелинейные задачи с помощью применения функции ядра (англ. Kernel), которая отображает данные из одного пространства в другое. Данная функция позволяет методу работать в пространстве признаков большей размерности.

В SVM используются различные типы функций ядер, включая линейные, нелинейные, полиномиальные (poly), радиально-базисные функции (radial basis function RBF) и сигмоидные ядра. Эти функции позволяют SVM решать более широкий спектр задач, включая те, у которых отсутствует линейная разделимость классов.

Опорные векторы – это точки, находящиеся ближе всего к гиперплоскости. Гиперплоскость – это подпространство размерностью $n-1$ от основного, используемое для разделения пространства на несколько секций.

Метод SVM стремится найти лучшую границу принятия решений для разделения двух классов с наивысшей обобщающей способностью [57]. Он

определяет оптимальность путем максимизации зазора, который представляет собой расстояние между точками данных и границей принятия решений. Ширина зазора имеет ключевое значение для SVM, поскольку он ищет более широкий зазор для достижения лучшей обобщающей способности. На рисунке 1.1 показаны два класса, разделенные гиперплоскостью с максимальным зазором.

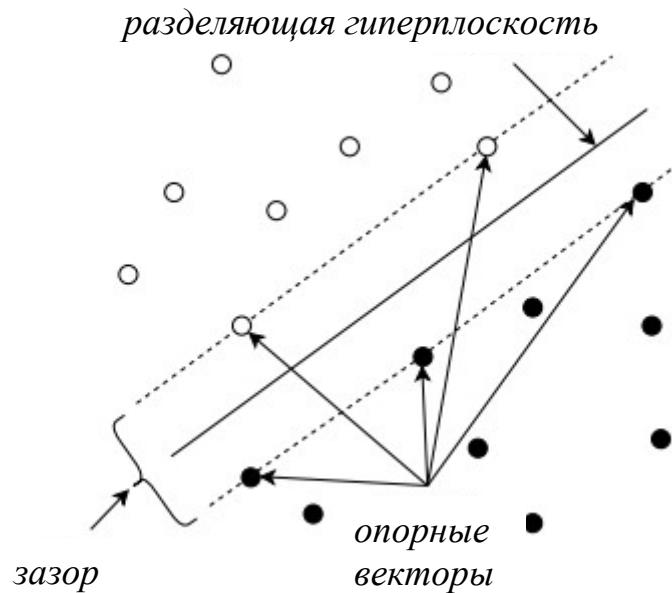


Рисунок 1.1 – Два (линейно разделяемых) класса, разделенные гиперплоскостью с максимальным зазором относительно опорных векторов

Задача оптимизации для нахождения разделяющей гиперплоскости включает в себя минимизацию $\|w\|$ при условиях $y_i(w \cdot x_i - b) \geq 1$ для всех примеров обучения (x_i, y_i) , где w – весовой вектор гиперплоскости, b – термин смещения, а y_i – метка класса обучающего примера.

Вектор w находится путем решения задачи оптимизации с использованием стандартных методов квадратичного программирования, таких как последовательная минимизация оптимизации или стохастический градиентный спуск. Его можно записать в следующей форме с использованием множителей Лагранжа α :

$$\arg \min_{w,b} \max_{\alpha \geq 0} \left\{ \frac{1}{2} \|w\|^2 - \sum_{i=1}^n \alpha_i [y_i(\langle w, x_i \rangle - b) - 1] \right\}. \quad (1)$$

Одним из классификаторов для проведения вычислительных экспериментов в данном исследовании выбран метод опорных векторов, так как он показал лучшие результаты в задаче классификации текста по сравнению с другими методами.

Искусственные нейронные сети (ИНС) – это тип алгоритмов машинного обучения, вдохновленных структурой и функцией человеческого мозга. Они состоят из взаимосвязанных узлов (нейронов), организованных в слои. Вся логика работы ИНС заложена в связях между нейронами [53].

Искусственные нейроны являются фундаментальными строительными блоками ИНС. Каждый нейрон получает входные сигналы от других нейронов или внешних источников, обрабатывает их с помощью функции активации и производит выходной сигнал, который передается связанным нейронам. Функции активации вводят нелинейность в ИНС, позволяя им моделировать сложные взаимосвязи между входными и выходными данными. Распространенные функции активации включают сигмоиду, \tanh и ReLU [20].

Веса и смещения – это числовые значения, связанные с соединениями между нейронами. Веса определяют силу сигнала, передаваемого от одного нейрона к другому в то время, как смещения корректируют общий уровень активации нейрона. ИНС организованы в слои, причем каждый слой состоит из группы взаимосвязанных нейронов. Типичная архитектура ИНС включает входной слой, один или несколько скрытых слоев и выходной слой. Выход каждого нейрона определяется взвешенной суммой его входов, за которой следует применение функции активации. Этот процесс можно математически представить следующим образом:

$$z = \sum_{i=1}^n (w_i \cdot x_i) + b, \quad (2)$$

где z – взвешенная сумма входных данных и смещений, w_i – веса, связанные с входными данными, x_i – входные значения, b – смещение.

Затем выходные данные нейрона проходят через функцию активации, такую как сигмовидная функция, \tanh или $ReLU$, чтобы внести нелинейность в сеть. Выход нейрона “ a ” определяется выражением:

$$a = f(z). \quad (3)$$

ИНС обучаются с использованием набора размеченных примеров, где каждый пример состоит из входного образца и соответствующего желаемого выхода. Процесс обучения включает в себя прямое распространение, вычисление ошибки, обратное распространение и оптимизацию с использованием алгоритмов, таких как градиентный спуск.

Искусственные нейронные сети нашли широкое применение в различных областях, включая распознавание изображений, прогностическое моделирование и обработку естественного языка (NLP), такие как машинный перевод, анализ настроений и суммирование текста.

Нейросетевой классификатор будет использоваться в вычислительных экспериментах для оценки эффективности работы с векторами, полученными разработанными методами (см. пункте 4.1).

Алгоритм k-ближайших соседей (k-NN) – это непараметрический метод обучения с учителем, используемый для задач классификации и регрессии. Он основан на предположении, что похожие элементы расположены близко друг к другу в пространстве признаков [58].

В случае классификации класс новой точки данных определяется большинством голосов ее k ближайших соседей. Математически это можно представить следующим образом:

$$\hat{y} = \text{majority vote}(\{y_i\}_{i=1}^k), \quad (4)$$

где \hat{y} – предполагаемый класс новой точки данных, y_i – классы k ближайших соседей. В случае регрессии значение новой точки данных оценивается на основе среднего значения ее k ближайших соседей. Математически это можно представить следующим образом:

$$\hat{y} = \frac{1}{k} \sum_{i=1}^k y_i, \quad (5)$$

где \hat{y} – расчетное значение для новой точки данных, y_i – значения к ближайших соседей.

Выбор значения k имеет решающее значение, поскольку он существенно влияет на производительность алгоритма. Для измерения сходства между точками данных можно использовать различные метрики расстояния, такие как евклидово расстояние, манхэттенское расстояние, расстояние минковского и расстояние Хэмминга [58].

Алгоритм k-NN требует значительных ресурсов для больших обучающих выборок при вычислении расстояний от тестового вектора до всех сохраненных примеров. Однако это не является препятствием для проверки данного классификатора в экспериментах для оценки эффективности разработанных методов.

Кластеризация (англ. Clustering) – похожа на классификацию, за исключением того, что группы не предопределены, но определяются путем изучения поведения данных (неконтролируемый обучение). Результатом кластеризации является разбиение объектов на группы. Примеров методов кластеризации много, в том числе: К-средние, DBSCAN, иерархическая кластеризация, спектральная кластеризация и другие¹ [59, 60, 47]. Ниже приводится подробное объяснение алгоритма К-средних и его рабочего механизма, поскольку он будет использоваться для построения и оценки эффективности предложенных в данном исследовании методов анализа текста.

Алгоритм кластеризации К-средних – популярный метод машинного обучения без учителя, используемый для группировки неразмеченных наборов данных в различные кластеры [61]. Это алгоритм, основанный на центроидах, который стремится разделить n наблюдений на k ($\leq n$) наборов $S = \{S_1, S_2, \dots, S_k\}$, чтобы минимизировать сумму квадратов внутрикластерных расстояний (т.е. дисперсию). Формально цель можно представить следующим выражением:

¹ Более подробная информация доступна по ссылке [2.3. Clustering — scikit-learn 1.3.2 documentation](#)

$$\arg \min_S \sum_{i=1}^k \sum_{x \in S_i} \|x - \mu_i\|^2, \quad (6)$$

где S – набор из k кластеров, x – точка данных, μ_i – среднее значение точек данных в i -том кластере.

Алгоритм состоит из следующих шагов:

1. Случайная инициализация k центроидов кластеров.
2. Назначение каждой точки данных ближайшему центроиду.
3. Пересчет центроидов, как среднего значения точек данных, назначенных каждому кластеру.
4. Повторение шагов 2 и 3 до сходимости.

Алгоритм K -средних чувствителен к начальному размещению центроидов и может сойтись к локальному минимуму. Для устранения этого недостатка можно использовать алгоритм K -средних++, который инициализирует центроиды таким образом, что обеспечивает лучшие результаты, чем случайная инициализация [62]. Алгоритм K -средних имеет различные применения, включая сегментацию изображений, кластеризацию документов и сегментацию рынка.

Другим вариантом K -средних является алгоритм сферических K -средних, который особенно эффективен при обработки разреженных и многомерных данных, таких как векторы документов [63]. Алгоритм сферических K -средних отличается тем, что он использует меру косинусного сходства для измерения расстояния между векторами, и поэтому будет использован для реализации предложенного метода векторизации путем применения его на этапе извлечения концептов из текста.

Причиной выбора сферических K -средних является то, что косинусное сходство более эффективно по сравнению с вариантом K -средних, кроме того, целевая функция в сферическом алгоритме K -средних больше подходит для идеи центроида как представления концепта. Что касается оценки эффективности разработанного метода векторизации текстов в задаче кластеризации текста, то применяется обычный алгоритм K -средних ввиду его гибкости и

масштабируемости, что делает этот выбор более подходящим для различных типов векторов.

Для задачи классификации были проведены предварительные эксперименты с использованием таких классификаторов, как искусственная нейронная сеть (ANN), метод случайного леса (Random Forest, RF) и метод опорных векторов (SVM), на векторах, созданных предложенными методами. SVM показал наилучшие результаты и является одним из общепринятых подходов для оценки дискриминационной способности векторов. Следовательно, он будет использоваться во всех экспериментах для оценки производительности предложенных методов по сравнению с другими подходами.

В этом пункте описаны задачи классификации и кластеризации, а также проанализированы методы их решения, которые будут применяться в данном исследовании при построении предлагаемых методов и в вычислительных экспериментах по оценке эффективности.

Чтобы оценить пригодность конкретного алгоритма и определить формат входных и выходных данных, важно полностью понимать все этапы процесса интеллектуального анализа данных, которые анализируются в следующем пункте.

1.1.3. Этапы процесса интеллектуального анализа данных

Интеллектуальный анализ данных – это основополагающий этап в цикле извлечения знаний. Традиционно выделяют следующие стадии в процессе интеллектуального анализа данных [47]:

1. Изучение предметной области, в результате которого формулируются основные цели анализа.
2. Сбор данных. На этом этапе происходит сбор необходимых данных. Это может включать в себя сбор данных из различных источников, таких как базы данных, файлы, API и т.д.
3. Предварительная обработка данных:
 - (а) Очистка данных – исключение противоречий и случайных "шумов" из исходных данных

(б) Интеграция данных – объединение данных из нескольких возможных источников в одном хранилище

(с) Преобразование данных. На этом этапе данные преобразуются к форме, подходящей для анализа. Часто применяется агрегация данных, дискретизация атрибутов, сжатие данных и сокращение размерности информационного пространства.

4. Анализ данных. В рамках этого этапа применяются алгоритмы интеллектуального анализа с целью извлечения информационных паттернов.

5. Интерпретация найденных паттернов. Данный этап может включать визуализацию извлеченных паттернов, определение среди них действительно ценных на основе некоторой функции полезности.

6. Использование новых знаний.

В этом пункте рассмотрены общие этапы процесса интеллектуального анализа данных. Операции, выполняемые на каждом этапе, различаются типами анализируемых данных. В данной диссертации разрабатываются методы анализа текстовых данных, поэтому в следующем пункте анализируется процесс интеллектуального анализа текста на естественном языке, а также его связь с технологиями Data mining.

1.1.4. Интеллектуальный анализ текста

Текстовые данные наиболее часто среди других источников используются для обмена и выражения информации. Кроме того, текстовые данные могут быть сгенерированы из других информационных источников, таких как автоматизированные системы опросов, путем преобразования звука в текст или добавления пояснений к видео для систем поиска информации и рекомендаций. Объем таких данных быстро растет, и возникает потребность в инновационных методах извлечения скрытой информации из них [64].

Анализ текста, также известный как интеллектуальный анализ текстовых данных или текстовая аналитика, представляет собой процесс извлечения высококачественной информации из текста. Он включает в себя автоматическое

обнаружение новой, ранее неизвестной информации путем извлечения и структурирования данных из различных источников, таких как веб-сайты, книги, электронные письма, обзоры и статьи. Для получения информации высокого качества часто используются методы разработки паттернов и анализа статистических тенденций [64, 65].

Интеллектуальный анализ текста – это часть интеллектуального анализа данных, которая фокусируется конкретно на неструктурированных и слабоструктурированных текстовых данных. Это междисциплинарная область, основанная на поиске информации, искусственном интеллекте, статистике, машинном обучении и компьютерной лингвистике. Интеллектуальный анализ текста предполагает использование методов обработки естественного языка (НЛП) для извлечения ценной информации из неструктурированных и слабоструктурированных текстовых данных [66].

Процесс интеллектуального анализа текста включает в себя такие этапы, как предварительная обработка текста (англ. Preprocessing), поиск информации (англ. Information retrieval) и применение различных алгоритмов интеллектуального анализа текста для извлечения ценной информации из данных [67, 64].

Интеллектуальный анализ текстов включает в себя такие же этапы, что и анализ данных, но с учетом специфики текстовой информации. На первом этапе проведён анализ предметной области в соответствии с темой диссертации. Достижение поставленной в диссертации цели предполагает решение задач построения моделей, методов и алгоритмов для создания низкоразмерных, интерпретируемых представлений текста, а также проверки эффективности полученных результатов.

Практически нет необходимости в сборе данных для экспериментов, проводимых в данном исследовании, так как будут использоваться эталонные наборы данных (BBC, REURTERS, OHSMED, 20NEWSGROUPS). Используемые наборы данных будут подробно описаны в пункте 4.1.1.

Следующий этап – предварительная обработка текста, которая включает в себя несколько методов очистки и подготовки текстовых данных. Некоторые из

распространенных методов предварительной обработки включают в себя следующие процедуры [68–72]:

1. **Токенизация** (англ. Tokenization) – это процедура разбиения текста на более мелкие единицы, называемые токенами. Этими токенами могут быть отдельные слова, фразы или даже целые предложения. Токенизация – это фундаментальный шаг в (NLP) и интеллектуальном анализе текста, поскольку он помогает привести текстовые данные к структурированной форме, которая может обрабатываться машиной. Для токенизации в NLP используются следующие инструменты: NLTK [73]; SpaCy [72]; Stanford CoreNLP [74] и GENSIM [70].

2. **Разметка части речи** (англ. Part of Speech Tagging, POS), также известная как грамматическая разметка, представляет собой процесс разметки слова в тексте. Данная разметка соответствует определенной части речи и должна быть связана с контекстом. Маркировка POS предполагает присвоение определенного тега слову в тексте, которое представляет его часть речи, например, существительное, глагол, прилагательное или наречие. Это помогает понять структуру и смысл текста.



I like to read books

Рисунок 1.2 – Пример маркировки частей речи

3. **Стеминг**. Этот процесс используется для построения базовых форм слов, то есть удалить множественное число из существительных, *-ing* из глаголов или других аффиксов. Основа – это естественная группа слов с одинаковым (или очень похожим) значением. После процесса создания основы каждое слово представлено своей основой [75, 76].

4. **Лемматизация**. Этот процесс заключается в уменьшении слов до их базовой или словарной формы, сосредоточиваясь на корневой форме слов. Лемматизация более точна, чем стеминг, так как проводит морфологический анализ слов, но может быть медленнее и менее агрессивным, чем стеминг.

Стеминг и лемматизация являются важными этапами предварительной обработки текста на естественном языке. Они помогают уменьшить размер слов и сделать текст более управляемым для дальнейшей обработки. В практике оба метода могут использоваться вместе, и стеминг часто предпочтительнее, когда пространство слов имеет малый размер и при сравнительно больших текстовых объёмов документов, в то время как лемматизация предпочтительна, когда пространство слов большое, а документы маленькие. Важно выбрать соответствующую технику в зависимости от конкретной задачи и характеристик данных [68].

Для выполнения лемматизации и стеминга на Python можно использовать библиотеки, такие как Spacy², NLTK³ и WordNetLemmatizer. Эти библиотеки предоставляют различные функции и инструменты для предварительной обработки текстовых данных с применением машинного обучения.

5. Удаление стоп-слов включает исключение из текста общеупотребительных слов, не имеющих важного значения. Однако не всегда это нужно делать, так как можно удалять важную информацию.

Применяя эти методы предварительной обработки, текстовые данные очищаются и систематизируются, что делает их готовыми к дальнейшему анализу с использованием алгоритмов Data mining для получения значимой информации. Как правило, первым шагом в анализе текста (после очистки текстовых данных и уменьшения шума) является представление текста в понятной машине форме с использованием корректной модели [77, 67]. Для этого используется множество моделей и методов представления текста для машинной обработки, таких как векторная модель [78], языковая модель (англ. Language models) [79] и тематическая модель (англ. Topic models) [18] и другие.

Процесс представления текста в виде числового вектора в соответствии с этой моделью называется **векторизацией** [80]. Векторизация позволяет компьютеру

² Подробно это будет объяснено в четвёртой главе.

³ Natural Language Toolkit

использовать для работы с текстом математические и статистические методы, такие как машинное обучение и анализ данных.

В данном пункте были даны определения основных понятий предметной области. Проанализированы методы интеллектуального анализа данных и, в частности, текстовых. Изучены этапы процесса интеллектуального анализа данных. Рассмотрены перспективы применения данных этапов для анализа текста на естественном языке. Раскрыта основная информация для дальнейшего выбора направлений исследования.

В следующем пункте проводится аналитический обзор методов представления текста, уделяется внимание концепции векторизации текста, а также наиболее важным методам, используемым для решения поставленных задач.

1.2. Обзор методов представления текста для машинной обработки

Формальное представление текста – это математическая структура, построенная по неструктурированному тексту [44]. Формальным представлением текста может быть алгебраическая структура, теоретико-множественная или графовая структура, комбинация распределений вероятностей слов. В данном исследовании применяются векторные представления, принадлежащие к алгебраической структуре. Далее в пункте проанализированы основные модели и методы векторного представления текстов на естественном языке.

1.2.1. Методы представления текста на основе векторной модели

Векторная модель – это одна из наиболее популярных моделей представления текста [34, 81]. В данной модели текст представляется вектором в пространстве слов (или других элементов текста, так называемых, терминов), причём каждому термину соответствует своя координата векторного пространства. В основе этой модели лежит так называемый мешок слов (англ. Bag of words) – принцип максимального упрощения структуры текста [82]. Согласно этому принципу, текст является множеством или мультимножеством входящих в него слов без учёта коротких и длинных, в том числе, анафорических и кореферентных связей [83].

Для улучшения представления *BoW* предложена модель векторизации текстов на основе статистической меры *TF-IDF* [84], которая в качестве значения вектора использует частоту термина в тексте. Если в общем пространстве терминов представляют два или более текста – так называемую коллекцию текстов – часто используют *TF-IDF* кодировку значений вектора, равную количеству вхождений термина в данный текст, делённому на логарифм относительного количества текстов, содержащих это слово [85].

$$tf\text{-}idf(t, d, D) = tf(t, d) \cdot idf(t, D) = \frac{n_t}{\sum_k n_k} \cdot \log \frac{|D|}{|\{D_i \in D | t \in D_i\}|}. \quad (7)$$

В этой формуле первый множитель *tf* – это локальный вес, то есть, частота термина *w* в тексте *d*, где n_t – количество вхождений слова *t* в документе *d*, а n_k – общее количество слов в этом документе. Второй множитель *idf* – это глобальный вес, показывающий логарифм от величины, обратной доле текста D_i , содержащий термин *w* среди общего числа текстов $|D|$. D_i – количество документов из коллекции *D*, в которых встречается *t*. Соответственно, *TF-IDF* снижает вес часто встречающихся во всех текстах коллекций терминов и повышает вес терминов, характерных для данного текста.

Для иллюстрации того, как работает метод, приведен следующий пример (рис. 1.3). Пусть дан текст, содержащий 5 слов: «Ребенок играет с новой машиной», – и слово «машина» встречается в нём 1 раз, то частота слова «машина» (TF) в документе будет 0,2 (1/5). IDF вычисляется как десятичный логарифм отношения количества всех документов к количеству документов, содержащих слово «машина». Таким образом, если «машина» содержится в 5 документах из 100 документов, то IDF будет равной: $\log(100/5) = 1,31$. TF-IDF вес для слова «машина» в выбранном документе будет равен: $0,2 \times 1,31 = 0,262$. По такому же механизму вычисляются веса остальных четырех слов, в результате формируется вектор документа, в котором каждый элемент представляет вес конкретного слова.

Ребенок играет с новой машиной

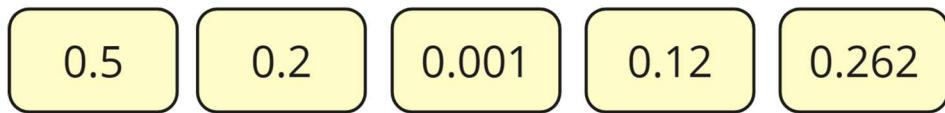


Рисунок 1.3 – Вектор TF-IDF

Основным преимуществом меры *TF-IDF* является её простота, а также то, что векторное представление текста позволяет использовать линейные алгебраические операции для решения многих задач анализа текста, таких как классификация [86, 87], кластеризация [88], категоризация [89, 90], определение сходства между текстами, а также в задаче упорядочения текста в соответствии с запросом [91, 92].

Однако, за внешней простотой кроются существенные недостатки. Прежде всего, векторная модель *BoW*, не учитывает синонимы между словами и не может уловить смысловые связи между ними. Кроме того, использование нормированного скалярного произведения в качестве меры сходства приводит к тому, что длинные тексты всегда имеют низкую степень сходства с остальными. Другой проблемой является размерность, поскольку количество признаков в результирующих векторах экспоненциально увеличивается с размером корпуса текста. Если размер вектора становится слишком большим и разреженным, тогда традиционные метрики определения расстояний (евклидово расстояние, косинусное расстояние) теряют смысл.

Перечисленные недостатки требуют уменьшения размерности векторного пространства для повышения эффективности определения семантических отношений. Для этого разрабатываются методы тематического моделирования, анализ которых проведён в следующем пункте.

1.2.2. Методы тематического моделирования

Тематическое моделирование – это метод статистического моделирования, используемый при обработке и анализе текстов на естественном языке для обнаружения абстрактных «тем», существующих в коллекции документов [93, 94]. Он включает в себя выявление шаблонов в словах и фразах, встречающихся в

документах, и группировку похожих слов по темам для создания тематических кластеров. Тремя наиболее распространенными методами тематического моделирования являются латентно-семантический анализ (Latent semantic analysis, LSA), вероятностный латентно-семантический анализ (англ. Probabilistic latent semantic analysis, pLSA) и скрытое распределение Дирихле (англ. Latent Dirichlet Allocation, LDA).

Скрытое распределение Дирихле (LDA) – это популярный алгоритм моделирования тем, используемый при обработке и анализе текстов на естественном языке для выявления скрытых тем в коллекции документов. В LDA документы представлены как смесь тем, и каждая тема представляет собой группу слов, находящихся в скрытом слое. Алгоритм находит для данного документа значение количества слов, принадлежащих определенной теме, а также определение числа документов для данной темы на основе анализа набора слов.

LDA находит апостериорную вероятность распределения тем по словам в документе. Но поскольку в данной работе используется векторное представление на основе концептов, автор проверил возможность применения LDA для извлечения тем и использования их вместо концептов для построения векторов. Оказалось, что эффективность полученных векторов невысокая, а их точность в анализе текста значительно уступает другим методам, поэтому было принято решение не использовать LDA в вычислительных экспериментах.

Латентный семантический анализ (LSA) – это метод тематического моделирования, используемый при обработке и анализе текстов на естественном языке для обнаружения скрытых семантических шаблонов в текстовых данных. LSA анализирует отношения между набором документов и содержащимися в них терминами, создавая набор тем, связанных с документами и терминами. Он предполагает, что близкие по значению слова будут встречаться в похожих фрагментах текста. LSA широко используется для снижения размерности векторного пространства представляемых документов [95]. В контексте применения к поиску информации его иногда называют скрытой семантической индексацией (Latent Semantic Indexing, LSI) [43].

Учитывая набор документов d_1, d_2, \dots, d_n и набор словарных слов w_1, w_2, \dots, w_m , LSA строит матрицу терминов документов X размерности $m \cdot n$, где элемент $x_{i,j}$ представляет общее количество вхождений w_j в d_i . Для уменьшения количества строк при сохранении структуры сходства между столбцами (уменьшения размерности матрицы документ-термин) применяется усеченное сингулярное разложение (SVD). Этот процесс представлен формулой 8, где k представляет собой количество рассматриваемых тем:

$$X \approx U_{m,k} \Sigma_{kk} V_{n,k}^T. \quad (8)$$

Низкоразмерный вектор документа i строится на основе выражения $\sum_{k,k} d_i$, где d_i – это i -я строка матрицы V . Приближение X происходит за счет выбора k наибольших элементов в первичной диагональной матрице σ и соответствующих столбцов и строк в матрицах U и V из исходного сингулярного разложения. Усеченное сингулярное разложение эффективно реализуется и обновляется при добавлении новых документов [96].

Как метод представления текста в виде семантических векторов, LSA имеет некоторые недостатки, в том числе сложность интерпретации полученных измерений, частичный учёт полисемии и ограничения модели BoW. Кроме того, LSA чувствительны к входным параметрам, таким как количество тем и размер входных документов. Они могут показывать плохую производительность при работе с короткими текстами, так как в них отсутствует контекстуальность информации, доступная в более масштабных документах.

Несмотря на эти недостатки, LSA по-прежнему эффективен для исследовательского анализа, кластеризации документов и уменьшения размерности пространства. Данный метод использовался в данном исследовании в качестве аналога для сравнения с предлагаемыми методами.

В следующем пункте проанализированы прогнозные модели, которые в отличие от тематических, способны извлекать лингвистически ценные представления слов на основе оценки контекста их окружения, другими словами, с учетом порядка и улучшения поиска синонимов.

1.2.3. Методы векторизации текстов на основе методов встраивания слов

Методы встраивания слов используются для векторного представления слов, которое позволяет оценить их семантическую близость. Самая популярный метод встраивания слов это Word2Vec (W2V), который генерирует распределенные векторные репрезентации слов фиксированной длины, фиксируя семантические и синтаксические отношения слов [20, 97, 98].

W2V объединяет несколько двухслойных нейронных сетей для построения встраивания слов, а именно архитектуры Continuous Bag-of-words (CBOW) и Skip-gram. В архитектуре CBOW модель предсказывает целевое слово w_i на основе набора его контекстных синонимов. Напротив, архитектура Skip-gram пытается предсказать набор слов контекста, учитывая целевое слово w_i (рис. 1.4).

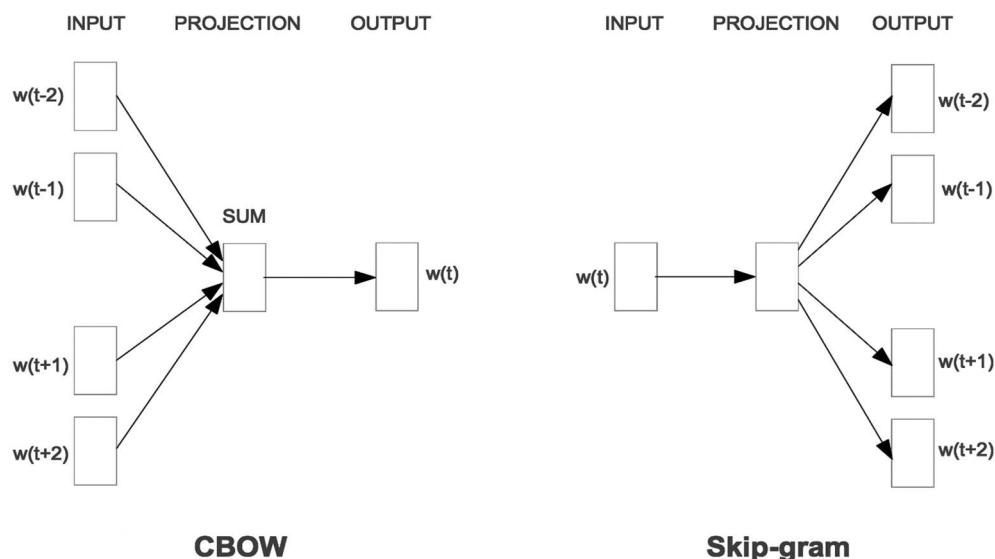


Рисунок 1.4 – Архитектуры ИНС Skip-gram и Continuous Bag-of-Word (CBOW)

Задача встраивания слов является оптимизационной, где целевая функция определена как максимум средней логарифмической вероятности нахождения слова в контексте:

$$\max_{\theta} \frac{1}{T} \sum_{t=1}^T \sum_{0 < |j| \leq c} \log P(w_{t+j} | w_t; \theta), \quad (9)$$

где, θ представляет параметры модели. T – общее количество слов в корпусе. c – размер контекстного окна. w_t представляет целевое слово в позиции t . w_{t+j} представляет контекстное слово в позиции $t+j$. $P(w_{t+j}|w_t)$ – это условная

вероятность наблюдения контекстного слова w_{t+j} при заданном целевом слове w_t и параметрах модели θ .

Скрытый слой представляет векторы слов по мере изучения отношений между словами и контекстом. Основным недостатком W2V является то, что векторные представления слов генерируются локально внутри документов, игнорируя глобальное представление слов во всех документах.

GloVe (глобальные векторы для представления слов) – это метод встраивания слов на основе подсчета частоты встречаемости слов, который использует глобальную матричную факторизацию для создания встраивания слов, фиксируя семантическое значение слов и их отношения в данном корпусе. GloVe основан на алгоритме обучения без учителя, который преобразует слова в плотные векторные представления. Создавая большое число совместных пар (слово × контекст), GloVe решает проблему W2V, заключающуюся в несоответствии глобальному контексту [99]. Однако проблема нестандартных слов в GloVe существует, также как в W2V.

FastText [16] – это расширение W2V, предложенное компанией Facebook в 2016 году. Вместо ввода отдельных слов в нейронную сеть FastText разбивает слова на несколько n-грамм (подслов). Таким образом, он создает векторные представления слов для всех n-грамм, учитывая набор данных обучения. Это позволяет правильно представлять редкие слова, поскольку с высокой вероятностью большинство из данных n-грамм будут встречаться в других словах.

Каждый из этих трех методов встраивания слов имеет свои преимущества и недостатки. Выбор конкретной метод зависит от характера решаемой задачи. Далее, в главе 4 описано, как выбор модели встраивания слов влияет на эффективность разработанного метода.

Для получения векторов документов все векторы слов усредняются [100, 101, 23]. Метод усреднения используется с любым типом вектора. Усреднённые векторные представления слов позволяют эффективно строить вектор текста, но при этом существует вероятность потери сложности и нюансов в смысле обрабатываемого текста. Одним из недостатков является потеря порядка слов,

искажающая контекст, так как каждое слово рассматривается одинаково при усреднении векторных представлений [18]. Это приводит к потере важной синтаксической и семантической информации. Кроме того, редкие или важные слова могут иметь меньшее влияние на конечное векторное представление текста, так как они усредняются с более общими словами.

В качестве альтернативы было предложено расширение W2V, известное как Doc2Vec (D2V) или «векторы абзацев» (paragraph to vector) [17, 18]. Doc2Vec – это популярный метод представления текста, который генерирует векторные представления слов и документов, фиксируя их семантическое значение и связи с другими словами и документами. Doc2Vec создает векторы документов непосредственно путем рассмотрения документа как специального контекстного токена, который добавляется в обучающие данные, чтобы модель могла изучать векторы токенов и рассматривать их как векторы документов. На основе W2V алгоритм D2V так же имеет два варианта реализации архитектуры нейронной сетей: распределенная память (Distributed Memory, DM) и распределенный мешок слов (Distributed Bag of Words, DBOW).

Doc2Vec обрабатывает слова, используя контекст, в котором они появляются в документе, в отличие от *TF-IDF*, которая полагается на частоту слов в корпусе. Для оптимизации производительности Doc2Vec использует ряд гиперпараметров, таких как: число итераций обучения; скорость обучения; алгоритм обучения. Обучение моделей Doc2Vec может быть дорогостоящим в вычислительном отношении и обычно требует обширных текстовых данных. Кроме того, векторы, которые он генерирует, плотные и не поддаются интерпретации.

В качестве альтернативного подхода к векторному представлению текста на основе слов, представление на основе концептов появилось как решение проблемы размерности и сохранения интерпретируемости. В этом подходе для представления текста вместо слов используются концепты, где каждое измерение вектора соответствует одному концепту. Этот подход основан на интеллектуальном анализе концептов как на важном предварительном этапе векторизации текста и обусловлен тем фактом, что концепт даёт развернутую информацию о содержании

документов на разных уровнях абстракции. В следующем пункте анализируются некоторые важные методы, соответствующие этому подходу, которые составляют основу разработки методов, предлагаемых в данной работе.

1.2.4. Методы векторизации текстов на основе построения концептов

При обработке и анализе текстов на естественном языке концепт определяется как набор слов или фраз, имеющих общее семантическое значение [29, 102, 103]. В литературе представлено несколько работ, посвященных обнаружению концептов при создании представлений документов. Данные работы по источнику информации делятся на два подхода [104, 105].

Первый подход основан на использовании внешних источников, таких как базы знаний или лексические базы данных, для обнаружения концептов. Методы, основанные на этом подходе, сначала извлекают характерные ключевые слова документа с использованием классических методов индексирования (токенизация, лемматизация, удаление стоп-слов и т. д.), а затем присваивают эти слова соответствующим записям в базе знаний, Википедии или онтологии, проходя через фазу устранения неоднозначности слов [103, 106–108].

Формально понятие (концепт) определяется следующим образом: Пусть $N = \{N_1, N_2, \dots, N_l\}$ – множество слов в документе, а R – это множество лексических отношений, таких как: синоним; гипероним (гипоним); мероним и др. Концепт C состоит из N_i и R_k , где $R_k \in R$, $N_i \in N$. Каждый N_i и C_j имеют вес, который представляет их соответствующие степени семантической близости в документе.

Вес определяется либо статистически (через его частотное распределение в содержимом документа) с использованием нормализованной версии классической схемы *TF-IDF*, как в [109], либо семантически через его центральность (англ. Centrality) в документе, т.е. его семантическое значение, связь с другими понятиями, как в [29, 110]. Хотя данный подход, дает конкретные и ясные концепты, он менее гибок и масштабируем, тем более, не все понятия, упомянутые в тексте, имеют запись в базе знаний или Википедии, в связи с чем сложно определить описываемые связи согласно описанию понятия.

Второй подход опирается только на текст как источник информации [3, 29, 22, 37, 111]. Для построения концептов методы этого подхода используют алгоритмы машинного обучения, для решения задач кластеризации и/или ассоциации при выявлении терминов, имеющих одно общее значение. Под общим значением подразумевается то, что между словами существует семантическая связь, например, они являются синонимами или имеют одинаковый гипоним.

Отличие от концепта, описанного при первом подходе, заключается в характере отношений, объединяющих слова в рамках единого понятия. В текущем подходе невозможно дать название отношениям, которые объединяют два слова, например, синонимы, гипероними и т.д. Это потому, что нет формального описания этих отношений, как в случае с онтологией, базой знаний и Википедией. Обозначение отношений не имеет большого значения, поскольку в задачах такого типа важна степень семантической близости как мера сходства между словами. Одним из наиболее распространенных способов вычисления семантической близости является оценка меры косинусного сходства между векторами слов.

В данной диссертации автора интересует второй подход, который извлекает концепты исключительно из текста. Данный подход в значительной степени опирается на то, как слова представлены при построении концептов. С появлением методов встраивания слов [97] появились попытки использовать эти представления для построения концептов и создания низкоразмерных векторных представлений документов. Эти работы направлены на то, чтобы определить семантическое значение слов или документов, представляя их в пространстве более низкой размерности, что позволяет сохранить эффективность векторов при решении задач обработки и анализа текстов.

Одна из этих работ представлена в [22, 12], где авторы разрабатывают метод, называемый «мешок концептов» (Bag of Concepts, BoC). BoC создает концепты, группируя векторы слов в кластеры, а затем использует частоты этих кластеров для представления векторов документов. На рисунке 1.5 представлена функциональная схема метода BoC. Блоки, отмеченные красным цветом, указывают местонахождение недостатков метода BoC. Как показано на рисунке 1.5, входными

данными являются текстовые документы, а выходными – их концептуальные векторы. Метод включает в себя два этапа: первый – построение концептов, второй – векторизация документа

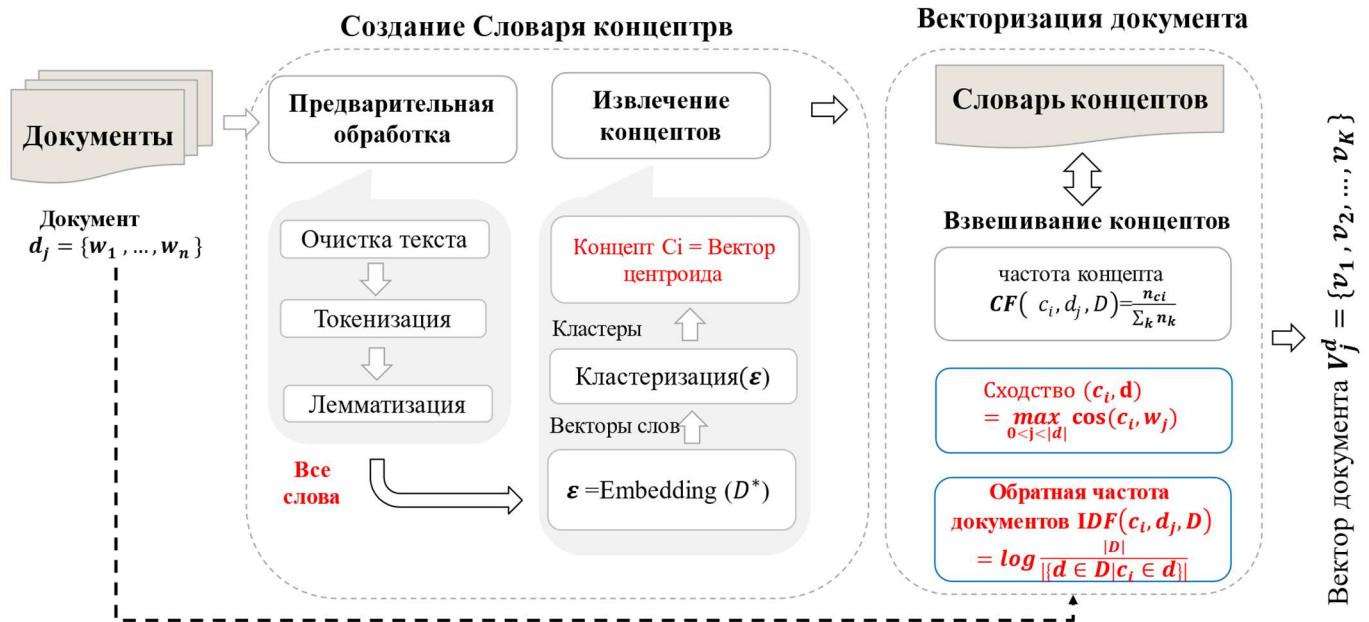


Рисунок 1.1 – Функциональная схема метода ВоС

Этап построения концептов включает в себя обработку текста, извлечение уникальных терминов, а также процедуры встраивания и кластеризации слов.

ВоС генерирует векторы слов документа с использованием моделей W2V, которые встраивают семантически схожие слова в соседнюю область. Это позволяет сгруппировать соседние слова в один общий кластер концептов. Кластеры слов генерируются на основе применения алгоритма К-средних к векторам слов. Полученные кластеры содержат слова с похожими значениями и являются концептами. Полученный набор кластеров образует «словарь эталонных понятий (концептов)» или, другими словами, «словарь понятий (концептов)», представленный на рисунке 1.6.

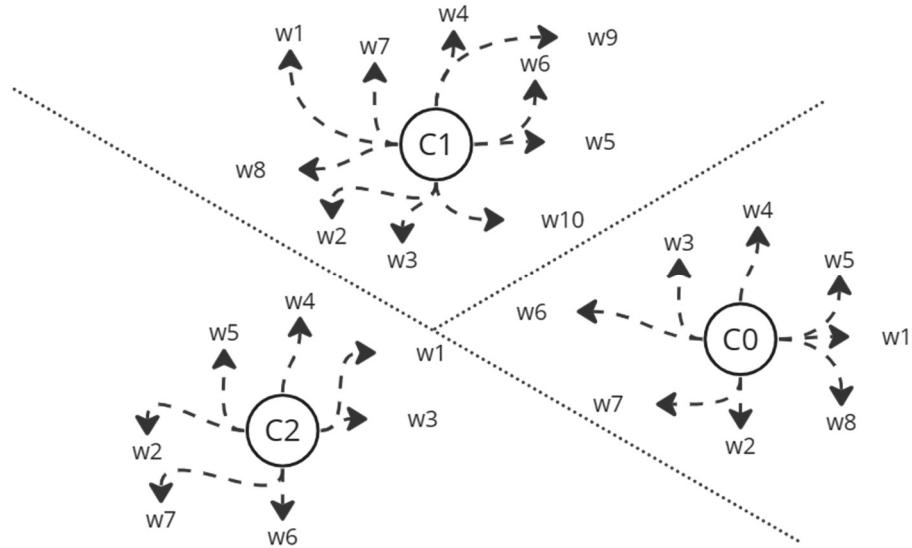


Рисунок 1.2 – Словарь эталонных концептов

В практическом применении концепт представляется вектором центроида кластера. Поэтому словарь эталонных концептов описан следующим образом:

$$\mathbf{C} = \text{clustering}(\varepsilon) = (\vec{c}_1, \vec{c}_2, \dots, \vec{c}_k), \quad (10)$$

где \mathbf{c}_i это вектор центроида i -го понятия.

Одним из недостатков этого метода является то, что словарь эталонных концептов формируется на основе всех уникальных слов корпуса текстов. Это создаёт дополнительную вычислительную нагрузку при выполнении алгоритма кластеризации для построения концептов. Чем больше размер корпуса текста, тем больше размер входных данных алгоритма и тем выше его вычислительная сложность. Кроме того, это приводит к попаданию маловажных слов в понятия, что увеличивает их зашумленность и делает центроид кластера неточной репрезентацией понятия.

Этап векторизации документов включает в себя сопоставление документов со словарем эталонных концептов с последующим их взвешиванием.

Аналогично методу «мешок слов», каждый вектор документа представлен частотой каждого кластера концептов в документе. Частота концепта в документе определяется на основе значения сходства S_c между концептом и документом:

$$CF(c_i, d_j, D) = \frac{n_c}{\sum_k n_k}, \quad (11)$$

где n_k – общее количество концептов в документе, а n_c – количество вхождений концепта c в документ. Вхождение концепта в документ подтверждается в том случае, если значение сходства S_c концепта со словом документа превышает заданный порог (Θ).

$$g(s) = \begin{cases} 1, & S_c > \Theta \\ 0, & \text{иначе} \end{cases}, \quad (12)$$

Сходство S_c между двумя векторами слов X (Вектор центроида) и Y (Вектор слова документа) вычисляется следующим образом [6]:

$$s = similarity(X, Y) = \cos(\theta) = \frac{X \cdot Y}{\|X\| \cdot \|Y\|} = \frac{\sum_{i=1}^n X_i Y_i}{\sqrt{\sum_{i=1}^n X_i^2} \sqrt{\sum_{i=1}^n Y_i^2}}, \quad (13)$$

где X_i и Y_i – компоненты вектора слов X и Y , соответственно.

Стоит отметить, что рассчитанное значение сходства используется только для расчета частоты и не включается непосредственно в весовую функцию. Неучет рассчитанных показателей сходства в весовой функции концептов считается одной из слабых сторон этого метода (рис 1.5).

Для смягчения влияния концептов, которые появляются в большинстве документов, используется весовая схема, аналогичная *TF-IDF*, заменяя частоту термина TF на частоту концепта CF. Поэтому она называется *CF-IDF* (Частота концепта с обратной частотой документа) и рассчитывается на основе следующего уравнения:

$$CF-IDF(c_i, d_j, D) = \frac{n_{c_i}}{\sum_k n_k} \cdot \log \frac{|D|}{|\{d \in D | c_i \in d\}|}, \quad (14)$$

где $|D|$ - количество документов в коллекции, а в знаменателе – количество документов из коллекции D , в которых встречается концепт c ; n_c – это количество вхождений понятия c в документе d , а n_k – общее количество понятий в этом документе.

Метод *BoC* решает проблему большой размерности векторного пространства в методе *BoW*, поскольку он линейно уменьшает размерность при преобразовании

пространства слов в пространство понятий на основе семантического сходства. Кроме того, было показано, что *BoC* обеспечивает лучшее представление документов, чем *BoW* и *TF-IDF*, в задаче классификации для нахождения двух наиболее похожих документов среди троек документов [12].

По мнению автора данного исследования, использование функции IDF нецелесообразно для снижения важности общих понятий. Причина этого в том, что появление понятия в документе связано с появлением в нем различных слов, имеющих общее значение. Поэтому использование формулы *CF-IDF* на уровне концепта неэффективно, так как частота появления концепта на уровне коллекции документов намного выше, чем частота одного из его терминов. Математически это приводит к очень малым значениям логарифма – близкими к нулю, что указывает ложный вывод.

Следуя тому же подходу, что и *BoC*, метод Vectors of Locally Aggregated Concepts (*VLAC*) группирует слова для формирования N эталонных концептов. Однако вместо вычисления частоты сгруппированных векторов слов *VLAC* использует тот же подход, что и метод *VLAD* (локально агрегированные векторные дескрипторы), который представляет собой метод генерации признаков, используемый для классификации изображений [22]. В *VLAD* изображения представлены числом вхождений его кластеризованных признаков (т. е. дескрипторов), где он использует кластеры признаков изображения вместо кластеров слов, в дополнение к включению статистики первого порядка (англ. First-Order Statistics, *FOS*) в векторы признаков. Подобно исходному подходу *VLAD*, *VLAC* берет сумму остатков каждого кластера относительно его центроида и объединяет их для создания вектора признаков.

Даны вектор слова w_i размером D , и c_k – центроид кластера. Для каждого слова в документе метод *VLAC* вычисляет поэлементную сумму остатков каждого слова, внедренного в назначенный ему центроид кластера. В результате получается k векторов признаков, по одному для каждой концепции, и все они имеют размер D. Затем все векторы признаков объединяются, нормализуются по степени и применяется стандартная нормализация $l2$, как и в исходном подходе *VLAD*.

Результирующие векторы признаков содержат более ценную информацию, чем векторы признаков BoC, благодаря дополнительному включению статистики первого порядка. Однако этот метод генерирует векторы относительно больших размеров с большими вычислительными затратами. Если бы из вложений из 300 слов было создано 30 понятий, результирующий вектор документа содержал бы 30×300 значений.

В данном пункте проанализированы два метода, использующие один и тот же подход извлечения понятий из текста. Выявлены их преимущества и недостатки. В дальнейшем, при разработке нового метода векторизации текста автором учтены результаты данного анализа.

В следующем пункте определены основные направления исследований на основе проведения общего сравнения методов векторизации текста. Результаты данного сравнения опираются на представленный в первом разделе аналитический обзор методов обработки и анализа текстов на естественном языке.

1.3. Общее сравнение методов представления текста

В приведённом аналитическом обзоре внимание уделяется двум основным характеристикам векторов, представляющих документы, а именно размерности и интерпретируемости.

Векторы большего размера при интеллектуальном анализе текста могут фиксировать больше признаков и повышать эффективность решения задач, на пример, таких как анализ настроений и моделирование тем. Тем не менее, есть негативные последствия, которые следует учитывать. Разреженность данных может создавать проблемы для алгоритмов, предполагающих обработку плотных (dense) данных, а векторы более высокой размерности требуют больше вычислительных ресурсов и занимают больше памяти, что влияет на эффективность и масштабируемость.

Интерпретируемость векторов признаков играет важную роль в текстовом анализе по нескольким аспектам. Она позволяет понять, как модель принимает решения, анализируя важность каждого признака, что способствует отладке и

улучшению модели. Интерпретируемые векторы признаков помогают выявлять и устранять предубеждения в текстовом анализе, анализируя влияние каждого признака на прогнозы модели. Это повышает доверие пользователей к моделям текстового анализа, так как пользователи могут понять и проверить признаки, влияющие на прогнозы.

Кроме того, интерпретируемость вектора связана с понятием дискриминационной способности вектора документа. Дискриминационная способность показывает, насколько эффективно вектор может различать разные классы или категории на основе своих признаков [9]. Это свойство позволяет оценить, насколько эффективно векторные представления документов могут разделять данные на основе определенных признаков. В контексте машинного обучения и анализа данных дискриминационная способность измеряется с помощью различных методов, таких как дискриминантный анализ или метод опорных векторов (SVM). В данной работе дискриминационная способность векторов определяется их применением для решения задач классификации и кластеризации текста с использованием алгоритмов машинного обучения, таких как метод опорных векторов (SVM) и K-средних [57].

Векторное представление на основе *BoW* и *TF-IDF*, показало хорошие результаты в задачах классификации и кластеризации документов, благодаря своей простоте, эффективности, точности и интуитивной способности интерпретировать векторы, которые они производят. Однако у них есть два основных недостатка. Во-первых, они часто имеют разреженные данные и высокую размерность. Во-вторых, эти методы не учитывают семантические отношения между словами. Эти недостатки ограничивают способность моделей анализа текста в точности отображать сходство между документами.

Скрытый семантический анализ (LSA) полезен для уменьшения размерности вектора по сравнению с *TF-IDF* и *BoW*, но используемое уменьшение размерности требует больших вычислительных затрат и создает неинтерпретируемые представления, что затрудняет понимание рабочей логики алгоритма анализа текста.

Методы встраивания слов могут обнаруживать скрытые значения и гибко манипулировать исходными текстами, что приводит к плотным и компактным представлениям слов, абзацев или документов. Использование инструмента векторного представления документа, такого как doc2vec, делает полученный вектор неинтерпретируемым, что требует тонкой настройки параметров для получения результатов, сопоставимых с традиционными методами. Для создания представлений более длинных текстов, таких как предложения или документы, с использованием векторного представления слов, распространены методы усреднения векторов слов, их суммирования или конкатенации. Операции суммирования и конкатенации векторов не решают проблему больших размерностей, а операции усреднения игнорируют некоторую информацию в документе. Кроме того, векторы после обработки теряют свою интерпретацию и быстро деградируют.

Соответственно, можно сказать, что применение методов встраивания слов для представления более длинных текста без потери информации при сохранении интерпретируемых признаков и их дискриминационной способности является задачей, которая до конца не решена.

Представления векторов, создаваемых описанными методами, являются либо многомерными, что приводит к разреженным векторам и, следовательно, требует больше памяти, либо низкоразмерными, не интерпретируемыми, что, в свою очередь, требует более высоких вычислительных затрат на обучение, настройку и обработку.

Таким образом, противоречие в теории обусловлено тем, что существующие методы не позволяют обеспечить семантические представления документов (векторов) с малыми размерностями, которые можно интерпретировать без негативного влияния на эффективность алгоритмов классификации и кластеризации с точки зрения обобщаемости результатов, и возможности интерпретации логики их работы.

1.4. Постановка задачи исследования

Для достижения цели исследования – повышение эффективности моделей, методов и алгоритмов классификации и кластеризации текстов, необходимо решить общую задачу исследования, а именно, разработать новые модель, алгоритмы и метод генерации векторных представления текста (векторизации текстов) с использованием методов интеллектуального анализа данных, позволяющих построить низкоразмерные векторное представление с высокой дискриминационной способностью и интерпретируемыми признаками.

Таким образом, общую постановку задачи исследования можно сформулировать в следующем виде.

Дан набор текстовых документов, представленных в виде последовательностей слов. Требуется построить модель для классификации и кластеризации текстовых документов в заранее определенные классы (для задачи классификации) или в кластеры (для задачи кластеризации). Разрабатываемая модель должна обеспечивать интерпретируемые векторные представления текста таким образом, чтобы в зависимости от векторных признаков можно было объяснить причины отнесения документа к тому или иному классу. Кроме того, сгенерированные представления должны быть малоразмерными, чтобы алгоритм машинного обучения не требовал больших вычислительных затрат. Наконец, применение этих векторов должно привести к тому, что частота ошибок алгоритма будет меньше или равна частоте ошибок, достигаемой с помощью известных методов.

Для решения представленной общей научной задачи необходимо решить **несколько частных задач**.

Как правило, алгоритмы машинного обучения требуют, чтобы текст был представлен в виде числовых векторов с одинаковой размерностью пространства, понятных машине. То есть количество признаков вектора всех документов должно быть одинаковым. Для этого, во-первых, построена и разработан модель и метод для представления текста документа в виде числового вектора на основе концептов, где каждый концепт (понятие) представляет собой определенное измерение. Такое представление считается линейным преобразованием пространства слов в

пространство концептов, позволяющим управлять размерностью векторного пространства. Признаки концепта должны позволять различать содержимое документа и достигать приемлемого качества представления документа на уровне слов.

Во-вторых, концепты, на которых основаны векторы представления документов, должны быть извлечены из самого текста. Исходя из информации, представленной в пункте 1.2.4, извлечение концептов из текста включает в себя два этапа: определение концептов и взвешивание концептов. Построения концептов в соответствии с подходом, использованным в методе *BoC*, происходит с применением методов Data mining, а именно кластеризации.

Алгоритм построения концептов должен позволять идентифицировать слова/фразы документа, наиболее характерные для его содержания, с целью снижения вычислительных затрат при построении концептов и обеспечения их оригинальности. Для этого необходимо оптимизировать алгоритм извлечения ключевых фраз таким образом, чтобы отфильтровать фразы, не относящиеся к контексту документа и способные привести к образованию зашумленных понятий.

В-третьих, для взвешивания концептов и определения того, какие концепты наиболее характерны для конкретной коллекции документов, предложены весовые функции на основе применения методов оценки семантической близости, которые позволяют распределить весовые коэффициенты и упорядочить концепты в соответствии с их важностью на уровне корпуса, и, таким образом, преодолеть недостатки существующих методов.

В-четвертых, для создания векторного представления документов с использованием словаря эталонных концептов построена математическая модель, которая описывает формат входа/выхода разработанного метода и основные функции и операции, необходимые для определения весов концептов.

В-пятых, с целью проведения исследования эффективности предложенных модели и метода векторизации текста разработано прикладное программное обеспечение, а также проведён сравнительный анализ и тестирование

генерируемых им векторов в различных задачах анализа и обработки текстов, таких как классификация, кластеризация и извлечения ключевых фраз.

Сформулируем **формальную постановку** задачи.

Задано конечное множество классов, а также множество объектов, часть из которых распределена по данным классам. Это часть (подмножество) называется обучающей выборкой. Принадлежность к классам остальных объектов не известна. Требуется создать алгоритм, способный классифицировать произвольный объект из исходного множества, т.е. указать номер (или наименование класса), к которому относится данный объект.

В задаче классификации текста объекты – это текстовые документы $D = \{d_1, d_2, d_3, \dots, d_N\}$, где каждый документ $d \in D$ представляет собой последовательность слов $W = \{w_1, w_2, w_3, \dots, w_{n_d}\}$, n_d – длина документа d . $Y = \{y_1, y_2, \dots, y_N\}$ – конечное множество меток классов. $y^* = D \rightarrow Y$ – неизвестная целевая зависимость, значения которой известны только на объектах конечной обучающей выборки $D_m = \{(d_1, y_1), \dots, (d_m, y_m)\}$. Требуется создать алгоритм $\alpha = D \rightarrow Y$, способный классифицировать произвольный объект $d \in D$.

Алгоритм α – это алгоритм машинного обучения, который принимает на вход одномерные числовые векторные представления V . Чтобы получить векторное представление фиксированной длины для всех документов, ставится следующая подзадача: пусть уникальные слова из всех документов образуют так называемый словарь слов W . И пусть $x_i \in R^S$ – векторное представление i -го слова словаря W , где S – размерность вектора слова. Множество всех векторов слов обозначается $\varepsilon = \{x_i, i = 1, \dots, |W|\}$.

Соответственно, документ может быть представлен векторами своих слов $N_i: x_{ij} \in \varepsilon, (i = 1, \dots, N, j = 1, \dots, N_i)$, где N_i – количество слов i -го документа, а x_{ij} – это вектор j -го слова из i -го документа.

Таким образом, необходимо найти преобразование $y = f(x): R^S \rightarrow R^C$, такое, что преобразованный вектор признаков $y_i \in R^C$ сохраняет большую часть преобразования или структуры в R^S . Соответственно, требуется найти

преобразование из пространства слов в пространство концептов, которое позволяет каждому документу быть представленным вектором фиксированной длины. $N_i: c_{ij} = (i = 1, \dots, K, j = 1, \dots, N_i)$, где k – количество извлеченных концептов, а c_{ij} – это признак j -го концепта i -го документа.

Полученное в итоге преобразование $y = f(x)$ позволяет уменьшить значение частоты ошибок алгоритма α .

$$f^* = \arg \min_f E_\alpha(f(x)), \quad (15)$$

$$E_\alpha(f(x)) \leq E_{\alpha,min}, \quad (16)$$

где, E_α – это функция, которая принимает преобразованный входной сигнал $f(x)$ (векторное представление концептов) и выводит частоту ошибок алгоритма α . $E_\alpha(f(x))$ – это частота ошибок алгоритма α при использовании преобразования y для входного параметра x . $E_{\alpha,min}$ – минимальная частота ошибок алгоритма α без каких-либо преобразований, которую можно определить как $E_{\alpha,min} = \min_\alpha E(x)$.

Соответственно, решение описанной выше задачи требует разработки методов и моделей представления текста, позволяющих строить низкоразмерные векторные представления с более информативными и интерпретируемыми признаками, а также применять и тестировать разработанные методы для повышения эффективности систем обработки и анализа текстов на естественном языке.

В данном пункте дана постановка основных задач исследования.

Задача классификации и кластеризации текстов на естественном языке представлена с точки зрения повышения эффективности моделей и методов анализа и обработки текстов, позволяющих повысить качество получаемых результатов в условиях большой размерности.

1.5. Выводы по разделу

В данной главе был представлен аналитический обзор научных исследований в области обработки и анализа текстов на основе современных методов интеллектуального анализа данных.

Даны определения основных терминов необходимых для понимания предметной области диссертационного исследования. Проведён аналитический обзор основных задач и методов технологии Data mining и, в частности, анализа текстовых данных. Изучены этапы процесса интеллектуального анализа данных и перспективы использования этих этапов для обработки и анализа текстов на естественном языке.

Проведённый анализ исследований показал, что методы представления текста, в частности векторного, становятся актуальными при экспоненциальном росте объёмов текстовых данных и являются наиболее значимыми при решении задач обработки и анализа текстов на естественном языке. При анализе методов представления текста основное внимание уделялось двум важным свойствам векторов, а именно, размерности и интерпретируемости.

По результатам проведённого анализа были получены следующие промежуточные результаты:

1. Выявлено противоречие в теории, которое заключаются в том, что существующие классические методы векторизации текстов, такие как *TF-IDF* и *BoW*, являются эффективными для решения задачи классификации и кластеризации документов, но только для небольших размеров текстовых документов. В то же время, современные методы распределенного векторного представления, такие как *LSA*, *D2V* эффективно обнаруживают скрытые семантические зависимости в больших данных, но при этом требуют значительных вычислительных ресурсов и времени, а также не обладают интерпретируемостью и уступают классическим методам в качестве полученных решений. Кроме того, что существующие методы не позволяют обеспечить семантические представления документов (векторов) с малыми размерностями, которые можно интерпретировать без негативного влияния на эффективность алгоритмов классификации и кластеризации с точки зрения обобщаемости результатов, и возможности интерпретации логики их работы. Данное противоречие указывает на необходимость разработки моделей, методов и алгоритмов векторизации документов при решении задач классификации и кластеризации текстов.

2. Сформулирована постановка задачи исследования, а именно задача разработки новых методов и алгоритмов генерации векторных представлений текста, позволяющих строить низкоразмерные векторные представления с более интерпретируемыми признаками.

3. Обоснована необходимость использования концептов для векторизации документов. В этом подходе для представления текста вместо слов используются концепты, где каждое измерение вектора соответствует одному эталонному концепту. Эталонный концепт представляет собой совокупность слов или словосочетаний, имеющих общее смысловое значение. Эталонные концепты извлекаются из самого текста без опоры на внешние источники, такие как базы знаний и онтологии. Сопоставляя слова документа с эталонными концептами на основе методов оценки семантической близости, этот подход позволяет создавать интерпретируемые векторы, а также управлять их размерностями, контролируя количество концептов.

4. Для этой цели проанализированы методы векторизации документов на основе концептов и выявлены недостатки, препятствующие достижению цели диссертации. К таким недостаткам относятся: низкая эффективность алгоритмов построения концептов из свободного текста, что приводит к образованию перекрывающихся понятий (*BoC*, *VLAC*) и появлению зашумленных концептов, не несущих смысла, а также недооценке значимости редких концептов; неадекватное представление концепта, так как представление концепта центроидным вектором кластера не считается точным представлением и может привести к смещению степени взвешивания концепта, особенно когда кластеры разбросаны, и их размеры велики.

5. Определены методы и инструменты, которые использовались автором в дальнейшем для преодоления указанных недостатков и развития предлагаемых методов. Эти методы относятся к пересечению трех направлений, а именно: Data mining, векторного представления слов и методов оценки семантической близости.

6. Описаны задачи классификации и кластеризации, а также проанализированы методы их решения, которые применялись далее в

исследовании при построении предлагаемых методов, а также в вычислительных экспериментах по оценке эффективности полученных решений. Для выполнения задачи классификации на основе создаваемых векторов в основном использовался метод *SVM*, а для кластеризации – алгоритм К-средних. Для получения векторного представления слов протестированы различные типы моделей встраивания слов, такие как *GloVe*, *FastText* и *W2V*. Для измерения семантического сходства между векторами будет использована мера косинусного сходства.

В последующих главах диссертации описаны предлагаемые решения, позволяющие устраниТЬ недостатки методов векторизации документов на основе концептов. Разработанный метод и набор алгоритмов решают задачу построения векторных представлений документов, повышающих эффективность методов векторизации документов. Во второй главе описывается разработка нового метода векторизации документов, основанного на технологиях Data mining для построения словаря эталонных концептов, а также на методах оценки семантической близости для сопоставления слов документа с эталонными концептами и построения векторов. При этом описывается алгоритм построения словаря эталонных концептов и предлагаемые модификации для его улучшения.

ГЛАВА 2. МОДЕЛЬ И МОДИФИЦИРОВАННЫЙ МЕТОД ВЕКТОРИЗАЦИЯ ТЕКСТА НА ОСНОВЕ МЕТОДОВ DATA MINING

2.1. Разработка метода векторизации текста на основе методов Data mining

Разработанный метод основан на подходе к векторизации документов на основе концептов, который является аналогом методу *BoC*. Таким образом, внутренняя логика предлагаемого метода так же состоит из двух этапов, на первом из которых создается словарь эталонных концептов, который используется в качестве основы для построения вектора. Каждый концепт в словаре соответствует одному измерению вектора.

Второй этап заключается в анализе каждого документа и извлечении содержащихся в нем концептов с последующим их взвешиванием путем сопоставления со словарем эталонных концептов. Для каждого документа строится семантический вектор, состоящий из N признаков. Каждый признак представляет степень присутствия концепта в словаре эталонных концептов документа. Контроль числа концептов в словаре эталонных концептов позволяет управлять размерностью создаваемых векторов.

В пункте 1.2.4. были проанализированы недостатки методов векторизации текста на основе концептов. Предлагаемый метод позволяет устраниТЬ эти недостатки, основываясь на этом анализе, а также с учетом результатов сравнения методов и алгоритмов, используемых в области предварительной обработки и анализа текстов и извлечения понятий.

Первое решение имеет место на этапе взвешивания концептов во время векторизации документа. Решение состоит в том, чтобы не ограничиваться центроидом кластера для расчета сходства между словом документа и концептом, а включать слова, ближайшие к центроиду кластера, в функцию определения сходства.

Второе решение имеет место на этапе построения словаря эталонных концептов. Центроид остается репрезентативным для концепта при условии, что

однородность кластера будет улучшена. Для этого предложено оптимизировать алгоритм формирования концептов путем взвешивания и фильтрации терминов, которые будут представлять документ и участвовать в алгоритме кластеризации.

Два предыдущих решения применяются отдельно или вместе. Во всех случаях применение этих решений приводит к улучшению качества получаемых векторов, как показывают вычислительные эксперименты, представленные в разделе 4.

Разработанный метод включает в себя следующие функции:

1. Функция взвешивания извлеченных концептов, в которой учитывается монотонность убывания обратной частоты документа, позволяющая контролировать влияние частоты документа на уровне корпуса и получать низкоразмерные векторные представления;

2. Для определения значимости (веса) концепта в документе введена эвристическая функция. Эта функция на основе оценки сходства слов, составляющих концепт, со словами документа извлекает дополнительную информацию о концептах.

Применение новых функций взвешивания концептов направлено на создание интерпретируемых и информативных признаков, способных эффективно анализировать и обрабатывать содержимое документа. В следующем пункте описано построение функциональной схемы предлагаемого метода и подробно описаны этапы его разработки.

2.1.1. Функциональная схема предлагаемого метода

На рисунке 2.1 представлена функциональная схема метода *BoWC*, которая содержит основные алгоритмы, операции и функции, выполняемые на разных этапах его работы. Синие блоки обозначают предлагаемые новые алгоритмы и функции, компенсирующие недостатки канонических методов.

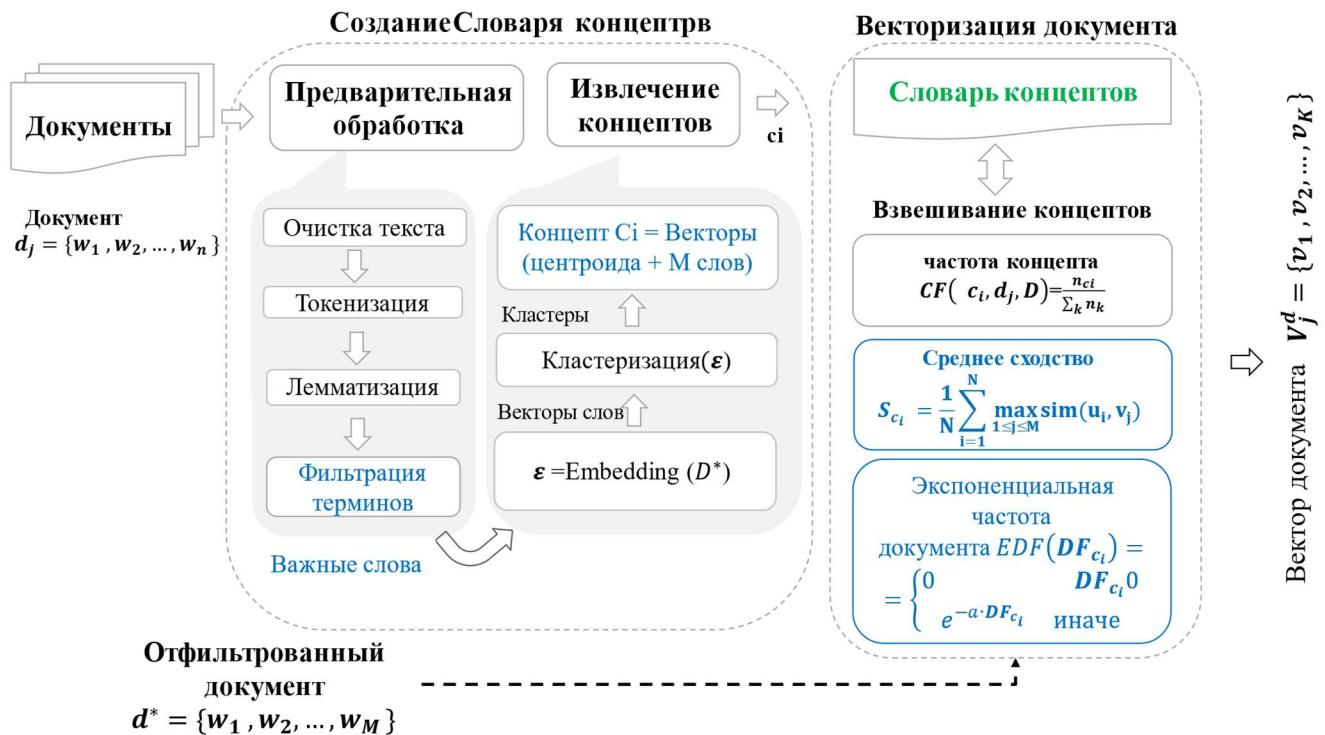


Рисунок 2.1 – функциональная схема метода BoWC

Входные данные, представляют собой текстовые документы, а выходные данные – векторы. Между входом и выходом выполняются наборы операций, направленные на построение концептов и представление документов с этими концептами. Далее описывается каждый этап и подробно описываются используемые математические функции.

При выполнении алгоритмов создания словаря эталонных концептов и их построения на первом этапе проводится обработка документов. Из документов извлекаются уникальные термины, составляющие словарь (англ. vocabulary), затем для каждого термина генерируются векторы встраивания в рамках подготовки к построению словаря эталонных концептов.

Формирование кластеров в предлагаемом методе отличается от простого алгоритма, используемого в методах, анализируемых в пункте 1.2.4. Основное отличие состоит в том, что концепт представлено М словами, ближайшими к центроиду кластера, а не только центроидом. Это позволяет решить первую проблему, заключающуюся в том, что центроида недостаточно для определения наличия или отсутствия концепта в документе. Ниже приводится подробное описание алгоритма формирования концептов.

 Алгоритм 2.1 – Алгоритм построения словаря эталонных концептов

Вход Набор $D = \{d_1, d_2, \dots, d_N\}$ из N документов

1: $M_{desired}$ // минимальное количество слов кластера

Выход $C = []$ // Словарь эталонных концептов

2: Отсканировать документы и создать словарь D

3: Инициализировать вложение слова E с помощью D

4: Инициализировать словарь T , запустив К-средних на E

5: $C = count(T)$ //количество концептов

6: **While** $i < C$ **do**

7: $M_{real} = count(clusters[i])$

8: $L = M_{desired} - M_{real}$ //Количество слов, которыми следует расширить кластер

9: **if**($L > 0$) **then**

10: **for** $j = 0$ **to** L **do**

11: $w := similarTo(T[i, j])$ //получить синоним слова j

12: $w^* = embedding(w)$ //векторная репрезентация слова

13: $T[i] \leftarrow w^*$ // Добавить это слово в текущий кластер

14: **end**

15: **else**

16: $sort(clusters[i], Similarities)$

17: $C[] \leftarrow clusters[:M_{desired}]$

18: **end**

19: **end**

20: Сохранить кластеры в виде словаря эталонных концептов C .

Все векторы слов кластеризуются в N_k кластеров на основе использования любого алгоритма кластеризации с применением меры косинусного сходства в качестве метрики расстояния. В данной диссертации для реализации метода применяется алгоритм сферических К-средних [61]. Причина выбора этого алгоритма заключается в том, что он использует меру косинусного сходства для измерения расстояния между векторами, а также из-за его высокой эффективности при обработке разреженных и многомерных текстовых данных. Также, в отличие от стандартных К-средних, сферические К-средние оптимизируют расположение центров кластеров, что делает их более подходящими для построения однородных концептов [63].

Для заранее определенного значения K алгоритм итеративно назначает каждую точку данных одному из k центроидов и обновляет каждый центроид с

учетом принадлежности точек данных. Слово может быть распределено более чем в один кластер, это фактически означает, что различные значения слова будут объединяться с их синонимами.

Необходимо обеспечить минимальное количество слов в каждом кластере, представляющем концепт (шаги 6-7). Для этого полученный размер кластера проверяется и расширяется синонимами его слов до приемлемого размера (шаги 6-7). Синонимы получаются на основе самой модели встраивания, как показано далее в разделе с описанием вычислительных экспериментов.

После этого слова распределяются между кластерами в соответствии с мерой близости к центроиду кластера, затем выбираются M слов, наиболее близких к центроиду кластера (рис. 2.2). Функция сортировки описывается следующим выражением:

$$sort(C, S) = [w'_1, w'_2, \dots, w'_M], \quad (17)$$

где $C_1 = (w_1^1, w_2^1, \dots, w_M^1)$ – это набор слов в кластере, $S_1 = (s_1^1, s_2^1, \dots, s_M^1)$ – это набор оценок близости слов с центроидом, а $[w'_1, w'_2, \dots, w'_M]$ – множество слов из C , отсортированных в порядке убывания их оценок близости.

В итоге получается группа концептов (понятий), каждое из которых представлено M словами, принадлежащими одному общему понятию. Полученный набор кластеров образует словарь эталонных концептов, который представляется следующим образом:

$$C = clustering(\varepsilon) = (w_1^1, w_2^1, \dots, w_M^1, w_1^2, w_2^2, \dots, w_M^2, \dots, w_1^K, w_2^K, \dots, w_M^K), \quad (18)$$

где w_i^j – i -е слово j -го кластера.

На рисунке 2.2 показана иллюстрация полученного кластера с показателями близости каждого слова к центроиду кластера.

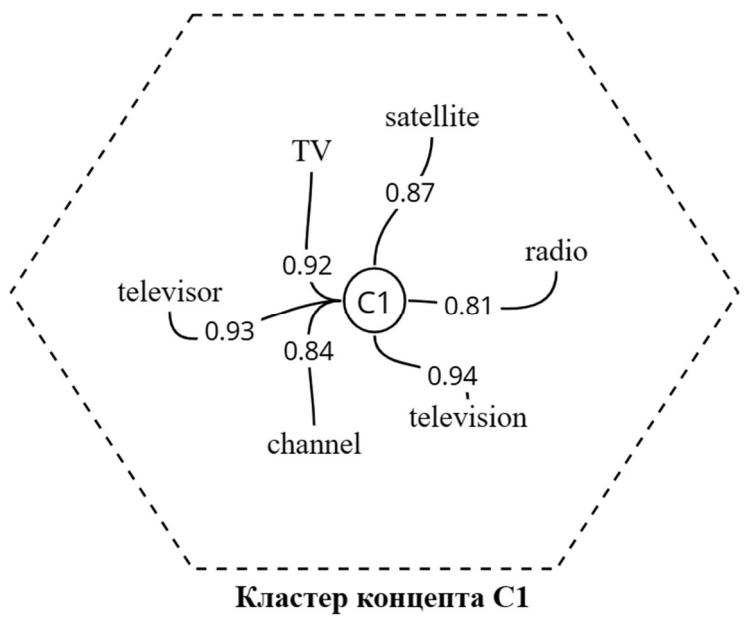


Рисунок 2.2 – Иллюстрация кластера с показателями близости каждого слова к центроиду кластера

Алгоритм векторизации документов выполняется после построения словаря эталонных концептов. Документ кодируется посредством процесса сопоставления между документом и словарем с использованием мер оценки семантической близости. Далее описана предлагаемая математическая модель представления документа.

Модель представления документа. Пусть N – количество текстовых документов, которые должны быть закодированы с использованием предложенной модели. Каждый документ описан векторами своих слов $N_i: x_{ij} \in \varepsilon, (i = 1, \dots, N, j = 1, \dots, N_i)$, где N_i – количество уникальных слов i -го документа, а x_{ij} – вектор вложения j -го слова из i -го документа.

Для рассматриваемого документа d создается вектор признаков размера k , равного количеству концептов $V^d = (c_1^d, c_2^d, \dots, c_K^d)$, где признак c_i^d выражает степень значимости i -го концепта (его вес) в документе.

$$V^d = \text{vectorization}(C, d^*, D^*, \theta) = \{c_1^d, c_2^d, \dots, c_K^d\}, \quad (19)$$

Функция векторизации включает в себя несколько операций, целью которых является вычисление значимости каждого концепта в данном документе.

Значимость концепта рассчитывается на основе применения оптимизационной процедуры, аналогичной используемым в методе *BoC* (*Bag of concepts*).

Вместо сопоставления слова документа w_d центроиду кластера, ему соотносятся M ближайших слов к центроиду. Затем максимальное значение меры близости используется в качестве показателя сходства этого слова к концепту, как это показано в выражении (20):

$$S_{c,w_d} = \max_{1 \leq j \leq M} \text{sim}(w_d, w_j^c), \quad (20)$$

где w_d – слово документа d ; w_j – j -е слово концепта c . Функция *Max* возвращает наивысшее значение оценки сходства, записанной для слова документа w_d , принадлежащего концепту c .

Конечная степень сходства концепта с документом S_c – это среднее значение степени сходства его слов со словами документа, к которому они принадлежат.

$$S_c = \frac{1}{N} \sum_{i=1}^N S_{c,w_i} = \frac{1}{N} \sum_{i=1}^N \max_{1 \leq j \leq M} \text{sim}(w_i, v_j), \quad (21)$$

где w_i – i -е слово документа, принадлежащее концепту, а *sim* это сходство между двумя векторами слов, которое вычисляется по (13).

Как и в BoC, на основе вычисленного значения сходства S_c между концептом и документом, определяется частота концепта в документе.

$$CF(c_i, d_j, D) = \frac{n_c}{\sum_k n_k}, \quad (22)$$

где n_k – общее количество концептов в документе, а n_c – количество вхождений концепта c в документ. В работе принято, что концепт содержится в документе, если значение сходства S_c концепта со словом документа превышает заданный порог (Θ).

$$g(s) = \begin{cases} 1, & S_c > \Theta \\ 0, & \text{иначе} \end{cases}, \quad (23)$$

где порог Θ , определяется экспертным путем (см. глава 4).

На рисунке 2.3 показан процесс определения наличия/отсутствия концепта C_1 в документе. Концепт C_1 присутствует в документе, так как сходство слова документа со словами концепта высокое ($S=1$).

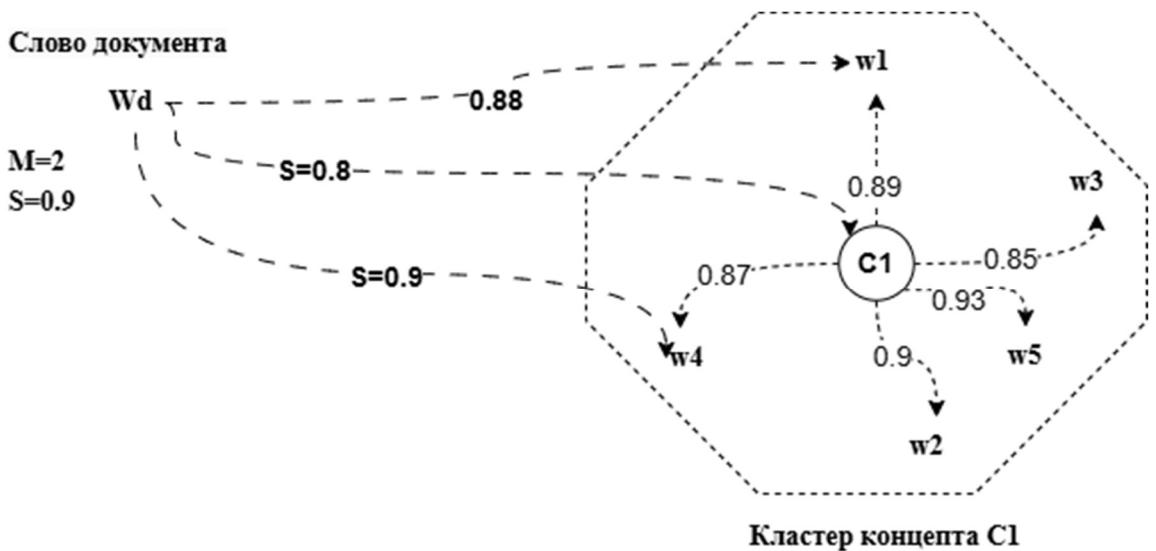


Рисунок 2.3 – Иллюстрация процесса сопоставления одного слова из документа с концептом

Помимо использования эвристической функции расчета сходства S_c для расчета частоты концептов, автор предлагает включить эту функцию в формулу взвешивания концептов. Цель состоит в том, чтобы максимально использовать рассчитанную статистику, а во-вторых, ожидается, что это повысит дискриминационную способность результирующего вектора документа. Формула определения веса концепта выглядит следующим образом:

$$BoWC_{c_i} = \frac{n_{c_i}}{\sum_k n_k} \cdot e^{S_i^j}. \quad (24)$$

Согласно анализу, представленному в пункте 1.2.4, использование функции IDF для уменьшения веса часто встречающихся концептов не подходит для концептуального представления, поскольку приводит к неверным выводам об объеме общих концептов. Поэтому автор предлагает новую функцию взвешивания, основанную на обратной частоте документа ранее установленной функцией монотонного убывания [112], цель использования которой состоит в том, чтобы уменьшить вес общих концептов на уровне корпуса текста и придать большее

значение характерным (редким) концептам. Функция задана следующим выражением:

$$f(F) = e^{-\alpha \cdot DF}, \quad (25)$$

где α – константа (по умолчанию $\alpha=1$), а DF – частота документа $DF = \frac{|\{d \in D | c_i \in d\}|}{|D|}$.

Экспоненциальная функция была выбрана так, чтобы значение f находилось в диапазоне $[0,1]$. Вдохновленная функцией IDF предложенная в этой работе функция называется экспоненциальной обратной частотой документа (Exponential document frequency EDF). Окончательная формулировка весовой функции понятия согласно $BoWC$ принимает следующий вид:

$$BoWC_{c_i} = \frac{n_{c_i}}{\sum_k n_k} \cdot e^{-\alpha \cdot \frac{|\{d \in D | c_i \in d\}|}{|D|}} \cdot e^{S_{c_i}}. \quad (26)$$

В результате метод генерирует вектор признаков $V^d = \{v_1^d, v_2^d, \dots, v_K^d\}$ для каждого заданного документа d , где v_i^j отражает важность (вес) i -го концепта, а k – количество концептов.

Функция $CF-EDF$ является модификацией функции $CF-IDF$ (14). Частота концепта CF является общим компонентом двух функций. Разница заключается в способе расчета обратной частоты документа. Чтобы прояснить этот момент, проводится сравнение между двумя функциями с точки зрения их соответствующего вывода (рис. 2.4). Предположим, что общее количество документов составляет 1000. Тогда, выход каждой функции рассчитывается для частот документа F от 10 до 999.

Отмечено, что значения функции EDF монотонно уменьшаются от 1 (для редких концептов) до 0,3 для распространенных концептов. Тогда как для IDF , при редких концептах (где концепт присутствует только в 10 документах из 1000) значение IDF становится равным 2.

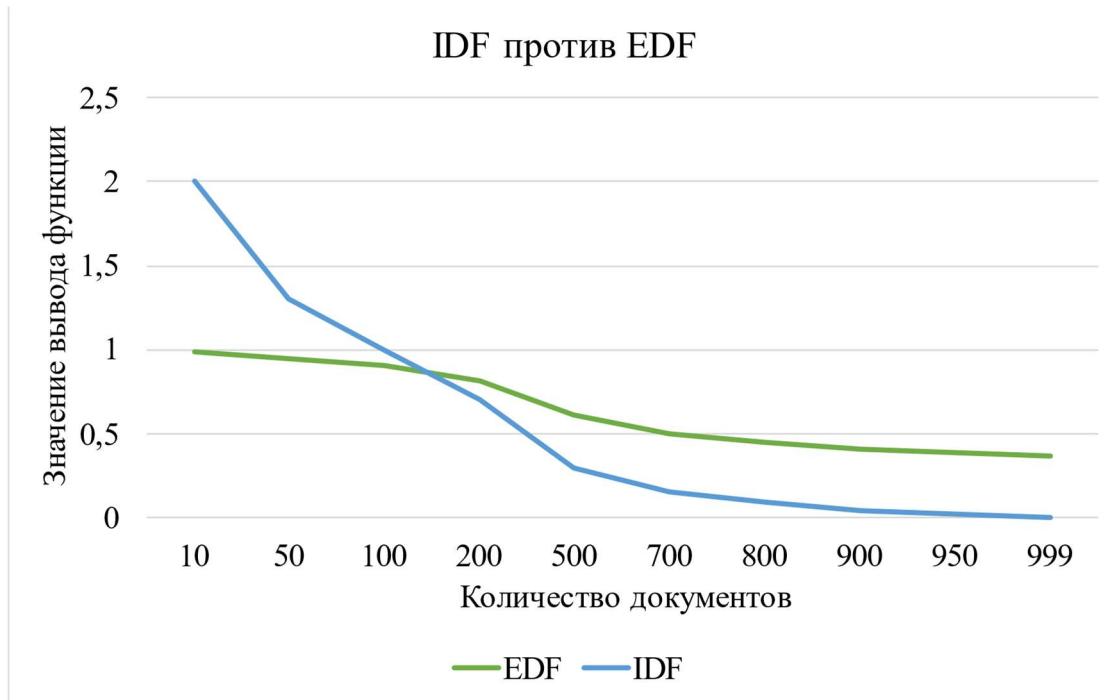


Рисунок 2.4 – Пример сравнения выходного значения каждой из двух функций, где общее количество документов равно 1000.

При наличии концепта в большинстве документов (999 из 1000) значения *IDF* становятся очень маленькими, приближаются к нулю. Это допустимо в случае вычисления частоты обратного документа терминов и неточно, когда речь идет о частоте концепта.

Концепт, даже если повторяется в большинстве документов, не обязательно должен быть одним и тем же словом. Он может быть представлен несколькими синонимами (разными терминами), относящимися к этому концепту с определенной степенью сходства. Поэтому частота встречаемости концепта на уровне коллекции документов намного выше, чем частота любого из его терминов. Это приближает значение веса концепта к нулю, что указывает на возможность ложного вывода.

С технической точки зрения, на этапе предварительной обработки документов удаляются стоп-слова, не несущие смысловой нагрузки, поскольку они являются общеупотребительными в большинстве документов. Термины, которые остаются после предварительной обработки, даже если они часто повторяются,

представляют интерес. Именно это отличает предлагаемую весовую функцию от других.

Кроме того, включение степени сходства в качестве термина в функцию определения веса концепта по мимо *CF-EDF* делает вес концепта более информативным. Единственным минусом предложенного решения является увеличение вычислительных затрат метода из-за учёта дополнительных слов со центроидом кластера при расчете веса концепта.

Далее в пункте рассматривается второе решение, которое предлагает применение фильтрации терминов на этапе построения словаря эталонных концептов. Это направлено на уменьшение шума в кластерах и позволяет использовать только вектор центроида кластера для расчета веса концепта.

2.1.2 Применение фильтрации терминов на этапе построения словаря эталонных концептов

Для повышения качества извлекаемых концептов и уменьшения зашумленности текста автор в предложенный ранее алгоритм (п. 2.1) ввел этап фильтрации терминов, который позволяет оставить только наиболее частотные термины для построения словаря эталонных концептов. Аналогичным образом документы фильтруются для исключения слов, не несущих информацию на уровне корпуса.

Выбор алгоритма взвешивания терминов. Процесс фильтрации заключается в выборе частотных слов в документе, которые более полно описывают его содержание. Для этого используется метод взвешивания терминов, позволяющий определить вес каждого слова в документе.

Алгоритмы взвешивания терминов разделены на два основных типа: интертекстуальные и внутритекстовые [113]. Интертекстуальные алгоритмы сравнивают распространение термина в данном тексте с его распространением в других текстах *TF-IDF*, *RAKE* [114]. При использовании внутритекстовых алгоритмов учитывается только внутренняя структура представленного текста (*Yake* [115], *Text Rank* [116]). Так как целью фильтрации является уменьшение

количества слов, участвующих в процессе формирования словаря эталонных концептов, то лучшим вариантом является использование интертекстуальных методов. В данном исследовании используется мера *TF-IDF*, описанная в пункте 1.2.1.

Ниже представлен алгоритм фильтрации терминов и встраивания слов документов, которые будут участвовать в процессе построения словаря эталонных концептов.

Шаг1. После обработки документов и удаления стоп-слов происходит обучение модели встраивания на самих документах. Согласно [117, 118], обучение модели встраивания текста на анализируемых данных эффективнее, чем использование модели встраивания, предварительно обученной на данных общего назначения.

Шаг2. Проводится взвешивание слов с использованием функция *TF-IDF*. Эта функция присваивает каждому слову вес, определяющий его важность в документе *TF*, взвешенную по его важности на уровне корпуса *IDF*.

Шаг3. После взвешивания терминов слова документов ранжируются в соответствии с их весами, а затем выбирает *N* верхних слов. После этого из набора отфильтрованных документов создается словарь уникальных слов.

Шаг4. Алгоритм возвращает словарь отфильтрованных слов doc_{wv} , где ключами являются слова, а значениями – их векторы, а также возвращает список документов, представленных векторами встраивания их наиболее характерных слов:

$$d_i^* = \{w_1, w_2, \dots, w_M\}, \text{where } M < N. \quad (27)$$

Фильтрация нерелевантных слов уменьшает размер входных данных алгоритма кластеризации, используемых для генерации понятий, и, таким образом, снижает его вычислительные затраты. С другой стороны, фильтрация снижает шум от получаемых кластеров, поскольку позволяет избавиться от схожих концептов. Это приводит к формированию отдельных кластеров понятий с большей выразительной силой, что, в свою очередь, позволяет представить концепт только

вектором центроида, который теперь является точным представлением набора слов, составляющих кластеры.

Это позволяет использовать либо центроид для представления кластера, либо центроид с ближайшими к нему M словами. В данном случае словарь эталонных концептов представляется следующим образом:

$$\mathcal{C} = clustering(\epsilon) = (\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_K), \quad (28)$$

где \mathbf{c}_i – вектор центроида i -го понятия.

Ниже представлен алгоритм оптимизации построения словаря эталонных концептов на основе фильтрации терминов.

Алгоритм 2.2 – Предварительная обработка текста и фильтрация документов

```

 $D = \{d_1, d_2, \dots, d_N\}$  Множество  $N$  документов
1:  $m //$  Максимальное количество слов в фильтруемом документе
    $doc_{WV} = \{\}$  //Словарь терминов
    $DEL = [] //$  пустой список документов с векторами их слов
2:  $D^* = preprocessing(D)$ 
3:  $\mathcal{D} = tokenization(D^*) //$  документ как список токенов
4:  $E = embedding(\mathcal{D}) //$  Построение векторов встраивания на основе
   собственного текстового корпуса.
5:  $\mathcal{W} = tf-idf(t, D)$ 
6:  $docTopN = top_n_words(\mathcal{D}, \mathcal{W}, m) //$  Список документов с топ-N
   ключевыми словами
7: for each  $doc$  in  $\mathcal{D}$  do:
8:    $l = len(doc)$ 
9:    $doc_{embedding} = [] //$  список векторов слов документа
10:  for each  $w$  in  $doc$ 
11:     $v = fasttext_embedding(E, w) //$  Встраивание слов
12:     $doc_{embedding}[ ] \leftarrow v$ 
13:    if  $w \in docTopN$ 
14:       $| doc_{WV} \leftarrow \{w: v\}$ 
15:    end
16:     $| DEL[ ] \leftarrow doc_{embedding}$ 
   end
End
16: Save  $DEL, doc_{WV}$ 

```

Применение предложенного алгоритма позволяет получить вектор центроида, являющийся достаточным для представления концепта при определении его степени присутствия. Представим это в виде следующего выражения:

$$S_{c_i} = \max_{0 < j < |d|} \cos(c_i, w_j), \quad (29)$$

где w_j – j-е слово документа, принадлежащее концепту (понятию); c_i – вектор центроида i-го понятия. Этот процесс упрощает сопоставление концептов со словами документа при оценке сходства между словами кластера и документом, что снижает вычислительные затраты.

На практике фильтрация терминов на этапе построения словаря эталонных концептов применяется всеми методами, использующими словарь эталонных концептов. В разделе 4 рассмотрено влияние этого этапа на эффективность разработанного метода, что подтверждает его эффективность.

В данном пункте рассмотрен метод векторизации текста «BoWC», описаны его функции и алгоритмы реализации. Представлены два варианта реализации метода, описанные как два решения. Первый включает в себя новую весовую функцию концептов *CF-EDF*, а второй касается модификации алгоритма построения словаря эталонных концептов путем фильтрации бессмысленных терминов. Учитывая полученные решения, в следующем пункте проводится анализ временной сложности предлагаемого метода.

2.2. Временная сложность разработанного метода BoWC

Важным моментом, связанным с встраиванием слов в предлагаемом методе, является то, что слово из документа иногда одновременно соответствует более чем одному понятию (мягкая атрибуция). Это связано с тем, что модели встраивания, используемые в этом исследовании, генерируют единый вектор слова независимо от его контекста, и, следовательно, нельзя окончательно судить о том, что слово соответствует одному понятию, а не другому. Таким образом, чтобы убедиться, что

концепт встречается в конкретном документе, каждое слово в документе должно быть проверено на соответствие всем концептам.

В результате процесс векторизации документа состоит из двух вложенных циклов: один для прохождения всех концептов (k концептов), а второй для выполнения сравнения между каждым концептом со словами каждого документа (n слов). Таким образом, в зависимости от размера каждого документа и количества концептов времененная сложность равна $O(kn)$. Так как в худшем случае k может быть равно n , тогда времененная сложность будет равной $O(n^2)$.

В результате применения предложенных автором модели, метода и алгоритмов число концептов значительно сокращается и, согласно экспериментальным исследованиям, не превышает 200 (см. глава 4), поэтому значение переменной k всегда несоизмеримо меньше значения переменной n , тогда в общем случае временная сложность предложенного метода становится равной $O(n)$.

Количество слов в документе можно контролировать, извлекая и используя только важные слова для представления документа, как это показано в следующем разделе.

2.3. Выводы по разделу

В данной главе описана разработка нового метода представления текста в виде семантических векторов (*BoWC*), который основан на подходе к векторизации документов путем присвоения слов документа эталонному набору концептов, извлеченных из самих текстов. Построена структура метода, подробно описан процесс его функционирования. Кроме того, представлено математическое описание используемых в методе функций, поясняющие их роль в процессе векторизации текстов.

В соответствии с этим подходом документ представляется как вектор, в котором каждый концепт имеет независимое измерение. Такое представление считается линейным преобразованием пространства слов в пространство

концептов, позволяющее управлять размером векторов. Полученное представление сохраняет возможность интерпретации. Каждый элемент вектора документа содержит вес концепта в словаре эталонных концептов. Сопоставив словарь с вектором и весами, вектор можно легко интерпретировать.

Реализация метода играет важную роль в определении достоверности концептов, так как алгоритм кластеризации концептов зависит от эффективности векторов встраивания слов и от эффективности используемого алгоритма кластеризации. В четвертой главе проанализировано влияние этих факторов на эффективность предлагаемого метода.

Метод *BoWC* опирается на словарь эталонных концептов для векторизации документов. Таким образом, на эффективность результирующей векторизации влияет качество используемого словаря эталонных концептов. Анализируемые в данном исследовании методы (*BoWC*, *BoC*, *VLAC*) используют словарь эталонных концептов, построенный на основе простого алгоритма построения концептов, основанного на группировке слов-униграмм [22, 12]. Согласно принципу распределенного представления, слова, находящиеся ближе в векторном пространстве, должны быть схожими по смыслу [19]. Поэтому логично, что одно и то же слово включается в несколько концептов при использовании распределенного представления слов. Однако это приводит к появлению зашумленных концептов, что, в свою очередь, отражается на эффективности векторов представления документов. Следующая глава посвящена решению данной проблемы

ГЛАВА 3. АЛГОРИТМ ПОСТРОЕНИЯ КОНЦЕПТОВ ПРИ РЕШЕНИИ ЗАДАЧИ КЛАСТЕРИЗАЦИИ С ИСПОЛЬЗОВАНИЕМ КЛЮЧЕВЫХ ФРАЗ

3.1. Решение задачи многозначности слова в словаре эталонных концептов

Построение концептов из текста является актуальной задачей и имеет множество применений, таких как векторизация, индексирование, классификация, визуальное описание содержимого документа и т.д. Анализируемые в данном исследовании методы (*BoWC*, *BoC*, *VLAC*) используют словарь эталонных концептов, построенный на основе простого алгоритма построения концептов, использующего группировку уникальных слов-униграмм в корпусе [22, 12]. Хотя формирование понятия на основе отдельных слов (униграмм) проще, оно считается проблематичным из-за многозначности смысла слов.

Решение этой проблемы требует устранения многозначности смысла слов путем определения их значений, соответствующих теме документа. Существует множество методов для решения данной задачи, таких как методы на основе знаний, которые используют внешние ресурсы (словари, онтологии или семантические сети) [119–121]. Другие методы зависят от статистических подходов для выявления соответствующих закономерностей в тексте с использованием методов определения семантической близости, ко-встречаемости и ассоциации.

Автор уделяет особое внимание методам, основанным только на тексте в качестве источника, потому что данные методы не имеют ограничений, свойственных другим аналогам. Для исключения этих недостатков, в данном исследовании разработан новый алгоритм построения концептов на основе применения извлечения ключевых фраз (n-грамм), вместо униграмм.

Использование ключевой фразы мотивировано тем, что она подтверждает значение слова и раскрывает многозначность его смысла, расширяя соседними словами имеющейся контекст. Если посмотреть на два концепта: *machine*, *machine*

learning, то присутствие одного только слова «*learning*» внутри кластера придает иной смысл, чем если бы в кластере присутствовало только «*machine*». К слову «*machine*» могут относиться автомобили, компьютеры и другие значения, но наличие рядом слова «*learning*» указывает на более конкретное понятие [122, 117].

Однако, сами по себе эти сочетания слов (фразы) не решают проблему полностью, поскольку значение фразы может меняться от одного контекста к другому. Поэтому автор считает, что алгоритм извлечения ключевых фраз должен обеспечивать подбор подходящих ключевых фраз на основе критерия, обеспечивающего смысловую привязку фраз к общему контексту. В соответствии с этим предложено применить алгоритм *FBKE* [123–125] (Frequency and Bert-based Keyword Extraction), использующий векторы встраивания на основе контекста для представления ключевых фраз, что позволяет вычислить значения сходства фразы с контекстом документа [126].

FBKE включает два этапа: на первом выбираются кандидаты *n*-грамм на основе их частоты встречаемости в тексте документа; на втором этапе кандидаты, после присвоения им веса, ранжируются в соответствии с их близостью к контексту документа.

Поскольку *n*-граммы извлекаются с помощью скользящего окна, то могут возникнуть некоторые ключевые фразы, которые содержат нежелательные части, например, начинаются с артикля или появляются нерелевантные глаголы или наречия [127]. Для устранения этого недостатка используется лингвистический анализатор, определяющий части речи для каждой ключевой фразы и затем отфильтровывающий или отсекающий неподходящие фразы.

На основе вышеизложенного в следующем пункте описана разработка алгоритма извлечения ключевых фраз на основе парсера, а затем представлен основанный на нем алгоритм построения концептов. В конце следующего пункта описана модификация функции взвешивания концептов метода *BoWC* на основе предложенных алгоритмов.

3.2. Алгоритм извлечения ключевых фраз на основе применения парсера

Извлечение ключевых слов является фундаментальной задачей при обработке и анализе текстов на естественном языке и включает в себя поиск наиболее релевантных и значимых слов или фраз в заданном тексте. Парсеры играют ключевую роль в этом процессе, анализируя синтаксическую структуру текста и помогая выявлять ключевые фразы, представляющие важные понятия или темы.

1. *Синтаксический анализ структуры для извлечения ключевых фраз.* Для выполнения извлечения ключевых фраз с использованием синтаксического анализа структуры, можно сосредоточиться на конкретных синтаксических единицах, таких как именные фразы (NPs) [18] и глагольные фразы (VPs). Эти фразы часто являются хорошими кандидатами на роль ключевых слов, так как обычно содержат важную информацию о субъекте, объекте или действии в предложении. Например, рассмотрим предложение: "The chatbot app answers user questions quickly." С помощью синтаксического анализа структуры будут выделены следующие фразы:

- Именные фразы (NPs): " The chatbot app", "user questions";
- Глагольная фраза (VP): "answers user questions quickly".

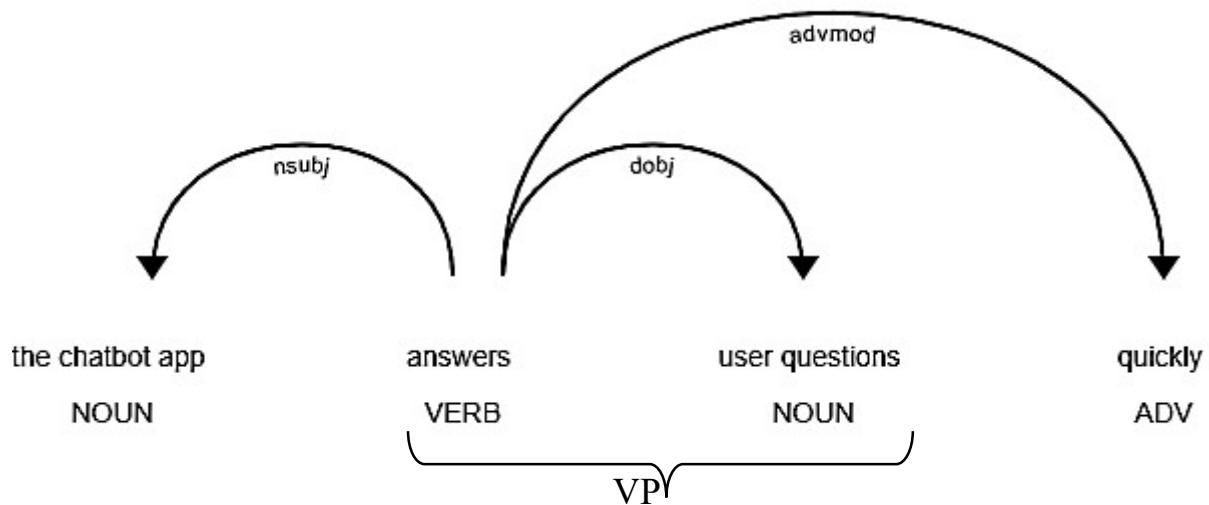


Рисунок 3.1 – Результат синтаксического анализа (дерева зависимостей) текстового предложения с использованием парсера

Из извлеченных фраз можно выделить следующие ключевые слова: "chatbot app" (приложение чат-бот); "user questions" (вопросы пользователей); "отвечает на вопросы пользователей" (перепрыгивает через), как существенные компоненты данного предложения.

2. Синтаксический анализ зависимостей для извлечения ключевых фраз.

Синтаксический анализ зависимостей помогает определить отношения между субъектом, глаголом и объектом, а также другие существенные синтаксические зависимости, которые вносят вклад в общий смысл предложения. Ключевые фразы могут быть извлечены из этих зависимостей, причем, предпочтение отдается словам, несущим значительный семантический вес и играющим важные роли в структуре предложения [34].

Исходя из предположения, что ключевая фраза обычно является существительным или именной фразой [128–130], предлагается алгоритм оптимизации извлечения ключевых фраз на основе оценки частоты их встречаемости в тексте, отличающийся применением функции парсера для маркировки части речи (англ. Part-of-speech tagging), для фильтрации фраз, не являющихся именными фразами. Это позволяет извлекать ключевые фразы с правильной грамматической структурой.

Предложенный алгоритм подробно описывает этапы рабочего процесса фильтра в виде псевдокода. Входными данными алгоритма является текст, а выходными данными является окончательный список ключевых фраз.

Сначала функция *preprocessing* обрабатывает текст для удаления ненужных символов, знаков препинания, стоп-слов и переводов текста в нижний регистр. После этого текст анализируется парсером, который, в свою очередь, возвращает предложения, составляющие текст, а также компоненты и атрибуты каждого предложения, включая именные фразы, глагольные фразы и именованные сущности.

Фразы в их исходной форме не подходят в качестве ключевых фраз, так как они могут содержать артикли или быть слишком длинными, поэтому они обрабатываются без артиклей. Длинные словосочетания и фразы, которые не

начинаются с существительного или прилагательного, исключаются из рассмотрения. В итоге получается список именных фраз на уровне текста, которые используются в качестве фильтра для ключевых фраз.

Алгоритм 3.1 – алгоритм фильтрации ключевых слов с помощью парсера

```

1 Ввод текст
2 Вывод список ключевых фраз KW_FNP
3 T* = preprocessing(T) //удаление ненужных символов, знаков препинания,
  стоп-слов и перевод текста в нижний регистр.
4 S = parsing(T*) // Разбор текста на список предложения S и их атрибуты
6 NP ← get_NP_NE(S)//сохранение именной фразы и сущности в списке
9 // преобразования именных фраз в ключевые фразы
10 KW_FNP = [] //Список для хранения отфильтрованных именных фраз
11 Foreach np in NP do
12   | if len(np) == 1:
13     |   | if np[0].pos_ in ['NOUN', 'PROPN', 'ADJ']:
14       |   |   // первое слово np – существительное, имя собственное или
          |   |   прилагательное
14       |   |   KW_FNP.append(np.text.strip())
15     |   | elseif 1 < len(np) < 6: // если длина np не превышает 6 слов
16       |   |   | if np[0].pos_ in ['DET']: //первый термин – это артикль
17         |   |   |   KW_FNP.append(np[1:].text) // удалить артикль
18       |   |   | Else
19         |   |   |   KW_FNP.append(np.text.strip())
20   |   | Else // длина np превышает 6 слов
21     |   |   | root_deps = np.root.dep_
22     |   |   | compound = [t.text for t in np.children if “compound” in root_deps]
22     |   |   |   KW_FNP.append(np.root.text + " " + compound) //Сохранить корень
          |   |   |   фразы или составное существительное
23   |   | End if
24 End for
25 Return KW_FNP
```

Если ключевая фраза-кандидат соответствует одной из именных фраз, она сохраняется. Если именная фраза является частью фразы-кандидата, в окончательном списке сохраняется именная фраза, так как она имеет более полную структуру. Этот процесс способствует повышению точности алгоритма извлечения ключевых фраз, поскольку он гарантирует, что фразы-кандидаты представляют собой не просто последовательность слов, а устоявшуюся фразу в однородном контексте.

В разделе 4 производительность алгоритма протестирована на наборе стандартных данных, и подтверждена эффективность предложенного решения для повышения точности алгоритма извлечения ключевых фраз. В следующем пункте описывается разработка алгоритма построения концептов, а затем исследуется применение этого алгоритма для улучшения методов векторизации документов, использующих словарь эталонных концептов.

3.3. Алгоритм построения концептов на основе извлечения ключевых фраз

Алгоритм построения концептов включает несколько основных этапов. Сначала из каждого документа извлекаются ключевые фразы, затем среди них выявляются уникальные ключевые фразы, и каждая из них представляется вектором встраивания. После чего применяется алгоритм кластеризации к векторам фраз, который формирует набор кластеров, каждый из которых содержит похожие ключевые фразы. На основе информации из пункта 1.2.4, концепт представляет собой кластер похожих ключевых слов, которые на практике представлены вектором центроида кластера.

Алгоритм построения концептов отличается от основного, используемого в методах векторизации текста, таких как *BoWC*, *BoC*, *VLAC*, следующими аспектами:

- использованием n-грамм (ключевых фраз) вместо отдельных слов (униграмм);
- включением алгоритма *FBKE* для оценки значимости ключевых фраз в контексте документа с учетом статистической контекстной информации при их взвешивании;
- интеграцией отфильтрованных ключевых фраз на основе лингвистического анализа, выполняемого парсером, для уменьшения уровня шума в кластерах представляющих концептов и повышения их однородности.

Алгоритм начинается с предварительной обработки документов. Далее следует этап извлечения ключевых фраз-кандидатов с использованием алгоритма *FBKE*. При этом, извлекаются различные типы *n*-грамм, а затем частота каждого типа *n*-грамм рассчитывается по следующей формуле:

$$TF_{n\text{-}gram} = \frac{n_t^i}{\sum_k n_k^i}, \quad (30)$$

где n_k относится к общему количеству n-грамм определенного типа (с элементами i), а n_t представляет количество вхождений токена t в документ.

Алгоритм 3.2 – Алгоритм построения словаря эталонных концептов на основе извлечения ключевых фраз

- ```

1: Ввод $D = \{d_1, d_2, \dots, d_N\}$ //множество N документов
2: Вывод CD //словарь эталонных концептов
3: $D^* = preprocessing(D)$
4: $CKW = extract_candidates_FBKE(D^*)$ // извлечения ключевых фраз-кандидатов на основе частоты их встречаемости в документе
5: $KW_FNP = Filtering(D^*)$ //алгоритм извлечения ключевых фраз на основе парсера
6: Foreach candidate in CKW do:
7: | If candidate in KW_FNP:
8: | | KWS [] \leftarrow candidate // добавить кандидата в список
9: | Else // проверка наличия общих терминов между кандидатом и именной фразой
10: | | Foreach np in FNP do
11: | | | If np \cap candidate $\neq \emptyset$ do:
12: | | | | KWS [] \leftarrow np // добавить pr в список
13: | | | | End
14: | | | End
15: | | End
16: CKW = sim(D^*, KW_FNP)//Оценка семантической близости фраз к документу
17: CKW = ranking(D^*, KW_FNP)// ранжирование выбранных ключевых фраз на основе их частоты и семантической близости по формуле (31)
18: UKWS = set(KWS) // Выбор уникальных фраз для кластеризации
19: concepts = Clustering(UKWS) // кластеризация ключевых фраз
20: Сохранить кластеры в виде словаря C.

```

На основании рассчитанной частоты составляются и выбираются первые N ключевых фраз-кандидатов. Однако, как упоминалось ранее, не все полученные фразы подходят для использования в качестве концептов, поскольку они не

подчиняются грамматическим правилам, регулирующим границы фразы. Для устранения этого недостатка применяется алгоритм фильтрации ключевых фраз с использованием парсера (п. 3.2), описанный в предыдущем пункте.

Фильтрация применяется к ключевым фразам-кандидатам перед созданием их векторов с использованием методов контекстного векторного представления SBERT, поскольку последний процесс занимает много времени. После этого ключевые фразы взвешиваются согласно следующему выражению:

$$rel(d, c^j) = 2 \cdot \frac{tf_{c^j} \cdot S_{norm}^{j,d}}{tf_{c^j} + S_{norm}^{j,d}}, \quad (31)$$

где  $S_{norm}^{j,d} = S^j \cdot e^{S^j / |c^j|}$  – нормализованное значение сходства кандидата  $c^j$  документа  $d$ .  $S^j$  – косинусное сходство  $j$ -го кандидата с документом.  $tf$  – частота ключевых фраз.

После применения предыдущих шагов получается список ключевых фраз наиболее релевантных контексту документа. Результирующие ключевые фразы и их векторы используются для создания словаря эталонных концептов в следующем формате:

$$C = (kp_1^1, kp_2^1, \dots, kp_M^1, kp_1^2, kp_2^2, \dots, kp_M^2, \dots, kp_1^K, kp_2^K, \dots, kp_M^K), \quad (32)$$

где  $kp_i^j$  –  $i$ -я ключевая фраза  $j$ -го кластера. На рисунке 3.1 показана форма кластера, представляющая концепт, состоящего из ключевых фраз, упорядоченных по степени их близости к центру кластера.

Проведение извлечения ключевых фраз при построении словаря эталонных концептов уменьшает шум в кластерах, что делает вектор центроида кластера точным представлением концепта (понятия), достаточным для вычисления сходства с документом. Данный алгоритм обеспечивает формирование более однородных кластеров концептов, так что ключевые фразы, входящие в кластер, соответствуют одному конкретному концепту. Перечисленные преимущества снижают вычислительные затраты на процесс кластеризации и сопоставления фраз

документа со словарем эталонных концептов, а также позволяют повысить эффективность алгоритмов обработки и анализа текстов на естественном языке.

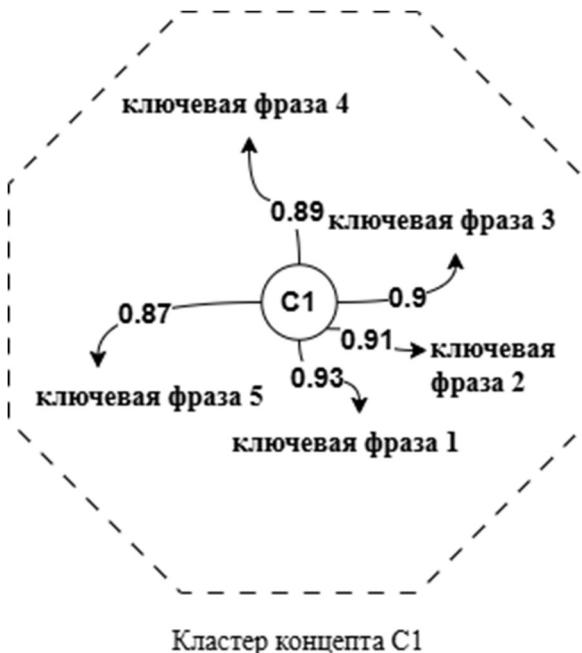


Рисунок 3.2 – иллюстрация кластера концепта с показателями близости каждой фразы к центроиду

В данном пункте представлено описание разработки алгоритма построения концептов. Так как в данном исследовании основное внимание уделяется методам векторизации документов, в следующем пункте описано применение алгоритма извлечения ключевых фраз при построении концептов для модификации функции взвешивания концептов метода *BoWC*.

### 3.4. Модификация функции взвешивания концептов метода BoWC

Применение алгоритма извлечения ключевых фраз на этапе построения концептов позволяет модифицировать весовую функцию *CF-EDF* метода *BoWC*, выраженную формулой (27).

В формуле (27)  $S_{c_i}$  является средней степенью близости слов документа с концептом, которому эти слова принадлежат. Однако, поскольку ключевые фразы документа выбираются на основе частоты их встречаемости в тексте и их семантической близости с контекстом документа, автор предлагает заменить

термин  $S_{c_i}$  на средние веса ключевых фраз документа, которые относятся к концепту, заданный следующей формулой:

$$S_i^j = \frac{1}{N} \sum_{n=1}^N \text{rel}(c_i, kw_n^j), \quad (33)$$

где  $N$  – количество ключевых фраз документа ( $j$ -го), появившихся в текущем концепте  $c_i$ . Полученный вес выражает отношение между ключевой фразой и документом, с одной стороны, и между ключевой фразой и понятием, с другой. Окончательная формулировка весовой функции понятия согласно *BoWC* принимает следующий вид:

$$\text{BoWC}^*_{c_i} = \frac{n_{c_i}}{\sum_k n_k} \cdot e^{-\frac{|\{d \in D | c_i \in d\}|}{|D|}} \cdot e^{S_i^j}. \quad (34)$$

На рисунке 3.2 представлена иллюстрация процесса сопоставления одной ключевой фразы из документа с концептом.  $C_1$  – кластер, представляющий концепт в словаре эталонных концептов, который содержит набор ключевых фраз и значений их степеней семантической близости к центроиду кластера.

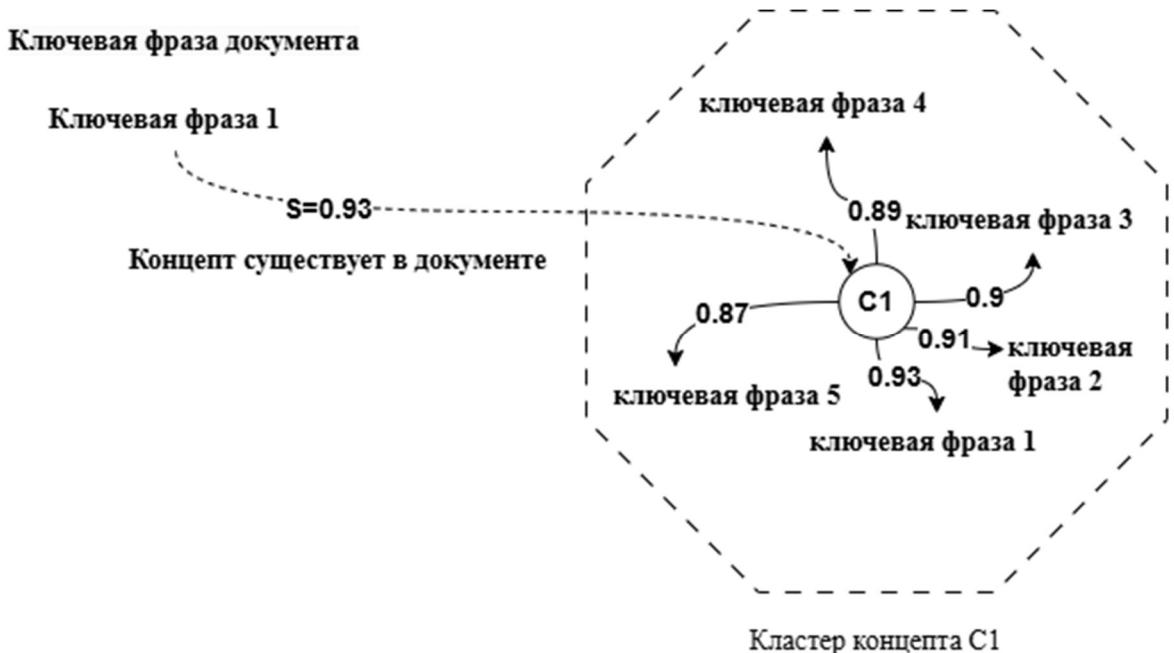


Рисунок 3.3 – Иллюстрация процесса сопоставления одной ключевой фразы из документа с концептом

Ключевая фраза 2 – это одна из фраз документа, для которого должна быть построена векторная репрезентация. Таким образом, концепт  $C_1$  присутствует в документе, так как сходство фразы документа со центроидом концепта высокое ( $S=0,93$ ). Путем сопоставления всех фраз документа со словарем эталонных концептов выявляются концепты, присутствующие в документе, а также рассчитывается вес каждого из них согласно модифицированной функции (35).

В результате этого процесса метод *BoWC* генерирует вектор признаков  $V^d$  для каждого заданного документа  $d$ , где  $v_i^j$  отражает важность (вес)  $i$ -го концепта, а  $k$  – количество концептов.

$$V^d = \text{vectorization}(C, d^*, D^*) = \{v_1^d, v_2^d, \dots, v_K^d\}. \quad (35)$$

Таким образом, получаются два разных варианта реализации метода *BoWC*. Первый – только за счет использования ключевых фраз вместо униграмм при сохранении той же функции взвешивания концептов *CF-EDF*. Второй вариант реализации предполагает использование ключевых фраз с модифицированной функцией взвешивания концептов (32). В четвёртом разделе вычислительных экспериментов тестируются и сравниваются все варианты метода *BoWC* с другими базовыми методами.

Предлагаемая в этом пункте модификация алгоритма позволяет включать дополнительную контекстную информацию о концептах, делая результирующий вектор документа более выразительным и менее неоднозначным. Это, в свою очередь, положительно отражается на производительности алгоритмов интеллектуальной обработки и анализа текстов.

В процессе векторизации документа, описанном в предыдущих пунктах, документ представляется его наиболее выразительными ключевыми фразами  $d_i = \{kp_1, kp_2, kp_3, \dots, kp_N\}$ . Имеется возможность использования данного результата независимо от алгоритма векторизации документов в различных задачах, таких как: визуализация документов; индексирование документов; улучшение профиля пользователя на основе контента и т.д.

### 3.5. Выводы по разделу

В данной главе описана разработка набора алгоритмов и функций, которые позволяют эффективно решать задачи обработки и анализа текстов на естественном языке. Предложен алгоритм построения концептов на основе извлечения ключевых фраз. Построена структура алгоритма и подробно описан процесс его работы. Принцип работы алгоритма основан на использовании ключевых фраз вместо слов-униграмм для решения проблемы многозначности слова в словаре эталонных концептов.

Для извлечения ключевых фраз автор разработал алгоритм извлечения ключевых фраз на основе парсера. Приведено описание данного алгоритма, представлено математическое описание его функций, используемых при оценке и определении весов релевантных ключевых фраз. Алгоритм позволяет отфильтровать ключевые фразы с неправильной языковой структурой. Применение этого алгоритма при построении словаря эталонных концептов способствует уменьшению зашумленности концептов и повышению их однородности.

Использование предложенных алгоритмов позволило модифицировать функцию определения весов концептов метода *BoWC*, включив в нее дополнительную контекстную информацию о них, что сделало результирующий вектор документа более выразительным и менее неоднозначным.

Сделанные выводы подтверждаются результатами вычислительных экспериментов, описание которых представлено в четвертой главе. Проведена оценка эффективности предложенных алгоритмов и их вклада в повышение точности решения задач интеллектуальной обработки и анализа текста.

## ГЛАВА 4. РАЗРАБОТКА ПРОГРАММНОГО ПРИЛОЖЕНИЯ И ПРОВЕДЕНИЕ ВЫЧИСЛИТЕЛЬНЫХ ЭКСПЕРИМЕНТОВ

В данной главе диссертации представлено описание разработки программного приложения, а также подготовка и реализация серии вычислительных экспериментов по оценке эффективности метода *BoWC*, разработанных алгоритмов и функций, при решении задач интеллектуального анализа текстов. Первая серия экспериментов включает в себя настройку параметров метода *BoWC*, определение наиболее подходящего классификатора, определение соответствующего порога сходства концептов для построения их словаря, а также сравнение и выбор наиболее подходящих моделей встраивания слов для реализации метода.

Вторая серия включает в себя эксперименты по оценке эффективности метода *BoWC* с применением алгоритма фильтрации терминов при построении словаря эталонных концептов. Также сравнивается производительность разных версий *BoWC* между собой (с применением алгоритма фильтрации терминов и без него), а также с другими методами решения задач классификации и кластеризации.

В пункте 4.6 предложенный алгоритм извлечения и фильтрации ключевых фраз тестируется независимо от метода, используемого в задаче извлечения ключевых фраз. В пункте 4.7 анализируется структура полученных векторов и выделяется свойство интерпретируемости.

### **4.1. Разработка компонентной архитектуры программного Веб-приложения**

Построена компонентная архитектура программного приложения, реализующего разработанные модели и алгоритмы интеллектуального анализа текстов, включая векторизацию текстов, извлечение ключевых фраз и концептов при обработке и анализе текстов на естественном языке [30, 31].

Программная система реализована на языке *Python* версии 3.12 с использованием таких основных библиотек как: gensim, scikik-learn, Fastapi, Streamlit [131, 132, 70, 66]. Пользовательский интерфейс реализован с

использованием HTML, CSS и JavaScript [133]. Использовался ноутбук Asus с процессором Intel Core i7 1,99 ГГц.

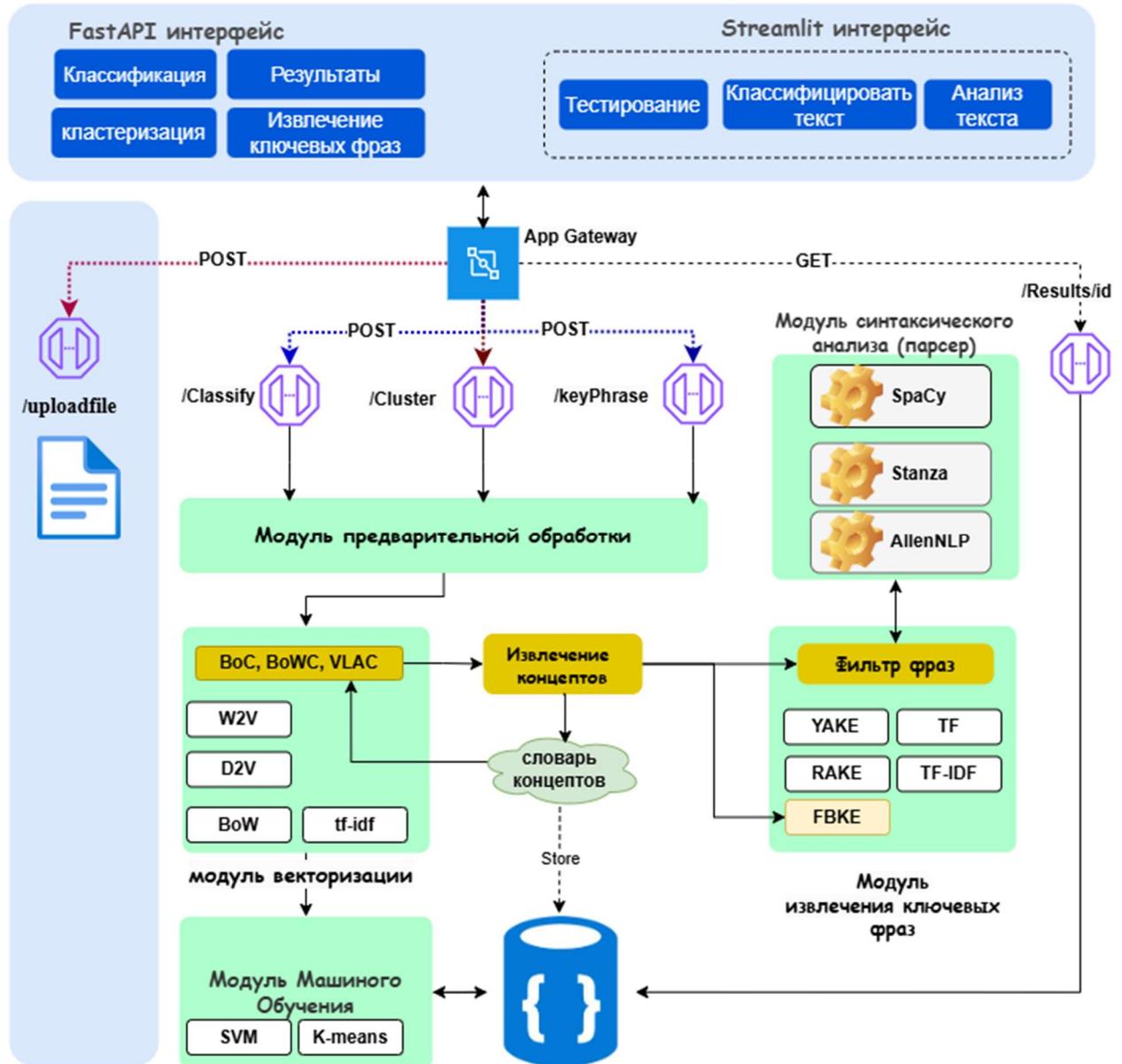


Рисунок 4.1 – Компонентная архитектура программного приложения

Клиентское приложение – это пользовательский интерфейс, написанный на HTML и JavaScript, подходящий для взаимодействия с приложением. Секции панели управления в интерфейсе включают в себя несколько ключевых элементов, каждый из которых соответствует определенной операции машинного обучения, такой как классификация, кластеризация и извлечение ключевых фраз.

Пользователи могут загружать CSV-файлы, содержащие текстовые данные, а также использовать выпадающие списки для выбора классификаторов и метод векторизации, эти параметры отправляются из клиентского интерфейса через тело

запроса. Результаты или сообщения об ошибках отображаются в специальной области для показа результатов.

Приложение также имеет интерактивный интерфейс для операций машинного обучения в реальном времени, разработанный с использованием библиотеки *Streamlit* на Python [131]. Оно взаимодействует с бэкендом *FastAPI* [132, 134] для упрощения обмена данными, позволяя пользователям визуализировать обучение моделей и предсказания в реальном времени. Кроме того, приложение предоставляет инструменты для визуализации результатов, проведения оценок и сравнения различных моделей.

Серверное веб-приложение разработано с использованием *FastAPI*, который является одним из самых быстрых и популярных веб-фреймворков для создания *API* на *Python*. *FastAPI* имеет несколько ключевых преимуществ для задач машинного обучения. Он обеспечивает высокую производительность благодаря асинхронной обработке запросов, что особенно важно для работы с большими данными. Гибкость и простота *FastAPI* делают его отличным выбором для создания масштабируемых и надежных приложений в области машинного обучения. На рисунке 4.1 показана компонентная архитектура приложения, в которой реализованы следующие конечные точки (англ. *end points*):

`POST /uploadfile/`: загрузка CSV-файлов из операционной системы и сохранение данных в MongoDB.

`POST /classify/`: выполнение задачи классификации на текстовом наборе данных с использованием выбранного классификатора и метода векторизации, а также отображение результатов классификации на интерфейсе.

`POST /cluster/`: выполнение задачи кластеризации на текстовых данных с использованием выбранных алгоритмов.

`POST /keyPhrases/`: извлечение ключевых фраз из входного текста или текстового набора данных.

`GET /results/`: получение информации о тестах и результатах из базовых данных в базе данных.

Приложение использует *MongoDB*, которая является NoSQL базой данных, известной своей масштабируемостью и способностью управлять неструктуризованными данными [135]. База данных *MongoDB* используется для хранения данных, таких как загруженные текстовые документы, метки классификации, параметры моделей, словарь эталонных концептов и результаты. Синергия между *FastAPI* и *MongoDB* делает этот выбор идеальным для приложений, требующих обработки и управления данными в реальном времени, что позволяет разрабатывать надежные решения в области машинного обучения.

При загрузке текстового файла, активируется конечная точка «/uploadfile», которая извлекает файл из операционной системы и преобразует его в формат *DataFrame* для удобства обработки. Кроме того, создается копия загруженного файла, которая сохраняется в базе данных для последующего использования.

Для классификации текстов в загруженном наборе данных необходимо выбрать тип классификатора и метод векторизации текста через интерфейс, а затем нажать кнопку "Классификация". Это действие инициирует процесс предварительной обработки, который включает удаление стоп-слов и нежелательных символов, а также преобразование текста в строчные буквы. После завершения предварительной обработки вызывается выбранный модуль векторизации. Этот модуль преобразует тексты в подходящие числовые векторы и отправляет их в модуль машинного обучения. Модуль машинного обучения выполняет алгоритм классификации с использованием указанного классификатора и отправляет результаты вместе с копией векторов обратно в базу данных. По окончании выполнения автоматически запрашиваются результаты с помощью GET-запроса к базе данных, чтобы отобразить окончательные результаты для пользователя.

Если выбранный метод векторизации опирается на словарь концептов, как в *BoWC*, *BoC* и *VLAC*, перед векторизацией выполняется дополнительный шаг – извлечение концептов и построение словаря концептов. Этот процесс осуществляется через совместный вызов модуля извлечения ключевых фраз и лексического анализатора.

В данном пункте описана разработку компонентной архитектуры программного приложения, имитирующего работу части системы искусственного интеллекта и машинного обучения для обработки и анализа текстов на естественном языке, а также позволяющего проводить вычислительные эксперименты для оценки качества алгоритмов, предложенных автором.

В следующем пункте описаны вычислительные эксперименты, которые были выполнены с использованием описываемого программного приложения. Перед проведением вычислительных экспериментов описываются используемые наборы данных и критерии, применяемые для оценки точности методов при решении задач интеллектуального анализа текстов.

#### **4.2. Задание настроек вычислительных экспериментов**

**Наборы текстовых данных.** Для проведения экспериментов используются 5 англоязычных стандартных наборов данных, описанных в таблице 4.1 и рисунке 4.2. Набор данных *BBC* [136] включает 2225 документов, полученных с веб-сайта новостей *BBC*, охватывающих новостные истории по пяти различным тематическим областям за период 2004-2005 гг. Набор данных *Reuters* (*RE*) состоит из статей, полученных из новостной ленты *Reuters*. В данном исследовании использовалось разделение R8 набора данных *Reuters*, включающее в себя общее количество – 8491 документ. Набор данных *20Newsgroups* (*20NG*) состоит из 18821 документа, классифицированных на 20 различных категорий новостных групп, обеспечивая примерно равномерное распределение между категориями.

Набор данных *OHSUMED* (ОН) является широко используемым набором данных в области информационного поиска и классификации текстов. Он состоит из коллекции медицинских аннотаций базы данных *MEDLINE*, которая охватывает различные темы, связанные со здоровьем и медициной. Набор данных содержит более 350000 документов, каждый из которых связан с набором соответствующих терминов *MeSH* (медицинские заголовки предметов). Исследователи часто используют набор данных *OHSUMED* для оценки и сравнения различных алгоритмов информационного поиска и классификации текстов. Он предоставляет

реалистичную и сложную платформу для выполнения задач, таких как классификация документов, кластеризация документов и ранжирование по релевантности. Набор данных является общедоступным и был предварительно обработан для удаления любой идентифицирующей информации. В данном исследовании использовалось разделение, содержащее 5380 документов из набора данных *OHSUMED*.

Набор данных *WebKB* включает веб-страницы, полученные из различных разделов информатики, собранные в рамках проекта World Wide Knowledge Base, проводимого группой изучения текста CMU [136]. Эти веб-страницы разделены на семь отдельных классов, включая студентов, преподавателей, сотрудников и т.д. Для этого исследования использовалась предварительно обработанная версия набора данных *WebKB*, состоящая из четырех различных классов, в общей сложности 4199 документов.



Рисунок 4.2 – Распределение документов и категорий в наборах данных

Помимо количества категорий и количества документов, содержащихся в каждом наборе данных, в таблице 4.1 указано среднее количество слов документа для каждого набора данных и процент документов, содержащих более 10 слов с векторами (по модели векторного встраивания *GloVe*).

Таблица 4.1 – Обзор статистики наборов данных

| Набор<br>данных | Кол.<br>Документов | Кол.<br>Классов<br>(категорий) | Среднее<br>количество<br>словарных слов<br>(in-vocabulary) на<br>документ | Кол. документов,<br>содержащие<br>более N слов с<br>векторами |
|-----------------|--------------------|--------------------------------|---------------------------------------------------------------------------|---------------------------------------------------------------|
| BBC             | 2225               | 5                              | 207                                                                       | 100%                                                          |
| RE              | 8491               | 8                              | 66                                                                        | 98.03%                                                        |
| OH              | 5380               | 7                              | 79                                                                        | 75.62%                                                        |
| 20NG            | 18821              | 20                             | 135                                                                       | 97.81%                                                        |
| WebKB           | 4199               | 4                              | 98                                                                        | 95.77%                                                        |

**Метрика оценки классификации.** F-мера – одна из распространенных метрик для оценки успешности классификатора [137]. Это среднее гармоническое между точностью и полнотой, которое определяется следующими формулами:

$$Precision_{C_i} = \frac{TP_{C_i}}{TP_{C_i} + FP_{C_i}}, \quad (36)$$

$$Recall_{C_i} = \frac{TP_{C_i}}{TP_{C_i} + FN_{C_i}}, \quad (37)$$

$$F\text{-score} = \frac{(1+\beta) \cdot Precision_{C_i} \cdot Recall_{C_i}}{\beta \cdot Precision_{C_i} + Recall_{C_i}}. \quad (38)$$

Здесь  $C$  – это метка класса,  $True\ Positive\ TP$  – это количество документов, правильно отмеченных классификатором как относящиеся к классу  $C$ , а  $False\ Positive\ FP$  – это количество документов, неправильно отмеченных классификатором как относящиеся к классу  $C$ . Между тем,  $False\ Negative\ FN$  ложноотрицательный – это количество документов, которые относятся к классу  $C$  и которые классификатор ошибочно определил как не относящиеся к классу  $C$ , а  $True\ Negative\ TN$  – это количество документов, не принадлежащих к классу  $C$ , правильно отмеченных классификатором как не относящиеся к классу  $C$ .

Коэффициент  $\beta$  настраивается так, чтобы обеспечить точность и/или полноту алгоритма кластеризации. Если  $\beta$  больше 1, точность оценивается выше при расчете, если  $\beta$  меньше 1, выше оценивается полнота.  $\beta$  по умолчанию равен 1.0.

Из-за несбалансированности и разного размера используемых наборов данных в этом исследовании использовалась микро-усредненная F1-мера [138]. Микроусреднение заключается в подсчете суммы всех истинно положительных, ложно отрицательных и ложно положительных результатов по всем классам. Затем, полученные значения используются для расчета соответствующих метрик, таких как точность, полнота и F-мера. Микро-усредненная F1-мера определяется по следующей формуле:

$$\text{Micro averaged F-score} = \frac{(1+\beta) \cdot \sum_{i=1}^C |TP_{C_i}|}{(1+\beta) \cdot \sum_{i=1}^C |TP_{C_i}| + \sum_{i=1}^C |FN_{C_i}| + \sum_{i=1}^C |FP_{C_i}|}. \quad (39)$$

**Метрика оценки кластеризации.** В данном исследовании для оценки качества результатов кластеризации используются два показателя. *Первый* применяется, когда известны метки групп, например, при тестировании производительности алгоритмов кластеризации на наборах данных, представленных в таблице 4.1.

В этом сценарии используется мера V1, которая представляет собой среднее гармоническое для различных оценок однородности и полноты [139]. Она объединяет меры однородности (Homogeneity,  $h$ ) и полноты (Completeness,  $c$ ), чтобы обеспечить общую оценку производительности кластеризации [140]. Однородность измеряет степень, в которой каждый кластер содержит только точки данных, принадлежащие одной группой, и оценивает согласованность между назначениями кластеров и истинными метками групп, а полнота измеряет степень, в которой все точки данных в группе назначены одному кластеру. V1 мера вычисляется как:

$$\text{Homogeneity } h = 1 - \frac{H(C|K)}{H(C)}, \quad (40)$$

$$\text{Completeness } c = \frac{H(K|C)}{H(K)}, \quad (41)$$

$$V\text{-мера} = \frac{(1+\beta) \cdot h \cdot c}{(\beta \cdot h) + c}. \quad (42)$$

где,  $H(C|K)$  – это условная энтропия распределения групп с учетом кластерных назначений;  $H(C)$  – это энтропия распределения групп. Коэффициент  $\beta$  устанавливается точно так же, как в формуле F1-мера. В этом исследовании было задано значение по умолчанию, равное единице. Более высокий показатель V-меры указывает на лучшую производительность кластеризации, значения варьируются от 0 до 1, где 1 представляет собой совершенный результат кластеризации.

*Второй показатель качества результатов кластеризации* используется, когда метки групп неизвестны, например, при построении словаря концептов из ключевых фраз. В этом случае используется *Индекс Дэвиса-Боулдина (Davies-Bouldin Index, DBI)* [4], который измеряет соотношение внутрикластерной вариации (within-cluster scatter) и межкластерного разделения (between-cluster separation):

$$DBI = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} \left( \frac{S_i + S_j}{d_{ij}} \right), \quad (43)$$

$$S_i = \frac{1}{n_i} \sum_{x \in C_i} \|x - \mu_i\|, \quad (44)$$

где  $S_i$  это среднее внутрикластерное расстояние, а  $d_{ij} = \|\mu_i - \mu_j\|$  – это расстояние между центроидами.

$S_i, S_j$  представляют среднее расстояние точек внутри кластеров  $i$  и  $j$  до их соответствующих центроидов. Более низкие значения для  $S_i$  и  $S_j$  указывают на более однородные кластеры. Формулу (43) можно переписать с использованием меры косинусного сходства следующим образом:

$$DBI_{cos} = \frac{1}{k} \sum_{i=1}^k (1 - \cos(d_{ij})), \quad (45)$$

В целом,  $DBI$  включает меру того, насколько плотен каждый кластер. Кластер с низким внутрикластерным разбросом можно интерпретировать как обладающий высокой однородностью, поскольку точки данных внутри этого кластера находятся близко друг к другу.

**Реализация метода *BoWC***<sup>4</sup>. Для реализации метода *BoWC* сначала проведена предварительная обработка документов, а затем построен словарь эталонных концептов по Алгоритму 2.1. Для этого были извлечены все уникальные термины в корпусе текстов и сгенерированы векторы слов с помощью модели встраивания с векторами размера 300. Получаем так называемый словарь терминов, на основе которого построен словарь эталонных концептов путем применения алгоритма сферических К-средних с порогом сходства равным 0,6 для кластеризации слов.

Чтобы проанализировать эффективность предложенного метода, несколько реализаций *BoWC* были протестированы друг против друга в двух задачах анализа текста, классификация и кластеризация. Для задачи классификации количество понятий систематически увеличивалось с 10 до 300, а пороговое значение  $\Theta$  с 0,2 до 0,8.

Несколько классификаторов были протестированы поверх методов векторизации текста, а именно: нейронная сеть *NN*, алгоритм ближайшего соседа *K-NN*, дерево решений *DT*, случайный лес *RF* и машины линейных опорных векторов *SVM* (рис. 4.3). По результатам этих экспериментов машины линейных опорных векторов *SVM*<sup>5</sup> был выбран в качестве базового классификатора благодаря быстродействию и его способности работать с небольшим количеством входных данных, что соответствует цели вычислительных экспериментов.

Для нашего метода лучше всего подходило ядро «*POLY*» для результирующих векторов. Для других методов результаты были подтверждены для всех типов ядра *SVM* (*POLY*, *LINEAR* и *RBF*), и среди них было выбрано самое высокое значение. Кроме того, в каждом экземпляре прогноза применялась 10-кратная перекрестная проверка, чтобы уменьшить вероятность переобучения данных и создания необъективных результатов.

Соответственно для задачи кластеризации предыдущий эксперимент был повторен со 100, а затем с 200 концептами с увеличением порога сходства от 0,2 до 0,8 для каждого набора данных. В качестве алгоритма кластеризации

---

<sup>4</sup> Ссылка на разные версии реализации <https://github.com/Ali-MH-Mansour/BoWC-Method.git>

<sup>5</sup> объяснено в разделе 1

использовался алгоритм К-средних. Для метода *VLAC*<sup>6</sup> прилагаемый API использовался для повторной реализации экспериментов с наборами данных из-за различий в источнике данных и критериях оценки. Было принято 30 понятий (9000 признаков) с учетом того, что большее количество понятий требует больших вычислительных затрат [22]. *VLAC* был реализован с предварительно обученным встраиванием *GloVe*, поскольку оно показало лучшие результаты среди всех других методов встраивания.

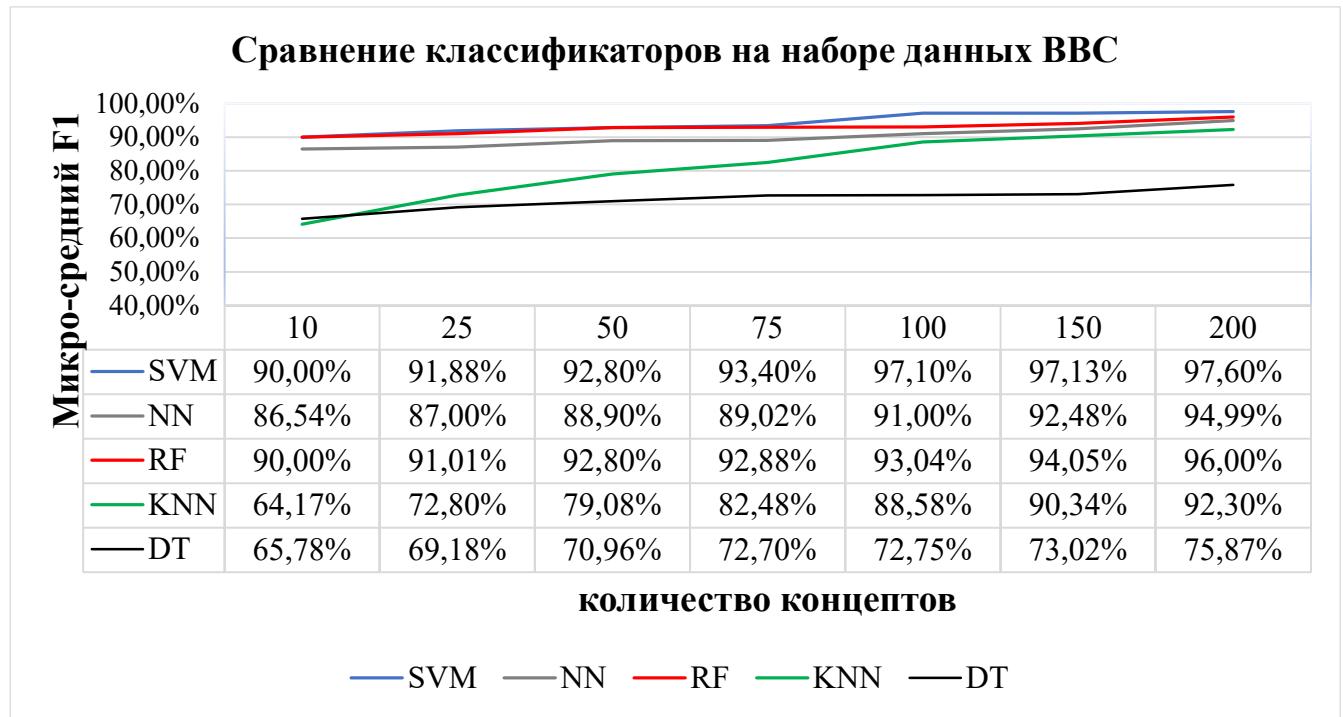


Рисунок 4.3 – Эксперименты по выбору классификатора

Учитывая, что *BoC* (Bag-of-Concepts) является частным случаем предлагаемого метода *BoWC*, он был реализован через код метода *BoWC*. Как показано в [12], *BoC* по сравнению с *TF-IDF* требует не менее 100 концептов для достижения конкурентных результатов. Таким образом, чтобы максимизировать точность *BoC*, количество понятий было установлено равным 200. Пороговое значение сходства, используемое с *BoC*, не упоминалось автором, поэтому оно было определено эмпирически и установлено как 0,4 для всех экспериментов. Для метода усредненных встраивания слов использовались предварительно обученные встраивания слов *GloVe* и self-trained (самообученные) *W2V*.

<sup>6</sup> Мы использовали реализацию, предоставленную авторами на <https://github.com/MaartenGr/VLAC>

В данном пункте описаны наборы данных и критерии оценки эффективности, используемые для тестирования разработанного метода и алгоритмов. Также описаны параметры реализации разработанного метода и других методов, с которыми он сравнивается, а также требования к вычислительным ресурсам, программным библиотекам и средствам, используемым для их реализации. В следующем пункте описаны результаты эксперимента, целью проведения которого являлась оценка эффективности предлагаемого метода и настройка его параметров.

### **4.3. Результаты эксперимента по анализу эффективности метода *BoWC* и настройка его параметров**

Результаты сравнения, предложенного метода *BoWC* с рассматриваемыми каноническими методами в задачах классификации и кластеризации документов представлены в таблицах 4.2 и 4.3. Подчеркнутые значения представляют собой самые высокие результаты для предложенного метода *BoWC*, тогда как значения, выделенные жирным шрифтом, представляют собой лучшие результаты по сравнению со всеми другими методами для одного набора данных. Во всех таблицах  $\Delta_{Best}$  показывает разницу между значением F1 и V1, достигнутым *BoWC*, и наилучшим результатом других методов, а  $\Delta_{BoC}$  отражает разницу между F1 и V1 и результатом метода *BoC*, на котором основан *BoWC*.

Как видно из результатов задачи классификации набора данных *Reuters*, предложенный метод превзошел все базовые методы со значительным различием в производительности (минимум 0,05-1%). В задаче классификации набора данных *BBC* наш метод превзошел *BoC*, *BoW* и *self-averaged W2V* с использованием всего 100 концептов, в то время как он показал схожую производительность с *TF-IDF* при использовании 200 концептов. Однако *BoWC* не смог превзойти *VLAC* (9000) на этом наборе данных. Более того, все методы достигают высокой точности классификации в этом наборе данных, что, по мнению автора, объясняется тем, что среднее количество слов в словаре относительно большое (см. таблицу 4.4). Это означает, что во всех документах достаточно слов для извлечения четкой

семантики о каждом концепте, что обеспечивает создание более качественного вектора концептов.

Учитывая, что в данном исследовании использовалась предварительно обработанная версия набора данных *WebKB*, где слова были приведены к основной форме таким образом, что они несовместимы с предложенным методом, который достиг лучшей производительности (87,77%) после TF-IDF (89,5%). Это дополнительный аспект, который отличает предложенный метод от других.

Таблица 4.2 – Результаты классификации,  
измеренные с помощью меры микро-усредненной F1

| <b>Методы (размер вектора)</b>      | <b>BBC</b>   | <b>RE</b>    | <b>ОН</b>    | <b>20NG</b>  | <b>WebKB</b> |
|-------------------------------------|--------------|--------------|--------------|--------------|--------------|
| BoWC <sub>glove</sub> (100)         | 97.01        | 92.86        | 62.997       | 69.00        | 76.25        |
| BoWC <sub>glove</sub> (200)         | 97.61        | 91.23        | 65.35        | 74.42        | 81.07        |
| BoWC <sub>w2v</sub> (100)           | 96.86        | 92.4         | 65.99        | 68.58        | 78.55        |
| BoWC <sub>w2v</sub> (200)           | 96.65        | 93.15        | 63.00        | 74.31        | 83.13        |
| BoWC <sub>FastText</sub> (100)      | 95.51        | 91.2         | 66.23        | 66.19        | 77.74        |
| BoWC <sub>FastText</sub> (200)      | 96.26        | 92.02        | 68.94        | 68.48        | 81.92        |
| Self BoWC <sub>w2v</sub> (100)      | 96.84        | 93.49        | 64.11        | 66.02        | 85.25        |
| Self BoWC <sub>w2v</sub> (200)      | 97.29        | 94.26        | 68.28        | 73.18        | 86.08        |
| Self BoWC <sub>glove</sub> (100)    | 95.8         | 92.29        | 64.78        | 84.95        | 43.44        |
| Self BoWC <sub>glove</sub> (200)    | 96.26        | 92.72        | 68.62        | <b>85.43</b> | 52.33        |
| Self BoWC <sub>FastText</sub> (100) | 96.99        | 93.65        | 68.36        | 70.07        | 86.24        |
| Self BoWC <sub>FastText</sub> (200) | <b>97.89</b> | <b>94.75</b> | <b>69.95</b> | 77.07        | <b>87.77</b> |
| TF-IDF                              | <b>97.89</b> | 94.7         | <b>74.31</b> | <b>91.49</b> | <b>89.5</b>  |
| BoW                                 | 96.98        | 94.15        | 69.89        | 83.85        | 85.56        |
| Averaged GloVe (300)                | 97.75        | 93.32        | 67.39        | 78.7         | 80.22        |
| self-averaged W2V (300)             | 91.599       | 87.81        | 56.78        | 62.37        | 83.27        |
| BOC <sub>CF_IDF</sub> (200)         | 95.48        | 90.9         | 48.8         | 69.44        | 78.26        |
| VLAC (9000)                         | 98.2         | 94.06        | 69.37        | 86.4         | 85.54        |
| $\Delta_{\text{Best}}$              | +0.0         | 0.05         | - 4.3        | - 6.06       | - 1.73       |
| $\Delta_{\text{BoC}}$               | +2.41        | +3.85        | +21.15       | +15.99       | +9.51        |

Очевидно, что в эксперименте на наборе данных *OHSUMED* метод *BoC* демонстрирует худшие результаты среди всех других канонических методов, достигая среднего микро-значения F1 48,8% по сравнению с 69,95% для предложенного метода. Это важно, потому что предложенный метод построен на основе того же подхода, что и в *BoC*, и это подтверждает эффективность введенной автором весовой функции *CF-EDF* (26) и ее способность одновременно

фиксировать семантику на уровне терминов и концептов. Также стоит отметить, что предложенный метод превзошел *BoC* по всем наборам данных на значительный процент (от 2 до 8% в зависимости от набора данных) с одновременным уменьшением количества признаков векторов.

Для *20Newsgroups*, который представляет собой относительно большой набор данных по сравнению с другими (около 20000 документов), все методы векторизации, основанные на концептах, работали плохо. Документы *20Newsgroups* разбиты на категории и подкатегории, то есть между категориями существует сходство, что требует большого количества концептов для четкого разделения категорий. По этой причине *TF-IDF* показал лучшую производительность (91%), поскольку он основан на терминах, а не на концептах. Однако, хотя метод *VLAC* работает на концептуальном уровне, его векторы (9000 признаков) содержат подробную информацию о словах концептов, и это объясняет, почему он достиг приемлемого результата (85,6%).

Таблица 4.3 – Результаты кластеризации, измеренные с помощью меры V1

| <b>Методы (размер вектора)</b>       | <b>Значения V1-мера (%)</b> |             |             |             |              |
|--------------------------------------|-----------------------------|-------------|-------------|-------------|--------------|
|                                      | <b>BBC</b>                  | <b>RE</b>   | <b>ОН</b>   | <b>20NG</b> | <b>WebKB</b> |
| BoWC <sub>glove</sub> (100)          | 72,8                        | 44,9        | 10,4        | 37,3        | 10,9         |
| BoWC <sub>glove</sub> (200)          | 73,3                        | 45          | 11,8        | 38,5        | 14,9         |
| BoWC <sub>w2v</sub> (100)            | 66,6                        | 44,5        | 16          | 34,5        | 12,8         |
| BoWC <sub>w2v</sub> (200)            | 61,6                        | 45,2        | 12,7        | 33,8        | 12,3         |
| BoWC <sub>FastText</sub> (100)       | 51,5                        | 37,9        | 14,3        | 27,5        | 16,6         |
| BoWC <sub>FastText</sub> (200)       | 62,5                        | 37,5        | <b>15,5</b> | 30,1        | 20,6         |
| Self BoWC <sub>w2v</sub> (100)       | 54                          | 50,5        | 12,3        | 21,1        | 33,7         |
| Self BoWC <sub>w2v</sub> (200)       | 53,6                        | 49,3        | 12,5        | 23,2        | 33,3         |
| Self BoWC <sub>glove</sub> (100)     | 79,4                        | 57,4        | 11,4        | 31,1        | 32,9         |
| Self BoWC <sub>glove</sub> (200)     | <b>81</b>                   | 57,4        | 11,7        | <b>39,6</b> | <b>34</b>    |
| Self BoWC <sub>FastText</sub> (100)  | 76,8                        | <b>59,1</b> | 12,9        | 31,6        | 31,8         |
| Self BoWC <sub>FastText</sub> (200)  | 77,1                        | 54,5        | 14          | 39,1        | 32,8         |
| TF-IDF                               | 66,3                        | 51,3        | 12,2        | 36,2        | 31,3         |
| BoW                                  | 20,9                        | 24,8        | 2,7         | 2,1         | 2,1          |
| Averaged GloVe (300)                 | 77,4                        | 48,1        | 10,9        | 38,1        | 21,9         |
| self-averaged w2v (300)              | 63,1                        | 55,7        | 11,5        | 32,5        | 32,4         |
| BOC <sub>CF_IDF</sub> (200) (аналог) | <u>74,3</u>                 | 52,7        | 10          | 39,4        | 8,8          |
| VLAC (9000)                          | 80,8                        | 45,6        | 11,8        | —           | 28,6         |
| $\Delta_{\text{Best}}$               | +0.2                        | +3.4        | +3.3        | +0.2        | +1.6         |

Результаты кластеризации. Кластеризация – это более сложная задача, для которой векторы должны быть выразительными, чтобы они могли четко разделять классы. Однако, в зависимости от использованных методов встраивания слов, предложенный метод *BoWC* только для 100 концептов смог превзойти все другие по большинству наборов данных, за исключением *20Newsgroups*, где требовалось 200 концептов (табл. 4.3). Эксперимент по кластеризации *VLAC* с набором данных *20Newsgroups* был исключен из-за его высоких вычислительных затрат, так как алгоритм кластеризации дал сбой из-за полной загрузки 12,72 ГБ ОЗУ.

При таком большом количестве различий между наборами данных трудно определить точную причину различий в производительности. Одной из очевидных причин может быть количество слов вне словарного запаса в документе, доля которых варьируется в зависимости от используемого метода встраивания (см. таблицы 4.1 и 4.4).

В таблице 4.4, *GloVe* (self-trained) означает, что метод встраивания *GloVe* обучена на собственном наборе данных *Reuters*, а *GloVe* (pretrained) означает что *GloVe*, предварительно обучена на большом корпусе. При использовании *GloVe* (self-trained) среднее количество слов в документе, имеющих векторы, составляет 66 слов на документ.

Процент документов, в которых более 50 слов имеют векторы, составляет 42%. Отметим, что *FastText* обученная на *Reuters* обеспечивает в среднем векторы встраивания для большего количества слов, чем другие методы встраивания.

Таблица 4.4 – Статистика векторного встраивания для набора данных *Reuters*

| <b>Встраивания слов набора данных <i>Reuters</i></b> |                     |                     |        |
|------------------------------------------------------|---------------------|---------------------|--------|
| <b>Среднее количество слов</b>                       | <b>&gt; 50 слов</b> | <b>&gt;100 слов</b> |        |
| Glove (Pretrained)                                   | 66                  | 42,00%              | 20,00% |
| Glove (Self-trained)                                 | 58,5                | 35,66%              | 14,73% |
| W2V (Pretrained)                                     | 65                  | 40,83%              | 18,12% |
| W2V (Self-trained)                                   | 66,2                | 41,62%              | 18,50% |
| Fasttext (Pretrained)                                | 65                  | 41,19%              | 18,33% |
| Fasttext (Self-trained)                              | <b>69,7</b>         | <b>45,02%</b>       | 19,97% |

Как упоминалось выше, метод *BoWC* зависит от метода встраивания слов, количества концептов и порога сходства, используемого для определения концептов документа. Ниже подробно обсуждается влияние каждого из этих параметров на производительность метода. Взаимосвязь зависимости проиллюстрирована только на одном наборе данных (*Reuters*), поскольку в целом он показывает общий характер.

**Порог сходства и тип векторов встраивания** (рис. 4.4 – 4.6). Роль порога сходства заключается в определении наличия или отсутствия концепта в документе. Когда степень сходства между вектором слова документа и вектором центроида кластера превышает пороговое значение, это указывает на наличие концепта в документе. Затем слово документа сопоставляется с остальными словами кластера концепта (M ближайших слов кластера к центроиду). Соответственно, чем выше пороговое значение, тем меньше операций сопоставления слов (рис. 4.4). Другими словами, снижается вероятность появления концепта в документе, что приводит к иррациональному взвешиванию концептов. Это влияет на эффективность результирующего представления, что, в свою очередь, проявляется в снижении показателя точности классификации/кластеризации.

#### **Результаты классификации набора данных *Reuters* со 100 концептами**

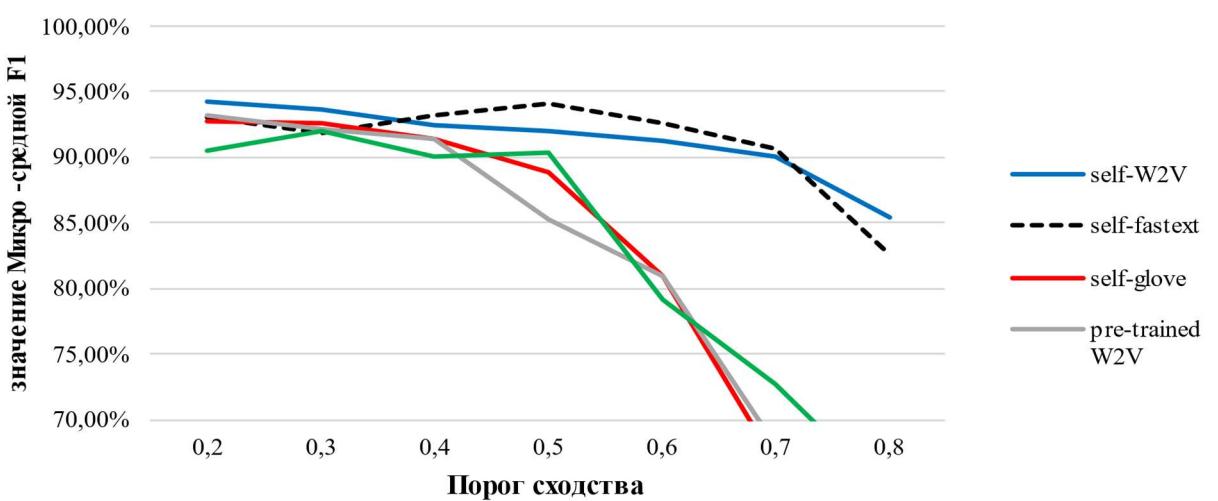


Рисунок 4.4 – зависимость точности классификации набора данных *Reuters* от пороговых значений сходства

Как правило, наблюдается, что для пороговых значений выше 0,5 точность значительно снижается по всем наборам данных. Анализируя влияние каждого метода встраивания в отдельности, заметим, что при использовании *self-W2V* и *self-FastText* метод сохраняет высокую и одинаковую точность классификации для всех пороговых значений (от 0,2 до 0,7) в соответствии с микро-усредненной мерой F1.

Более того, влияние порогового значения также зависит от метода встраивания, и это нормально, учитывая, что репрезентативная способность векторов встраивания варьируется от одного метода к другим в зависимости от характера и размера данных, на которых метод встраивания был обучен.

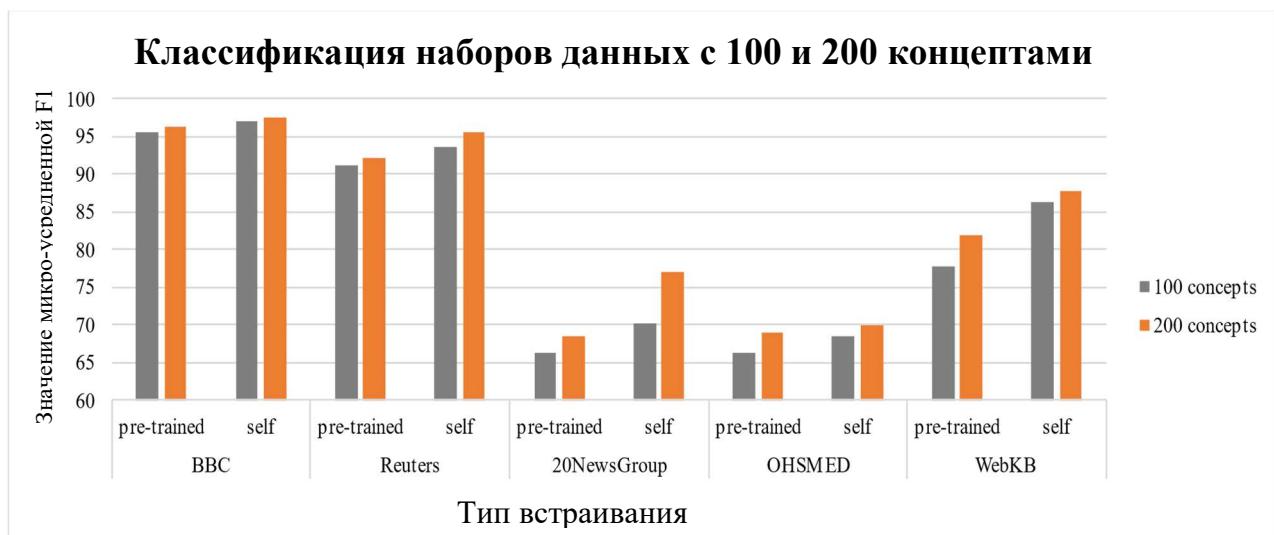


Рисунок 4.5 – Зависимость точности классификации от типа модели встраивания

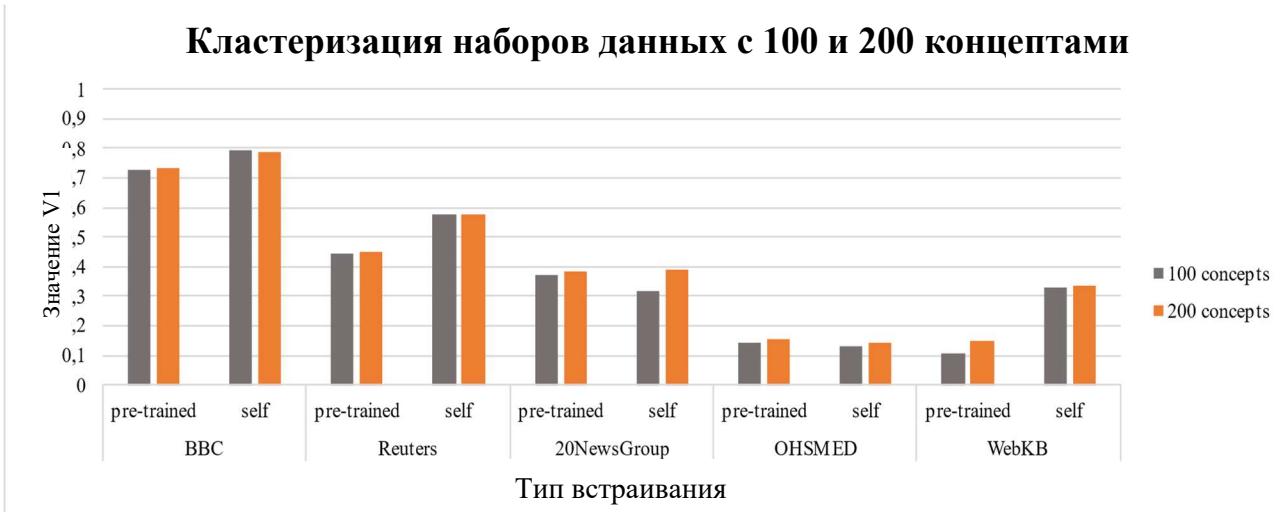


Рисунок 4.6 – Зависимость точности кластеризации пяти наборов данных от типа модели встраивания

Общая производительность *BoWC* сильно зависит от используемого метода встраивания слов (рис. 4.5 и 4.6). Эксперименты также подтверждают, что векторные представления, полученные в результате обучения модели встраивания на самих тестовых данных, работают намного лучше, чем предварительно обученные представления векторов слов.

**Количество концептов.** Для анализа взаимосвязи между количеством концептов (для разных пороговых значений) и точностью классификации (по микро-усредненной F1) было проведено несколько экспериментов по классификации наборов данных с использованием *BoWC* против других методов векторизации текстов (рис. 4.7 – 4.11). Метод встраивания слов, использованный в этом эксперименте, представляет собой *self-FastText*, обученный на самих наборах данных (рис. 4.7). Количество концептов было постепенно увеличено с 10 до 300, при этом пороговые значения схожести увеличены с 0,2 до 0,6.

Эксперимент по классификации набора данных *Reuters* (рис. 4.7) выбран в качестве стандарта для анализа, поскольку все эксперименты имеют одинаковый характер. Отмечается, что *BoWC* начинает давать соревновательные результаты, используя всего 50 концептов. Точнее, он превзошел *BoC* на 1% всего с 20 концептами, а также превзошел *averaged-GloVe* с 90 концептами. Отметим, что для ряда концептов, превышающих 200, точность начинает незначительно снижаться из-за того, что концепты становятся пересекающимися и, следовательно, недискриминационными.

Еще один важный момент, связанный с встраиванием слов, заключается в том, что при реализации метода слово из документа может одновременно принадлежать более чем одному концепту (мягкая атрибуция). Это связано с тем, что методы встраивания, используемые в этом исследовании, генерируют единый вектор слова независимо от его контекста, и, следовательно, нельзя достоверно судить о том, что слово принадлежит одному концепту, а не другому. Таким образом, чтобы убедиться, что концепт встречается в конкретном документе, каждое слово документа должно сопоставимым со всеми концептами.

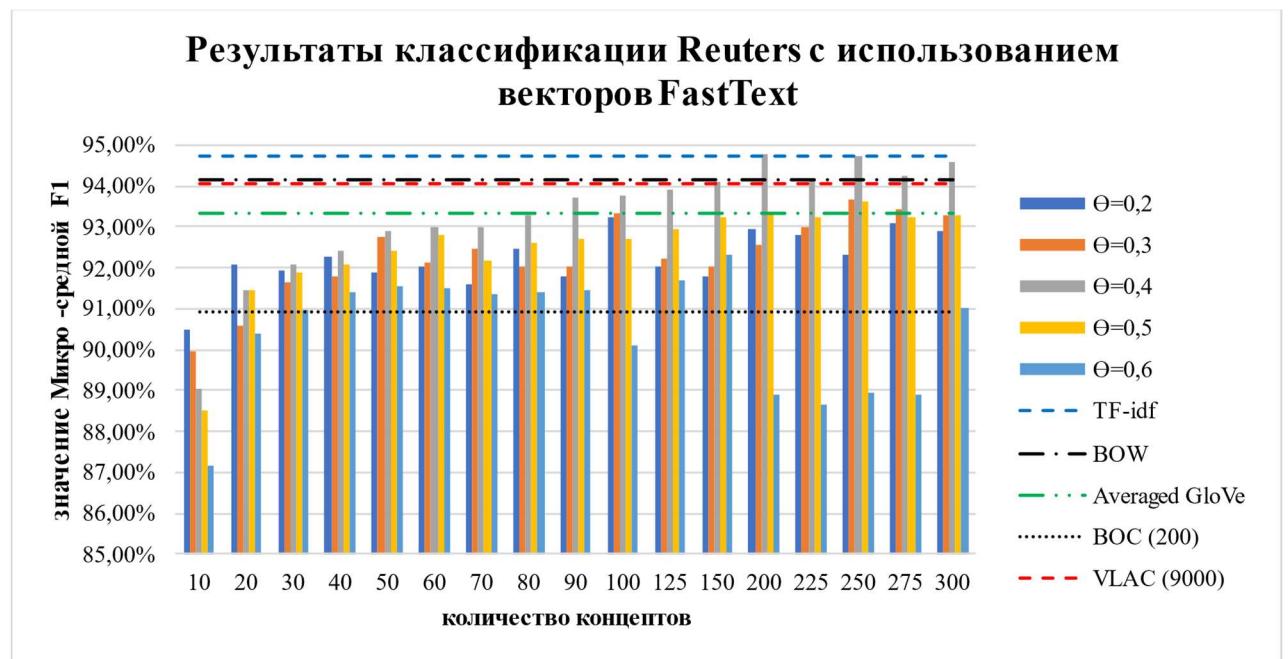


Рисунок 4.7 – Зависимость точности классификации набора данных *REUTERS* от количества концептов и порога сходства

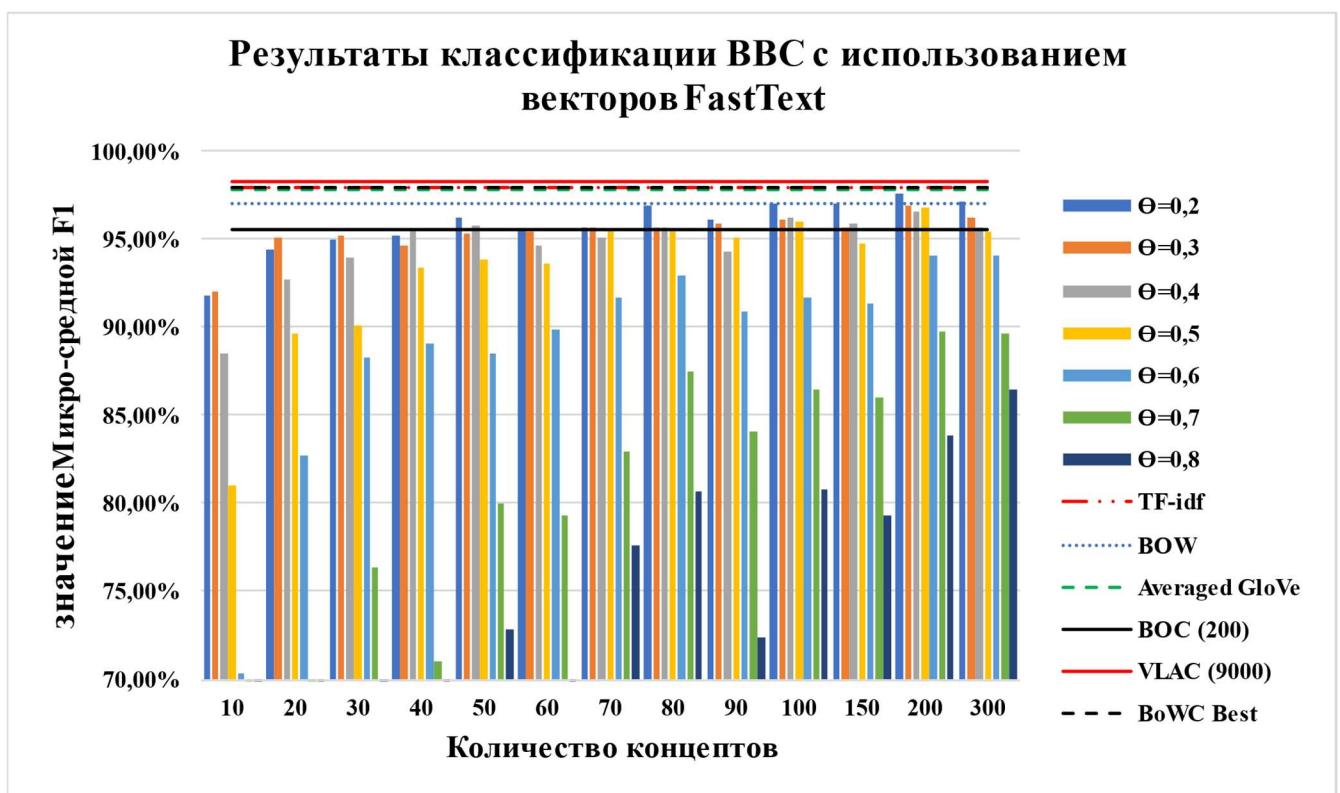


Рисунок 4.8 – Зависимость точности классификации набора данных BBC от количества концептов и порога сходства

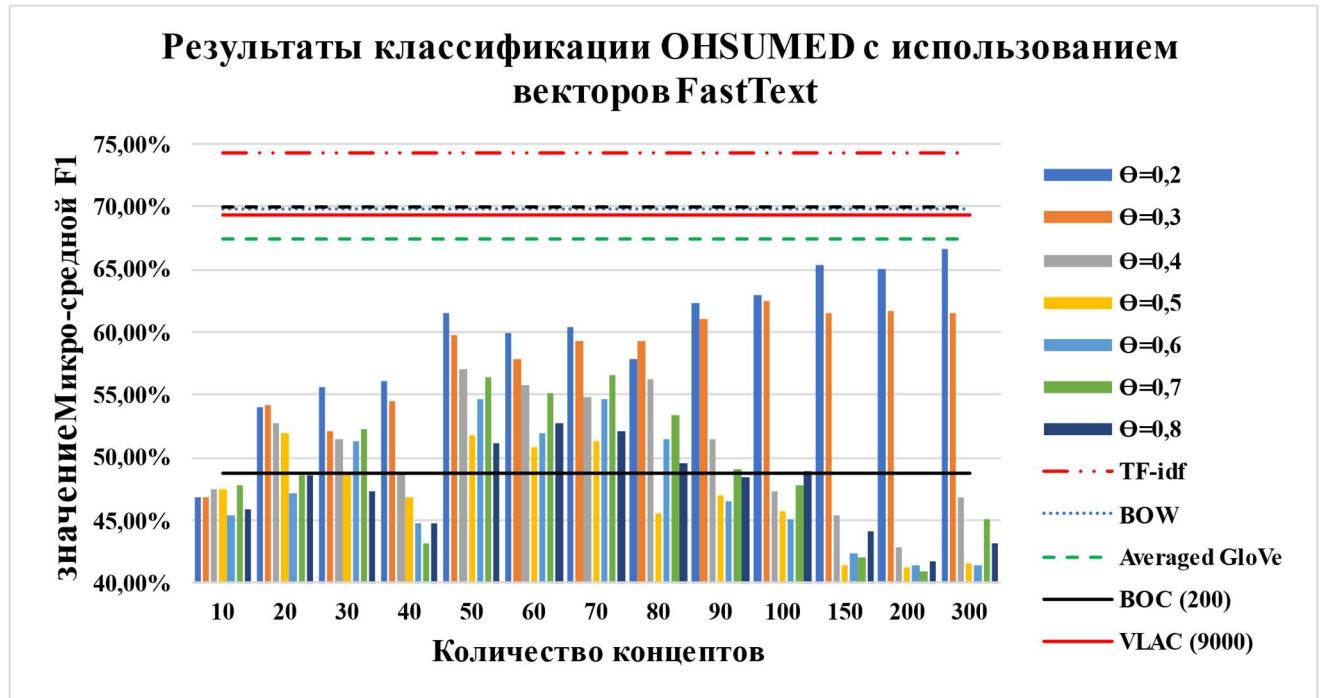


Рисунок 4.9 – Зависимость точности классификации набора данных OHSUMED от количества концептов и порога сходства

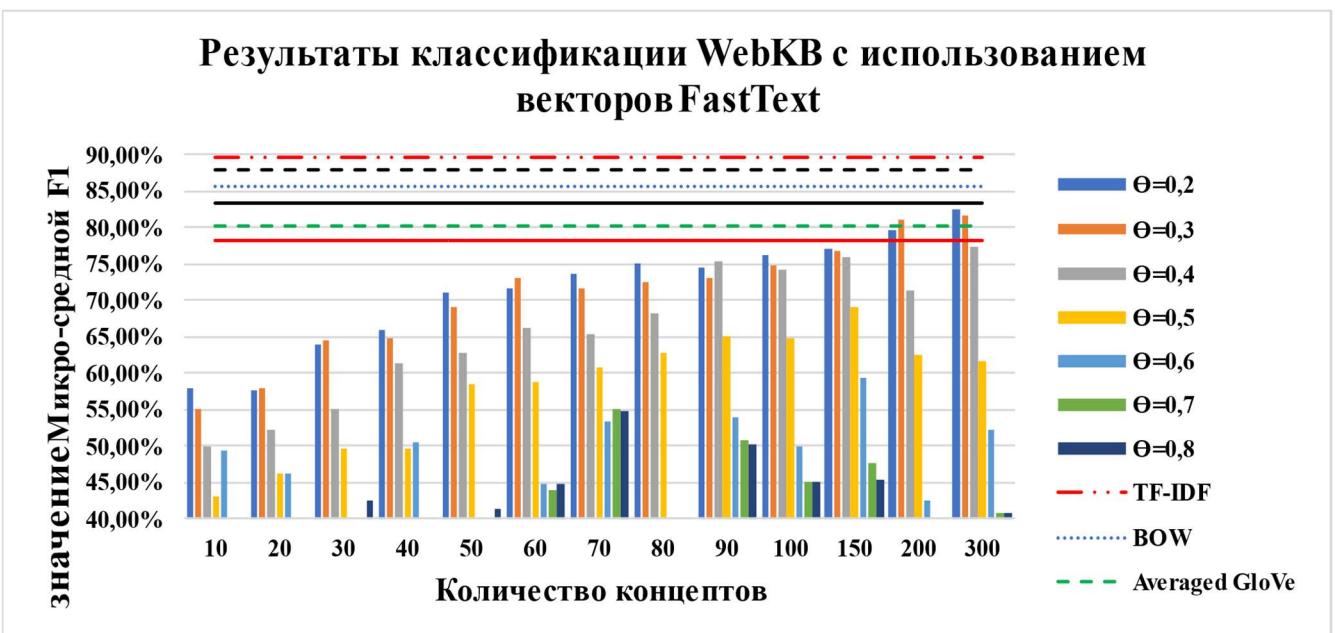


Рисунок 4.10 – Зависимость точности классификации набора данных WebKB от количества концептов и порога сходства

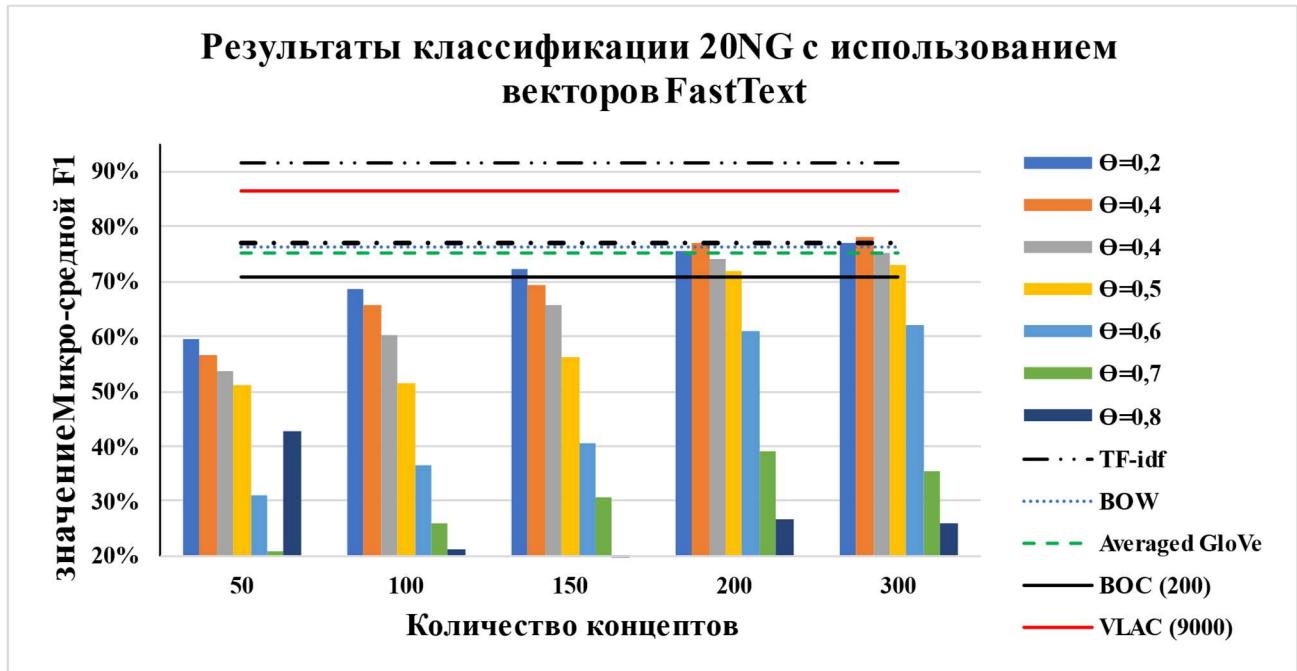


Рисунок 4.11 – Зависимость точности классификации набора данных 20Newsgroups от количества концептов и порога сходства

Предложенный метод использует новую функцию взвешивания концептов, которая сочетает семантику уровня терминов с семантикой уровня концептов, что позволяет создавать более релевантные признаки и низкоразмерные векторы. Это обеспечивает более высокую точность и меньшие вычислительные затраты при использовании алгоритмов интеллектуального анализа текста, таких как классификация и кластеризация.

Во всех экспериментах производительность *BoWC* измерялась и оценивалась на основе наборов данных *Reuters*, *20Newsgroups*, *BBC*, *OHSUMED* и *WebKB* с использованием классификаторов *SVM* и *KNN*. *BoWC* сравнивали с несколькими методами, включая Bag-of-words, *TF-IDF*, Bag-of-Concepts, averaged embedding и *VLAC*. Было показано, что в среднем *BoWC* превосходит большинство базовых методов с точки зрения минимального количества признаков вектора и максимальной точности классификации и кластеризации.

**Потребление памяти.** Разработанный метод *BoWC* генерирует числовые векторы типа float16, каждый элемент которых требует 2 байта для хранения. Следовательно, для хранения 2225 векторов (для набора данных BBC) длиной 200

элементов каждого требуется всего 1,7 мегабайта, в то время как хранение векторов TF-IDF требует 4,26 мегабайта (рис. 4.12).

То же самое касается BERT, VLAC и doc2vec. Таким образом, требования к памяти предлагаемого метода намного ниже, чем у конкурентов, при этом сохраняется высокая точность классификации и кластеризации, как показали вычислительные эксперименты.

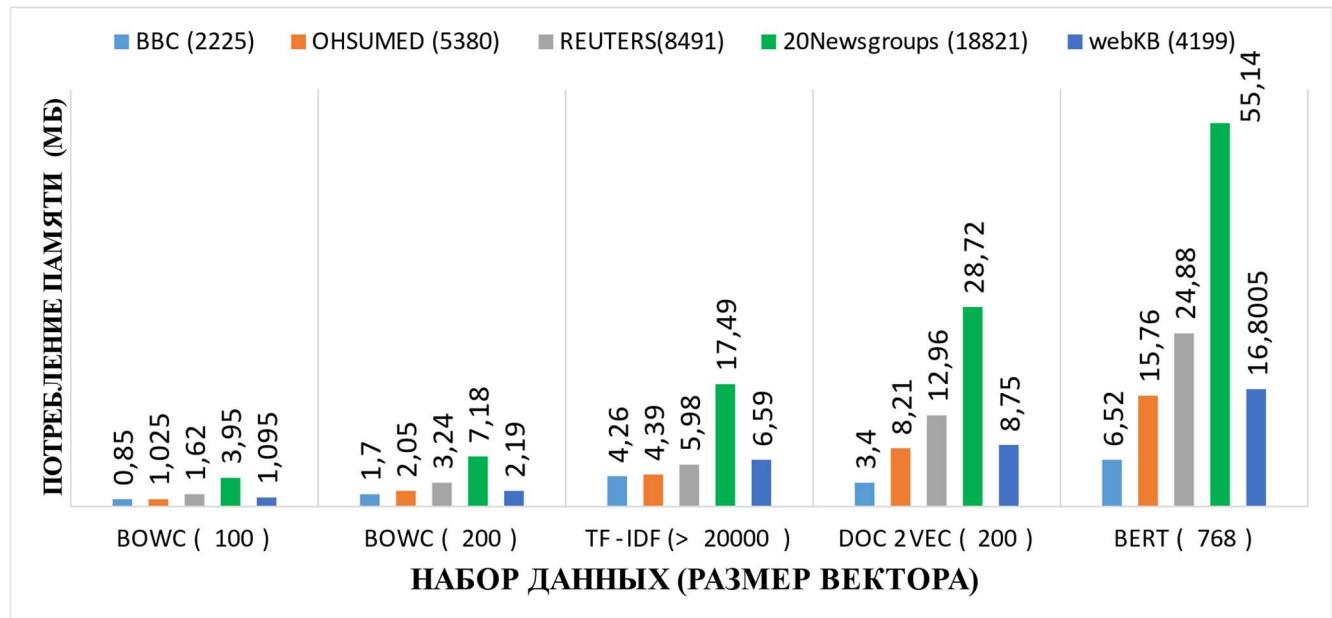


Рисунок 4.12 – Потребление памяти программным приложением

Реализация метода *BoWC* в проведенных экспериментах осуществлялась в соответствии с первым решением, описанным в пункте 2.1.1, которое характеризуется применением функции *CF-EDF*.

В следующем пункте описаны эксперименты по методу *BoWC* с применением алгоритма построения словаря эталонных концептов на основе фильтрации терминов (П. 2.1.2). Данные эксперименты раскрывают другие особенности предложенного метода и позволяют выделить преимущества полученных решений.

#### **4.4. Результаты эксперимента по анализу эффективности метода BoWC с применением фильтрации терминов**

В этом пункте описаны эксперименты по оценке эффективности метода *BoWC* с применением фильтрации терминов на этапе построения словаря эталонных

концептов (П. 2.1.2). Соответственно, для реализации фильтрации терминов в алгоритме построения концептов используется статистическая мера *TF-IDF*. Выбор данной меры обусловлен её эффективностью при определении важных слов на уровне документа и корпуса в целом.

На рисунках 4.13 и 4.14 показаны результаты сравнения первой версии метода *BoWC* со второй версией *BoWC<sub>Filtering</sub>*, применяющей фильтрацию при выполнении задач классификации и кластеризации на пяти наборах данных. *BoWC<sub>Filtering</sub>* (с 100 концептами) значительно превосходит первую версию метода (с 200 концептами). Также отмечается, что примененная фильтрация терминов оказала большее влияние на точность кластеризации, чем – классификации, где *BoWC<sub>Filtering</sub>* достигает разницы в точности кластеризации в пределах от 2,2% (*Reuters*) до 8,7% (*20newsgroup*). В то время, как в задаче классификации разница в точности колеблется от 0,58 % (*Reuters*) до 5,93% (*20newsgroup*).

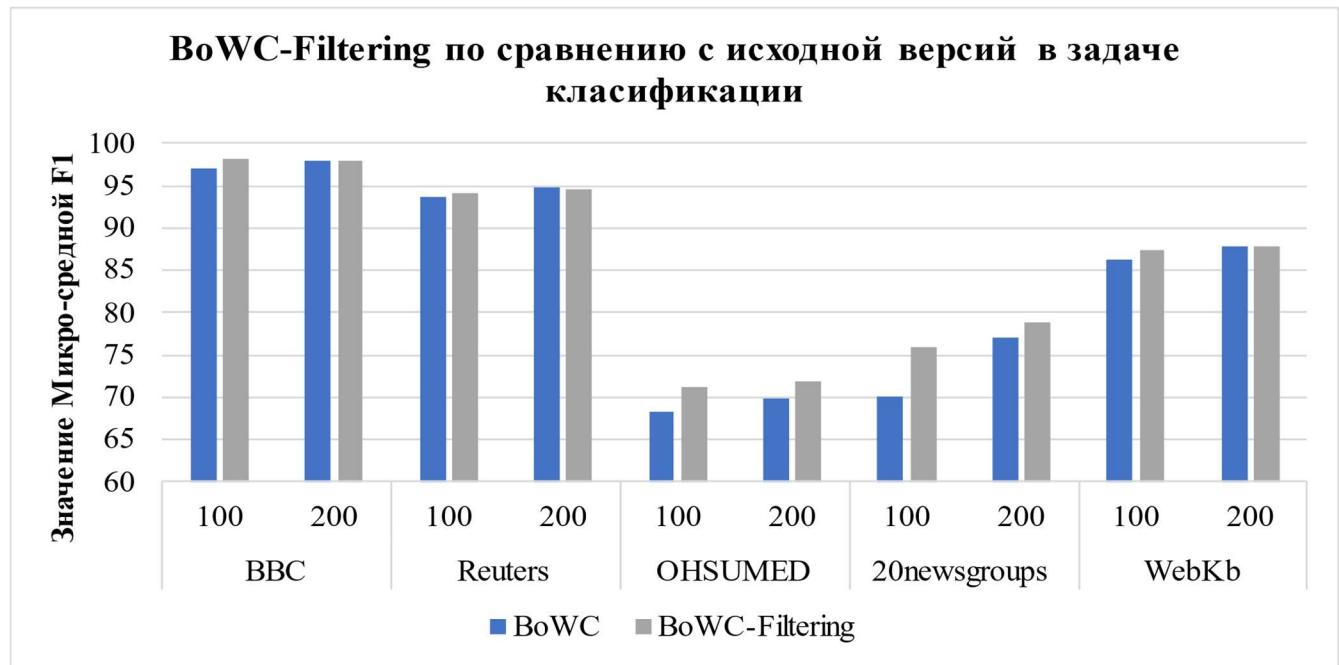


Рисунок 4.13 – Результаты сравнения версий *BoWC* / *BoWC<sub>Filtering</sub>*, в задаче классификации 5 наборов данных

Таблицы 4.5 и 4.6 показывают результаты сравнения метода *BoWC<sub>Filtering</sub>* с остальными базовыми методами в задачах классификации и кластеризации соответственно. При классификации набора данных BBC, *BoWC<sub>Filtering</sub>* превзошел все остальные канонические методы с векторами размера 100 и достиг той же

точности, что и *VLAC* с 9000 признаками. Следует отметить, что точность *BoWC<sub>Filtering</sub>* при классификации для набора данных *Reuters* на 0,07% ниже, чем точность, достигнутая при использовании *BoWC*. Тем не менее, он сохраняет высокую точность и превосходит все другие методы.

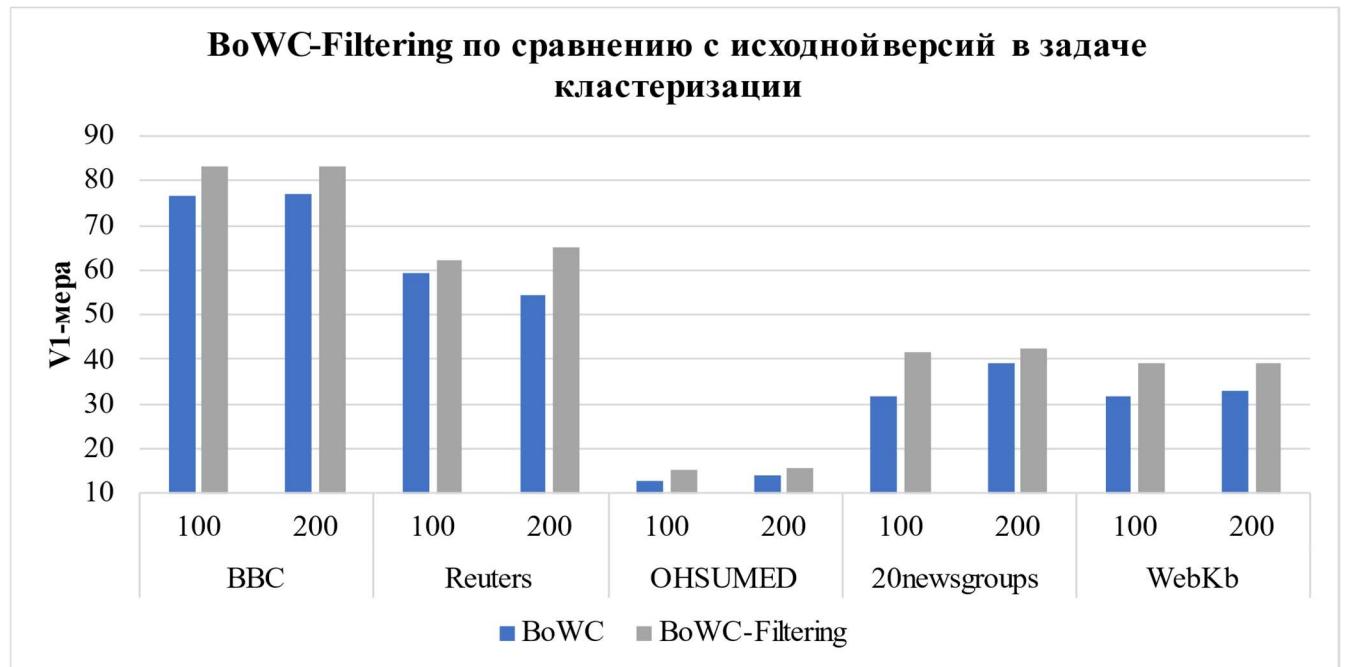


Рисунок 4.14 – Результаты сравнения версий метода *BoWC* / *BoWC<sub>Filtering</sub>*, в задаче кластеризации 5 наборов данных

Таблица 4.5 – Результаты классификации, измеренные с помощью меры Микро-усредненной F1

| Методы (размер вектора)               | Микро-усредненная мера F1 (%) |              |              |              |              |
|---------------------------------------|-------------------------------|--------------|--------------|--------------|--------------|
|                                       | BBC                           | RE           | ОН           | 20NG         | WebKB        |
| <b>BoWC (200)</b>                     | 97.89                         | <b>94.75</b> | 69.95        | 77.07        | 87.77        |
| <b>BoWC<sub>Filtering</sub> (100)</b> | <b>98.2</b>                   | 94.23        | 71.25        | 86           | 87.3         |
| <b>BoWC<sub>Filtering</sub> (200)</b> | 98                            | 94.7         | <u>72.0</u>  | <u>85.89</u> | <u>87.93</u> |
| <b>TF-IDF (&gt;20000)</b>             | 97.89                         | 94.7         | <b>74.31</b> | <b>91.49</b> | <b>89.5</b>  |
| <b>BoW (&gt;20000)</b>                | 96.98                         | 94.15        | 69.89        | 83.85        | 85.56        |
| <b>Averaged GloVe (300)</b>           | 97.75                         | 93.32        | 67.39        | 78.7         | 80.22        |
| <b>self-averaged w2v (300)</b>        | 91.599                        | 87.81        | 56.78        | 62.37        | 83.27        |
| <b>BoC<sub>CF</sub>_IDF (200)</b>     | 95.48                         | 90.9         | 48.8         | 69.44        | 78.26        |
| <b>VLAC (9000)</b>                    | 98.2                          | 94.06        | 69.37        | 86.4         | 85.54        |
| $\Delta_{\text{Best}}$                | +0.0                          | +0.05        | <b>+2.31</b> | -5.6         | <b>+1.57</b> |

Хотя *BoWC<sub>Filtering</sub>* обеспечивает более высокую точность, чем *BoWC*, он по-

прежнему занимает второе место после *TF-IDF* в следующих наборах данных: *OH*; *20newsgroup*; *Webkb*. Автор утверждает, что это связано с тем, что в некоторых текстах недостаточно слов для извлечения информации о каждом понятии. Это снижает вероятность построения лучшего вектора концептов, потому что среднее количество терминов вне словаря значительно увеличивается. Это приводит к выводу, что версии *BoWC* и *BoWC<sub>Filtering</sub>* лучше работают с длинными документами и могут превзойти другие методы в этой области.

Для задачи кластеризации, как показано в таблице 4.6, с использованием модели встраивания *FastText* и использованием всего 100 концептов, предложенный метод превзошел все другие методы на всех наборах данных. Поле «Best» в таблице 4.6 показывает, что *BoWC* превзошел все другие методы со 100 концептами, но результаты варьировались в зависимости от используемых встраивания слов.

Таблица 4.6 – Результаты кластеризации, измеренные с помощью меры V1

| Методы (размер вектора)               | Значения V1-мера (%) |             |              |             |           |
|---------------------------------------|----------------------|-------------|--------------|-------------|-----------|
|                                       | BBC                  | RE          | OH           | 20NG        | WebKB     |
| BoWC (Best)                           | 81                   | 59.1        | 15.5         | 39.6        | 34        |
| BoWC <sub>FastText</sub> (100)        | 76.8                 | 59.1        | 12.9         | 31.6        | 31.8      |
| BoWC <sub>FastText</sub> (200)        | 77.1                 | 54.5        | 14           | 39.1        | 32.8      |
| <i>BoWC<sub>Filtering</sub></i> (100) | <b>83.2</b>          | 61.3        | 15.3         | 40.3        | 38.9      |
| <i>BoWC<sub>Filtering</sub></i> (200) | <u>82.7</u>          | <b>64.3</b> | <b>15.54</b> | <b>42.5</b> | <b>39</b> |
| TF-IDF                                | 66.3                 | 51.3        | 12.2         | 36.2        | 31.3      |
| BoW                                   | 20.9                 | 24.8        | 2.7          | 2.1         | 2.1       |
| Averaged GloVe (300)                  | 77.4                 | 48.1        | 10.9         | 38.1        | 21.9      |
| self-averaged w2v (300)               | 63.1                 | 55.7        | 11.5         | 32.5        | 32.4      |
| BoC <sub>CF_IDF</sub> (200)           | 74.3                 | 52.7        | 10           | 39.4        | 8.8       |
| VLAC (9000)                           | 80.8                 | 45.6        | 11.8         | —           | 28.6      |
| Δ <sub>Best</sub>                     | +2.4                 | +8.6        | +3.34        | +3.1        | +6.6      |

Например, в наборах данных *BBC*, *20newsgroup* и *WebKb* метод достиг наилучших результатов с самообученной (self-trained) моделью встраивания *GloVe*, а в наборе данных *OHSUMED* наилучшие результаты были получены с предварительно обученным встраиванием *FastText*. В наборе данных *Reuters* именно самообучающийся *FastText* дал наибольшую точность.

В целом результаты показывают, что метод *BoWC* работает более эффективно

с длинными текстами, чем с короткими. Обоснование заключается в том, что чем длиннее документ, тем больше слов в словаре (с векторным представлением), что позволяет извлечь больше информации о каждом концепте.

Результаты экспериментов показали, что применение фильтрации терминов повысило эффективность разработанного метода векторизации. Это объясняется тем, что фильтрация удаляет бессмысленные слова, вызывающие шум в кластерах и, таким образом, кластеры становятся более очищенными и выразительными.

Во всех предыдущих экспериментах униграмма использовалась как наименьшая единица для представления документа и создания словаря эталонных концептов. В следующем пункте обсуждаются результаты экспериментов по применению алгоритма построения словаря эталонных концептов, основанного на извлечении ключевых фраз вместо униграмм, целью которого является решение проблемы неоднозначности слов и повышение репрезентативной способности строящихся концептов.

#### **4.5. Результаты эксперимента по использованию словаря эталонных концептов на основе ключевых фраз**

В данном пункте представлены результаты эксперимента по оценке метода *BoWC* с применением алгоритма построения концептов на основе извлечения ключевых фраз и модифицированной весовой функции концептов, предложенных в главе 3 (п. 3.3). Модификации предполагают использование ключевых фраз (н-грамм) в качестве наименьших единиц, которые представляют документ и участвуют в алгоритме построения словаря эталонных концептов.

Целью этих экспериментов является оценка влияния реализации алгоритма построения концептов на дискриминационную способность векторов методов *BoWC* и *BoC*. Это достигается путем оценки однородности концептов в словаре, а затем оценки эффективности построенных на его основе векторов в задаче кластеризации текста. Для метода *BoC*, который следует тому же подходу, что и метод *BoWC*, протестированы две его версии, первая из которых использует весовую функцию *CF-IDF* и обозначена как «*BoC<sub>CF-EDF</sub>*» в таблице 4.7. Вторая

версия применяет модифицированную функцию определения весов концептов, расширенную функцией веса ключевой фразы, используемой в методе *BoWC*, и она обозначена как «*BoC<sub>CF-EDF-FBKE</sub>*» в таблице 4.7.

Выполняются те же операции предварительной обработки текста, что и в предыдущих экспериментах (перевод в нижний регистр и удаление стоп-слов, символов и цифр). Для сравнительных методов были реализованы те же настройки, что и в предыдущих экспериментах, а в качестве алгоритма кластеризации документов использовался алгоритм К-средних.

Согласно алгоритму построения словаря эталонных концептов на основе извлечения ключевых фраз (п. 3.3), для построения концептов на основе n-грамм сначала извлекаются n-граммы (длиной 2-4) с помощью метода *FBKE* [123], а затем эти n-граммы фильтруются с использованием функции парсера *SpaCy*, чтобы получить именные фразы, с правильной лингвистической структурой. Парсер *SpaCy* был выбран благодаря его скорости и эффективности, как показано в экспериментах, проведенных в пункте 4.6. При этом *SpaCy* применяется к n-граммам, а не к предложениям.

Векторы встраивания были сгенерированы с использованием модели встраивания *FastText* с размером вектора 300. Модель встраивания слов обучается на самих данных с размером окна 15. Весовая функция алгоритма *FBKE* применяется для выбора 200 наиболее важных ключевых фраз для представления документа. Второй шаг заключается в создании словаря эталонных концептов путем применения алгоритма кластеризации сферических К-средних к векторам ключевых фраз с порогом сходства 0,6.

Как уже упоминалось, каждый кластер представляет собой концепт. Порог сходства ключевой фразы с центроидом кластера был эмпирически установлен равным 0,42. Векторы документа генерируются с помощью функции (38) с количеством концептов от 100 до 200 для каждого набора данных.

В качестве метрики оценки дискриминационной способности векторов использовалась V-мера [141] при решении задачи кластеризации набора данных *BBC*. На рисунке 4.15 показано сравнение производительности исходной и

модифицированной версии метода *BoWC* при выполнении задач кластеризации набора данных *BBC*. Версия *BoWC*, реализующая алгоритм извлечения ключевых фраз, называется «*BoWC<sub>FBKE</sub>*». В целом, *BoWC<sub>FBKE</sub>* (всего 100 концептов) значительно превосходит оригиналный *BoWC* (всего 200 концептов).

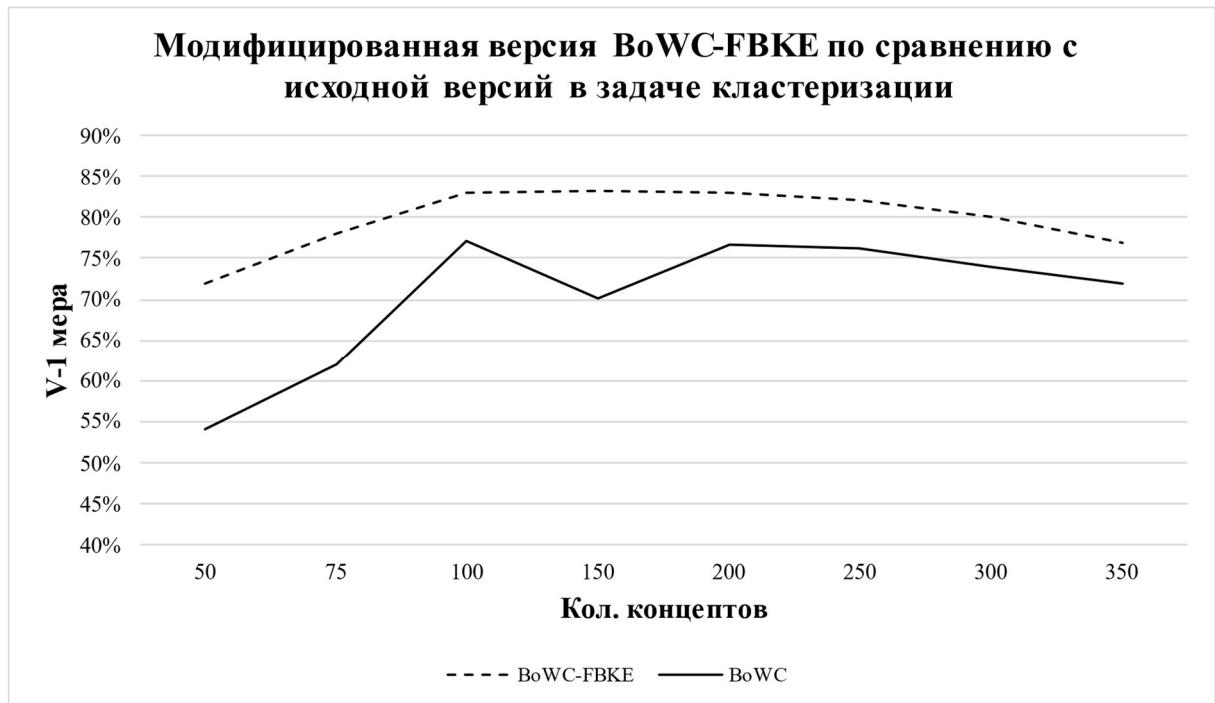


Рисунок 4.15 – Результаты сравнения модифицированной и исходной версий *BoWC* в задаче кластеризации *BBC*

Также стоит отметить, что взвешивание концептов с использованием среднего значения весов ключевых фраз, входящих в концепт, привело к усовершенствованию не только метода «*BoWC*», но и метода «*BoC*» (рис. 4.16, табл. 4.6). Это подтверждает важность предложенного коэффициента для улучшения качества результирующих векторов и делает их более дискриминационными.

Относительно влияния количества концептов на точность кластеризации отмечается, что *BoWC* начинает обеспечивать производительность, сравнимую с другими методами, только со 100 концептами (рис. 4.15, табл. 4.7). Для ряда концептов, превышающих 200, точность незначительно снижается, учитывая, что концепты становятся перекрывающимися и, следовательно, недискриминационными.

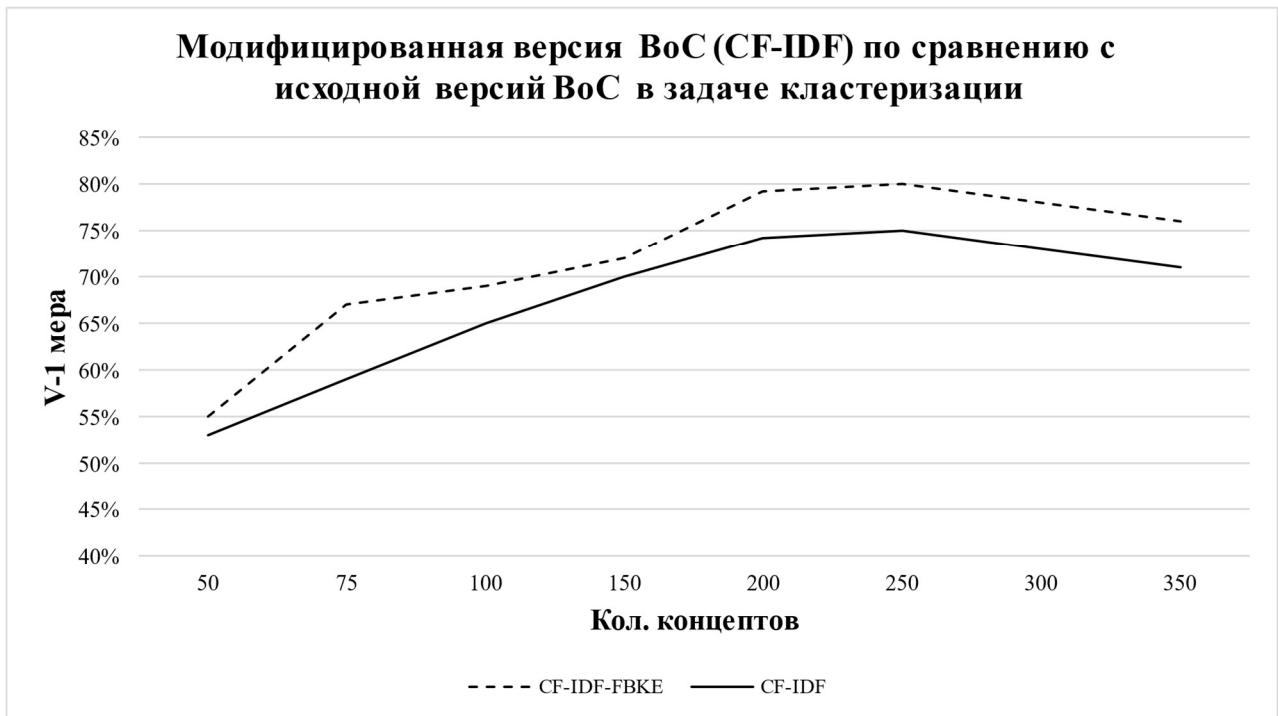


Рисунок 4.16 – Результаты сравнения модифицированной и исходной версий *BoC* в задаче кластеризации BBC

Таблица 4.7 – Результаты кластеризации, измеренные с помощью меры V1

| <b>Метод (размер вектора)</b>                     | V1-мера (%) |             |             |             |              |
|---------------------------------------------------|-------------|-------------|-------------|-------------|--------------|
|                                                   | <b>BBC</b>  | <b>RE</b>   | <b>OH</b>   | <b>20NG</b> | <b>WebKB</b> |
| BoWC <sub>FastText</sub> (100)                    | 76.8        | <u>59.1</u> | 12.9        | 31.6        | 31.8         |
| BoWC <sub>FastText</sub> (200)                    | 77.1        | 54.5        | 14.0        | 39.1        | 32.8         |
| модифицированный BoWC <sub>FBKE</sub> (100)       | <u>83.0</u> | <u>62.3</u> | <u>15.4</u> | <u>41.4</u> | <u>39.0</u>  |
| модифицированный BoWC <sub>FBKE</sub> (200)       | <u>83.2</u> | <u>64</u>   | <u>15.5</u> | <u>42.6</u> | <u>39.1</u>  |
| BoC <sub>CF_IDF</sub> (200)                       | 74.3        | 52.7        | 10          | 39.4        | 8.8          |
| модифицированный BoC <sub>CF_IDF-FBKE</sub> (200) | 79.3        | 59          | 13          | 40.0        | 20           |
| TF-IDF (>20000)                                   | 66.3        | 51.3        | 12.2        | 36.2        | 31.3         |
| BoW (>20000)                                      | 20.9        | 24.8        | 2.7         | 20.1        | 20.1         |
| Averaged FastText (300)                           | 77.4        | 48.1        | 10.9        | 38.1        | 21.9         |
| self-trained FastText (300)                       | 63.1        | 55.7        | 11.5        | 32.5        | 32.4         |
| $\Delta_{\text{Best}}$                            | +3.9        | +8.3        | +3.3        | +3.2        | +6.7         |

В целом применение алгоритма построения словаря эталонных концептов привело к значительному повышению эффективности как метода *BoWC*, так и *BoC*, что наглядно отразилось в снижении частоты ошибок алгоритма кластеризации на 2-6% по сравнению с исходной версией двух методов. Это подтверждает важность параметра, добавленного в формулу (34), в повышении дискриминационной способности результирующих векторов документов.

Для оценки однородности концептов, созданных предложенным алгоритмом, автор использовал *Индекс Дэвиса-Боулдина DBI* (форм. 45), который показывает соответствие ключевых фраз, включенных в один кластер, одному конкретному концепту.

Таблица 4.8 демонстрирует результаты эксперимента по оценке алгоритма построения концептов на основе извлечения ключевых фраз (*n*-граммы\_FBKE). Сравниваются оценки однородности концептов, созданных предложенным алгоритмом, с теми, которые были получены с помощью других алгоритмов извлечения ключевых фраз, таких как *YAKE* [115], *RAKE* [114] и алгоритм на основе меры *TF-IDF*. Результаты алгоритмов на основе *n*-грамм также сравниваются с базовым алгоритмом на основе унigrams, используемым в концептуальном подходе к решению задачи векторизации.

Таблица 4.8 – однородность кластеров, измеренная с помощью меры *Индекса Дэвиса-Боулдина*

| <b>Нормализованные значения (DBI) в диапазоне [0,1]</b> |             |             |            |             |                  |
|---------------------------------------------------------|-------------|-------------|------------|-------------|------------------|
| Алгоритм                                                | Кластер 1   | Кластер 2   | Кластер 3  | Кластер 4   | Среднее значение |
| Унigrams                                                | 0.7         | 0.77        | 0.72       | 0.69        | 0.72             |
| <b>n</b> -граммы_FBKE                                   | <b>0.25</b> | <b>0.35</b> | <b>0.2</b> | <b>0.23</b> | <b>0.32</b>      |
| n-Граммы_Yake                                           | 0.48        | 0.46        | 0.45       | 0.49        | 0.5              |
| n-граммы_Rake                                           | 0.65        | 0.62        | 0.6        | 0.64        | 0.636            |
| n-граммы_TF-IDF                                         | 0.72        | 0.9         | 0.81       | 0.73        | 0.79             |

Для наглядности анализа была сформирована выборка из 4 кластеров (концептов) словаря эталонных концептов для каждого алгоритма, а также

рассчитана среднее значение однородности для всех концептов. Результаты показывают, что предложенный алгоритм превосходит другие аналоги и достигает значительного улучшения однородности кластеров, представляющих концепты. Это улучшение также отразилось на дискриминационной способности векторов, созданных на основе этих кластеров, как показали предыдущие результаты (таблица 4.7, рисунки 4.15 и 4.16).

Данные результаты доказывают эффективность предложенных автором решений, а также их вклад в повышение производительности методов векторизации текста.

Отметим, что предложенный алгоритм построения словаря эталонных концептов помимо повышения эффективности векторизации текста успешно используется для решения следующих прикладных задач:

- расширения ключевых фраз, описывающих документ, путем извлечения ключевых слов и фраз, не упомянутых явно в самом тексте;
- построения интерактивных визуализаций концептов документов, которые позволяют создавать настраиваемые резюме для длинных и многотемных документов [142];
- разработки профилей пользователей для выработки рекомендации по использованию контента.

Отметим, что алгоритм построения концептов опирался на алгоритм извлечения ключевых фраз на основе парсера (п. 3.3). Этот алгоритм разработан как универсальный и может применяться к задачам извлечения ключевых фраз независимо от словаря эталонных концептов или метода *BoWC*. С целью подтверждения данного утверждения, а также закрепления результатов, в следующем пункте рассматривается эксперимент по оценке эффективности алгоритма извлечения ключевых фраз на основе парсера.

#### 4.6. Анализ эффективности алгоритма извлечения ключевых фраз на основе парсера

В пункте 4.5 производительность метода *BoWC* была протестирована после добавления фильтра на основе SpaCy в качестве выходного слоя алгоритма извлечения ключевых фраз *FBKE*, что показало значительное улучшение производительности. В данном пункте представлены результаты экспериментов, проведенных над алгоритмом извлечения ключевых фраз на основе парсера с целью подтверждения эффективности предлагаемого решения.

**Реализация алгоритма.** Согласно пункту 3.3, алгоритм извлечения ключевых фраз разработан с использованием парсера, который отфильтровывает фразы с неправильной языковой структурой. Чтобы выбрать ключевые фразы-кандидаты, необходимо применить весовую функцию к n-граммам, извлеченным из текста. В текущих экспериментах функция *TF* в формуле (28) использовалась для взвешивания и ранжирования извлеченных ключевых фраз. Затем были применены три типа лингвистических парсеров для сравнения по точности и скорости.

Приведём описание данных парсеров:

SpaCy – это бесплатная библиотека с открытым исходным кодом для расширенной обработки естественного языка (NLP) в *Python* [72]. Данная библиотека имеет свой одноимённый парсер, который отвечает за анализ синтаксической структуры предложений. Реализация парсера SpaCy выполняет синтаксический анализ зависимостей и включает в себя методы машинного обучения для прогнозирования в процессе разбора. Он использует статистическую модель, обученную на размеченных данных, для предсказания наиболее вероятного перехода на каждом шаге. Парсер применяет вариант немонотонной дуговой системы с переходами [143], с добавлением перехода "break" для выполнения сегментации предложений. Для обеспечения возможности предсказания неверных разборов парсер использует псевдопроективное преобразование зависимостей, предложенное в [144].

AllenNLP [46] – это платформа для исследований в области глубокого обучения методов обработки естественного языка. AllenNLP не имеет встроенного парсера, подобного SpaCy, однако предоставляет предварительно обученные модели для синтаксического парсинга составляющих и парсинга зависимостей. Парсер зависимостей следует модели глубокого биаффинного внимания для нейронного синтаксического анализа зависимостей, который использует нейронное внимание в простом графовом парсере зависимостей [145]. Парсер составляющих построен на основе минимальной нейронной модели, основанной на независимой оценке меток. Модель использует встроенную процедуру ELMo и кодирует последовательность текста с помощью стекового энкодера Seq2SeqEncoder.

Stanza [146] – это пакет для анализа естественного языка, написанный на Python, созданный Стенфордской группой NLP. Он предоставляет несколько инструментов для анализа естественного текста, одним из которых является представление синтаксической структуры в виде дерева зависимостей. Как и в AllenNLP парсер зависимостей в Stanza реализует Bi-LSTM-сеть, основанную на глубокой биаффинной нейронной модели, которая относится к категории парсеров зависимостей на основе графов.

Результаты работы предложенного алгоритма сравнивались с результатами известных алгоритмов, *Yake* и *Rake*<sup>7</sup>.

Алгоритм *Yake* (Yet Another Keyword Extractor) [115] использует комбинацию статистических и лингвистических признаков для определения важности слов или фраз в тексте. Он учитывает как частоту и распределение терминов внутри документа, так и их семантическую связь. Алгоритм использует подход скользящего окна для выявления кандидатов на ключевые фразы, а затем применяет функцию оценки для их ранжирования по степени релевантности.

Алгоритм *Rake* (Rapid Automatic Keyword Extraction) [114] опирается на распределение слов в тексте для определения фраз-кандидатов. Алгоритм сначала

---

<sup>7</sup> Реализация указанных алгоритмов <https://github.com/Ali-MH-Mansour/Keyword-Extraction.git>

разбивает текст на отдельные слова, а затем генерируются фразы-кандидаты, идентифицируя последовательности слов, разделенные стоп-словами или знаками пунктуации. Алгоритм присваивает оценки этим фразам на основе их частоты и степени совместной встречаемости слов.

Для *Yake* из каждого текста извлекаются первые 20 *n*-грамм (от 1 до 3). То же самое и с *Rake*, но *Rake* не позволяет указать количество извлекаемых слов. При этом, не выполнялась никакая предварительная обработка текстов, чтобы не повлиять на структуру, но после получения именных фраз обработка текста проводилась, например, такая как удаление стоп-слов.

**Набор данных и метрика оценки.** Для оценки предложенного алгоритма по сравнению с ранее описанными использовался набор данных *Inspec* и мера оценки *MAP@K*.

Набор данных *Inspec* [147], состоит из 2000 рефератов статей из научных журналов по компьютерным наукам. Каждому документу присвоены два набора ключевых слов. Контролируемые ключевые слова, которые являются назначенными вручную ключевыми словами и появляются в тезаурусе *Inspec*, но могут не появляться в документе, и неконтролируемые ключевые слова, которые свободно назначаются редакторами.

Для оценки результатов отслеживалось *точное совпадение*, при котором автоматически извлеченная ключевая фраза из документа должна точно соответствовать ключевой фразе эталонного паттерна для документа. Традиционно извлечение ключевых слов по своей природе является задачей ранжирования [148]. Исходя из этого, для определения эффективности предлагаемого алгоритма используется одна из наиболее часто используемых мер качества для ранжирования, а именно *MAP@K* – средняя точность на *K* элементах (Mean average precision at K). Таким образом, совпадения проверяются среди первых *K* ключевых слов, возвращаемых алгоритмом. Для оценки *MAP@K* сначала подсчитывается точность на *K* элементах *P@K*. Базовая метрика качества ранжирования для одного объекта определяется следующим выражением:

$$pr_k = \frac{\sum_{k=1}^K r_{true}(e_i)}{k}, \quad (46)$$

где  $e_i$  это элемент (ключевое слово)  $e \in E$ , который в результате перестановки оказался на  $i$ -ой позиции,  $r_{true}(e_i)$  – функция равная 1, если  $e$  релевантен, 0 – в противном случае. Недостаток этой метрики заключается в том, что не учитываются позиции правильных элементов (порядок элементов). Далее на основе  $pr_K$  рассчитывается средняя точность на  $K$  элементах ( $apr_K$ ) на основе следующего выражения:

$$apr_K = \frac{1}{K} \sum_{k=1}^K r_{true} \pi^{-1}(k) \cdot pr_k. \quad (47)$$

Такая мера учитывает позиции элементов, но качество ранжирования оценивается для отдельно взятого объекта. Для того, чтобы посчитать MAP@K для  $N$  различных объектов следующим образом вычисляется среднее по  $apr@K$  для каждого:

$$map_k = \frac{1}{N} \sum_{i=1}^N apr_k^i, \quad (48)$$

где  $apr_k^i$  – это  $apr_K$  для  $i$ -го объекта.

**Результаты и выводы.** Производительность предложенного алгоритма сравнивается при использовании трех разных парсеров с производительностью двух исследованных алгоритмов *Yake* и *Rake*.

Результаты вычислительного эксперимента представлены в таблицах 4.9 и 4.10. Из таблицы 4.9 заметим, что разработанный алгоритм (с использованием любого парсера) превосходит методы *Yake* и *Rake*, подтверждая утверждение о том, что большинство ключевых слов являются именными фразами. Наилучшие результаты достигаются парсером SpaCy, как с точки зрения точности, так и с точки зрения скорости. Результаты *Yake* и *Rake* приближаются к показателям TF-SpaCy по метрике MAP@20.

В таблице 4.10 показаны результаты эксперимента по оценке способности алгоритма определить правильные ключевые фразы. В строке “Gold keywords”

выделены полужирным шрифтом ключевые слова, определенные из текста экспертами. Что касается фразы вне текста, то в данной работе они не учитываются.

Таблица 4.9 – Точность алгоритмов, измеренная с помощью MAP@K

| Метод<br>К  | MAP@K (%) |           |            |             |
|-------------|-----------|-----------|------------|-------------|
|             | @1        | @5        | @10        | @20         |
| TF-SpaCy    | <b>36</b> | <b>15</b> | <b>9.5</b> | 7.7         |
| TF-Stanza   | 30.3      | 13.1      | 8.3        | 6           |
| TF-AllenNLP | 29.9      | 12.8      | 8.1        | 6.4         |
| Yake        | 26.7      | 12.7      | 8.4        | <b>7.75</b> |
| Rake        | 16.9      | 10.3      | 8.1        | 7.4         |

Из Таблицы 4.10 видно, что разработанный алгоритм оптимизации определения ключевых фраз находит 4 правильных фраз из 5 с помощью *SpaCy*, в то время как *AllenNLP* определяет только 2. Некоторые из парсеров неверно определяют ключевую фразу. Например, *Stanza* правильно находит ключевую фразу “Bar code labels”, в то время как *SpaCy* определяет только её часть “code labels”. *AllenNLP* не распознает “food processing” и “Bar code labels”, потому что в нем учитываются ключевые фразы “Fresh tracks [food processing]” и “Bar code labels” как одна фраза на основе существительного.

Алгоритмы *Yake* и *Rake* неправильно определили глагол «linked» как часть ключевого слова «wireless terminals», тогда как трем другим парсерам удалось идентифицировать его правильно и исключить из фразы.

Хотя *SpaCy* превосходит другие парсеры в задаче определения именных фраз, нельзя однозначно утверждать, что парсеры, основанные на переходах, всегда превосходят парсеры, основанные на графах. Это зависит от нескольких факторов, включая структуру и сложность анализируемого текста, а также эффективность реализации парсера и его внутренних алгоритмов.

Парсеры на основе переходов постепенно строят дерево и принимают решения о разборе на основе локального контекста, что может помочь распознавать и извлекать именные фразы, имеющие явные зависимости в пределах короткого расстояния. Таким образом, поскольку текст, использованный в этих экспериментах, относительно короткий, он содержит простые предложения, в

которых преобладают локальные зависимости, подходящие для данного типа синтаксического анализатора. С другой стороны, парсеры на основе графов учитывают всю структуру предложения при построении дерева зависимостей. Этот глобальный контекст может быть полезен для определения неявных зависимостей между словами, которые охватывают несколько слов или предложений, как в примере *AllenNLP* в таблице 4.10.

Таблица 4.10 – Текст из набора данных в качестве примера, показывающего эталонные ключевые слова и ключевые слова, определенные каждым алгоритмом

|               |                                                                                                                                                                                                                                                                                          |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Текст         | Fresh tracks [food processing] <b>Bar code labels</b> and wireless terminals linked to a centralized database accurately track meat products from receiving to customers for Farmland Foods                                                                                              |
| Gold keywords | [' <b>food processing</b> ', ' <b>bar code labels</b> ', ' <b>wireless terminals</b> ', ' <b>Farmland Foods</b> ', 'automatic data capture', 'Intermec Technologies', ' <b>bar codes</b> ', 'data acquisition', 'food processing industry', 'mobile computing', 'production control', "] |
| TF-SpaCy      | ['Fresh tracks', ' <b>food processing</b> ', ' <b>code labels</b> ', ' <b>wireless terminals</b> ', 'centralized database', 'meat products', ' <b>Farmland Foods</b> ']                                                                                                                  |
| TF-AllenNLP   | [' <b>wireless terminals</b> ', ' <b>Farmland Foods</b> ', 'centralized database', 'Fresh tracks', 'meat products']                                                                                                                                                                      |
| TF-Stanza     | [' <b>wireless terminals</b> ', ' <b>Farmland Foods</b> ', ' <b>food processing</b> ', 'customers for Farmland Foods', 'Fresh tracks', 'meat products', 'centralized database', ' <b>Bar code labels</b> ']                                                                              |
| Yake          | [' <b>Bar code labels</b> ', 'wireless terminals <u>linked</u> ', 'centralized database accurately', 'database accurately track', 'accurately track meat', 'track meat products', ' <b>Bar code</b> ', ' <b>Farmland Foods</b> ', 'customers for Farmland', ' <b>food processing</b> ']  |
| Rake          | ['wireless terminals <u>linked</u> ', ' <b>bar code labels</b> ', 'fresh tracks', ' <b>food processing</b> ', ' <b>farmland foods</b> ', 'receiving', 'customers']                                                                                                                       |

**Анализ времени выполнения.** Рисунки 4.17 и 4.18 демонстрируют различие в скорости работы предложенного алгоритма по сравнению с методами *Yake* и *Rake* при обработке разного количества документов. Поскольку время выполнения алгоритмов *Rake*, *Yake* и *SpaCy* очень мало по сравнению с остальными алгоритмами, для наглядности их временные характеристики продемонстрированы в другом масштабе (рис. 4.17).

Время выполнения предложенного автором алгоритма представляет собой сумму времен выполнения нескольких операций: время извлечения *n*-грамм в качестве ключевых слов-кандидатов; время анализа текста на компоненты предложений и именных фраз; время фильтрации этих именных фраз; время фильтрации ключевых слов-кандидатов по именным фразам.

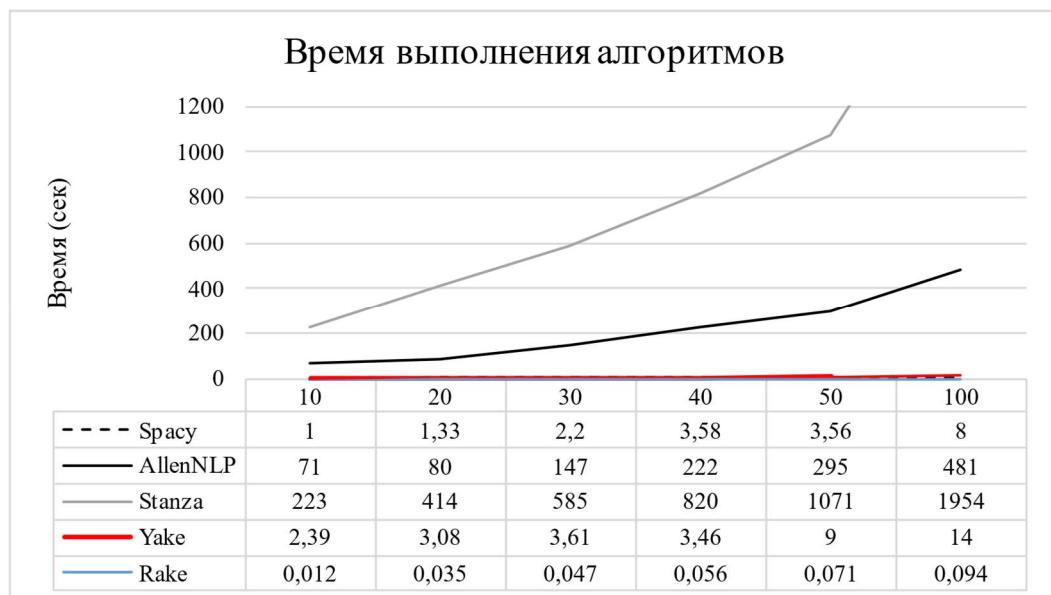


Рисунок 4.17 – Сравнение времени выполнения алгоритмов извлечения ключевых фраз

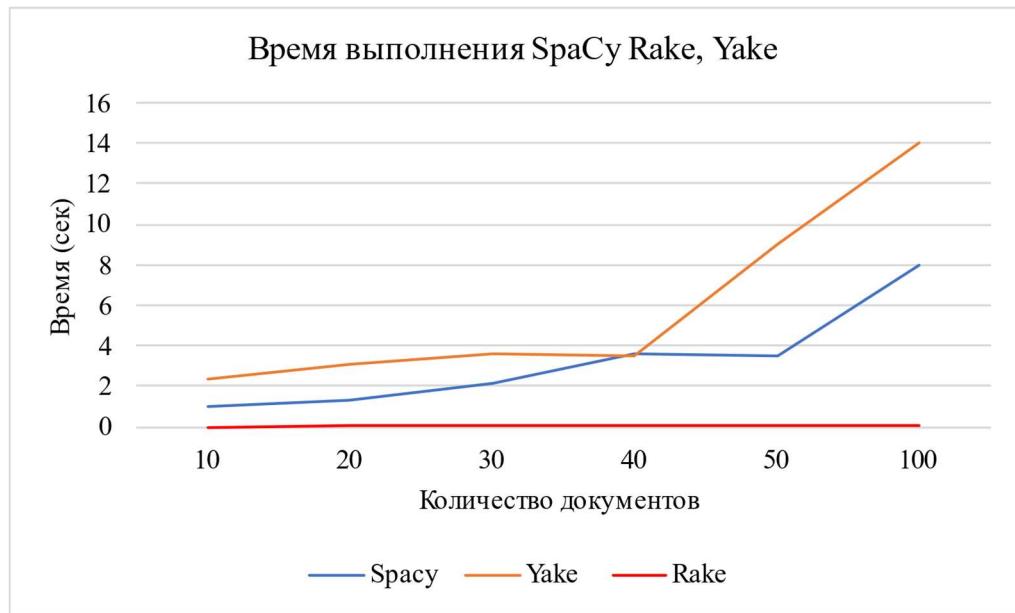


Рисунок 4.18 – Сравнение времени выполнения алгоритмов *SpaCy*, *Yake* и *Rake*

Хотя статистические алгоритмы работают быстрее, чем алгоритмы на основе парсера, поскольку их работа не требует анализа текста, предложенный алгоритм с парсером *SpaCy* (TF-SpaCy) превзошёл все рассмотренные канонические алгоритмы кроме *Rake*.

Время выполнения предложенного алгоритма с парсером *SpaCy* значительно меньше, чем у двух других парсеров, потому что парсеры, которые используют переходы, применяют детерминированный или машинно-обучаемый набор действий для пошагового построения дерева зависимостей. Они обычно работают быстрее и требуют меньше памяти по сравнению с парсерами, основанными на графах, что делает их более эффективными для анализа больших объемов текста. С другой стороны, AllenNLP и Stanza используют модели парсинга на основе графов, которые полагаются на миллионы созданных вручную признаков, что ограничивает их способность к обобщению и замедляет скорость анализа по сравнению с парсерами на основе переходов.

В этом пункте описаны результаты вычислительного эксперимента, проведённого для исследования эффективности алгоритма извлечения ключевых фраз на базе парсера. Результаты подтвердили данную эффективность предложенного алгоритма в выявлении релевантных ключевых фраз и его способность конкурировать с каноническими методами. Кроме того,

эффективность этого алгоритма была доказана достижением повышения уровня производительности метода векторизации текстов (п. 4.5) за счёт оптимизации построения словаря эталонных концептов.

Одной из основных целей, достигаемых применением этих алгоритмов, является создание концептуальных векторов документов с интерпретируемыми признаками. Чтобы подкрепить полученные результаты, в следующем пункте исследуется свойство интерпретируемости результирующих векторов, которое имеет высокую значимость в задачах обработки и анализа текстов на естественном языке.

#### **4.7. Анализ результатов вычислительного эксперимента по исследованию характеристик интерпретируемости векторов**

В данном пункте подчеркивается значимость характеристики интерпретируемости векторов, созданных предложенным методом *BoWC*. Представлен практический пример, иллюстрирующий, как этот метод и разработанные алгоритмы могут повысить эффективность поиска целевой аудитории в системе искусственного интеллекта и машинного обучения «*Look-alike*», с акцентом на роль свойства интерпретируемости векторов в улучшении этого процесса.

**Интерпретируемость векторов признаки.** Одним из преимуществ векторов документов, сгенерированных предложенным автором методом *BoWC*, является то, что они имеют интерпретируемые признаки. Важность интерпретируемости признаков векторов в задачах текстового анализа заключается в нескольких аспектах:

1. *Понимание модели.* Интерпретируемые векторы позволяют нам получить представление о том, как модель делает прогнозы. Анализируя важность и вклад каждого признака, определяются аспекты текста, влияющие на процесс принятия решений моделью. Это позволяет создать достоверную и адекватную модель.

2. *Интеграция предметных знаний.* Интерпретируемые векторы позволяют интегрировать предметные знания в процесс текстового анализа. Определяя

наиболее важные признаки, предметные эксперты могут предоставить дополнительные идеи и контекст для улучшения информативности модели.

*3. Обнаружение и устранение предубеждений.* Интерпретируемые векторы помогают выявлять и устранять предубеждения в задачах текстового анализа. Анализируя вклад каждого признака, определяются те, которые наиболее сильно изменяют прогноз модели. Это позволяет обнаруживать и устранять предубеждения, обеспечивая справедливое и беспристрастное принятие решений.

*4. Соблюдение регулятивных требований.* В некоторых отраслях, таких как финансы и здравоохранение, интерпретируемость является неотъемлемым условием соблюдения регулятивных требований. Часто требуется, чтобы модели предоставляли объяснения своих прогнозов. Интерпретируемые векторы обеспечивают соблюдение этих требований, предоставляя прозрачное и понятное обоснование решений модели.

*5. Доверие и признание пользователей.* Интерпретируемые векторы повышают доверие и признание пользователей к моделям текстового анализа. Когда пользователи понимают сущность признаков, влияющих на прогнозы, они склонны больше доверять результатам модели.

*6. Анализ ошибок и улучшение модели.* Интерпретируемые векторы облегчают поиск признаков, искажающих прогнозы, а также выявляют закономерности и источники ошибок. Этот помогает уточнить модель, повысить ее точность и снизить вероятность появления ложноположительных или ложноотрицательных результатов.

В целом, интерпретируемость векторов документов играет важную роль в задачах обработки и анализа текстов, обеспечивая прозрачность, интеграцию предметных знаний, обнаружение предубеждений, соблюдение регулятивных требований, достижение доверия пользователей и улучшение модели. Далее приведен пример, описывающий свойства интерпретируемости у сгенерированных векторов документов при решении задачи поиска целевой аудитории «lookalike».

**Технология «Look-a-like»** (технология поиска целевой аудитории на основе аналогий) – это инструменты для поиска объектов (людей, компаний, устройств и

др.), схожих с существующей целевой аудиторией по характеристикам, поведению или интересам, чтобы расширить охват и повысить эффективность маркетинговых действий [32].

**Набор данных.** Используется выборка из набора данных «+7 Million Company». Этот набор данных содержит информацию о компаниях, включая название, местоположение, отрасль и ссылку на LinkedIn. Описания компаний извлекаются в процессе сбора данных с веб-сайтов (web scraping) с использованием Python-библиотеки BeautifulSoup<sup>8</sup>.

Пусть целевая компания – это финтех-стартап, специализирующийся на платежных решениях на основе блокчейна (blockchain-based payment), задача – найти другие компании с похожей деятельностью, технологиями или фокусом на рынке. Формально *задача* заключается в том, чтобы определить компании, которые наиболее релевантные деятельности целевой компании. В этом контексте деятельность компании соответствует *запросу* пользователя. *Релевантность* результатов запросу пользователя оценивается путем вычисления косинусного сходства (формула 13) между вектором представления запроса (целевой компании) и векторами представления других компаний [148].

Ниже представлено верхнеуровневое описание алгоритма построения векторов концептов с использованием предлагаемого метода *BoWC* и его применения при реализации процесса поиска релевантных компаний (рис. 4.19).

*Шаг 1.* Обработка текста – удаление ненужных символов и слов.

*Шаг 2.* Для каждого текста (описания компании) применяется алгоритм извлечения ключевых фраз *FBKE* с фильтром на основе парсера *SpaCy* (п. 3.2). Этот алгоритм работает с отдельным текстом и определяет важность ключевых фраз, на основе их семантической близости к контексту текста и частоте их появления [25]. Затем *SpaCy*-фильтр удаляет бессмысленные слова. В результате получается список ключевых фраз для каждого текста (компании), выражющий его содержание.

---

<sup>8</sup> <https://beautiful-soup-4.readthedocs.io/en/latest/>

*Шаг 3.* Выбираются уникальные ключевые фразы на уровне коллекции текстов, которые передаются алгоритму кластеризации для формирования концептов. Количество концептов определяется в соответствии с вычислительными экспериментами как  $k=200$  (п. 3.3). Использование *SpaCy* для фильтрации фраз и извлечения сущностей с последующей группировкой их в кластеры для построения концептов, повышает качество извлеченных концептов и гарантирует, что они будут специфичными для предметной области и семантически значимыми.

*Шаг 4.* Тексты выводятся в виде списка их ключевых фраз, а также выводится словарь эталонных концептов. Результаты затем сохраняются в базе данных для дальнейшего использования. В таблице 4.11 представлена часть словаря эталонных концептов, построенного на основе анализа описаний компаний.

Таблица 4.11 – Словарь эталонных концептов,  
созданный на основе описания компаний

| ID  | Метка кластера         | Ключевые фразы кластера (концепта), ранжированные по расстоянию до центроида                                                                                          |
|-----|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 34  | Blockchain payment     | ["blockchain payment solutions", "payment solutions", "blockchain platforms", "blockchain payment", "blockchain payment technology", "decentralized payment systems"] |
| 40  | fast payment           | ["instant payment", "instant payment", "instant charges systems", "fast payment platforms", "fast charges system"]                                                    |
| 53  | secure payment         | ["safe payment solutions", "protected payment apps", "encrypted payment networks", "fraud-proof payment technology", "secure payment"]                                |
| 72  | healthcare             | ["healthcare technology platforms", "healthcare solutions", "Digital health platform", "E-health platform"]                                                           |
| 85  | e-commerce payment     | ["e-commerce systems", "digital payment solutions", "e-commerce payment platforms", "e-commerce payment integrations"]                                                |
| 113 | social media marketing | ["social media marketing", "social media advertising", "social media ads", "social media campaigns", "social media promotions"]                                       |

Метка кластера – для улучшения демонстрации каждому концепту можно присвоить метку на основе простого алгоритма, основанного на парсере. Для каждого концепта алгоритм определяет корни ключевых фраз, ближайших к

центроиду кластера. Определяется наиболее часто встречающееся составное существительное во фразах кластера. Метка формируется на основе комбинации корня фразы с наиболее часто встречающимся составным существительным.

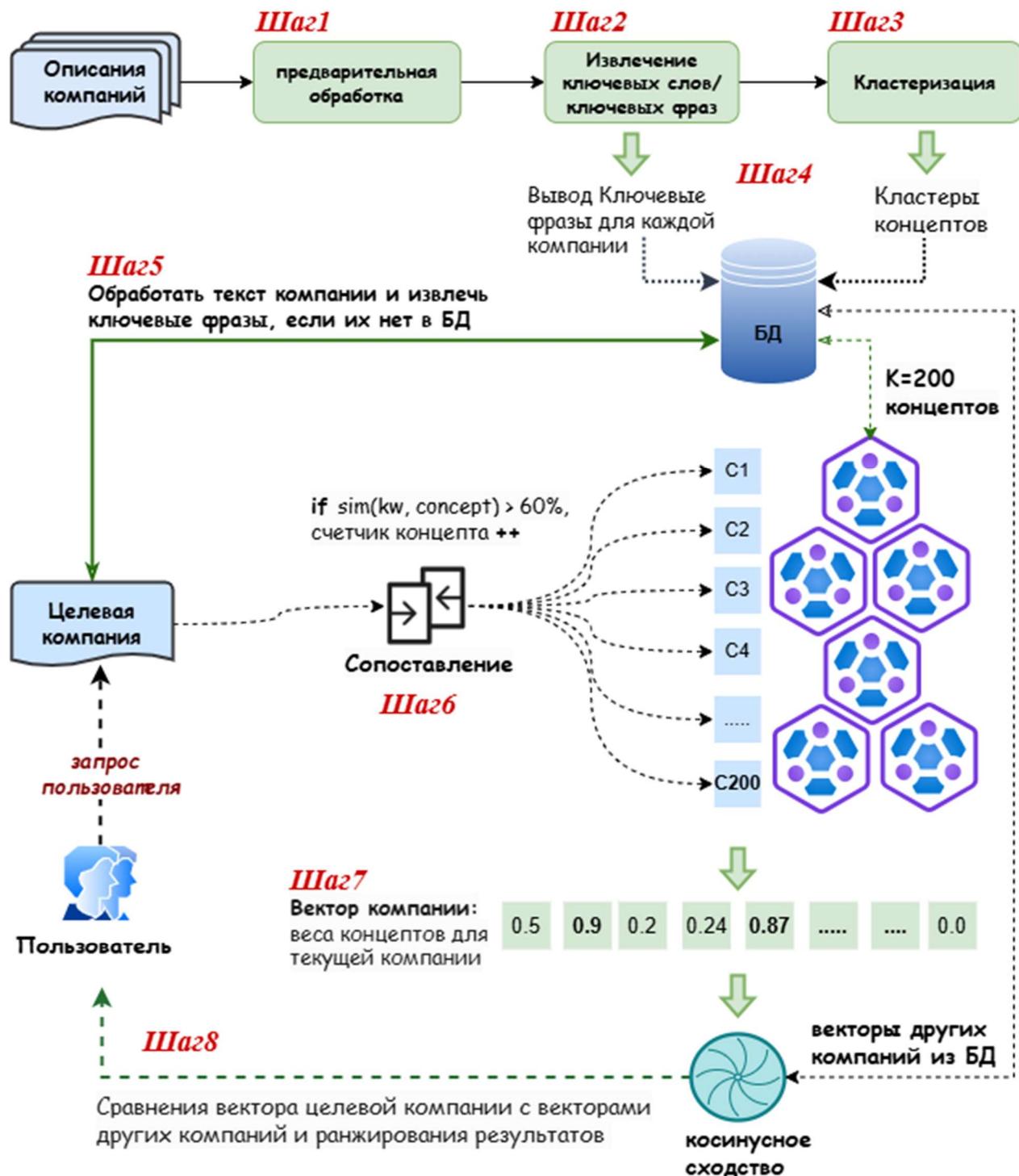


Рисунок 4.19 – Укрупнённая схема алгоритма векторизации описания компаний и реализация процесса поиска релевантных компаний

Далее показано, как векторизация на основе *BoWC* может быть применена для идентификации компаний, похожих на заданную целевую компанию. Предположим, что целевая компания – это FinTech Startup X имеет следующее описание: «*FinTech Startup X is an innovative company developing blockchain-based payment solutions. We offer fast, secure, and decentralized payments for businesses and individuals.*».

Также анализируются описания других компаний с различными сферами деятельности, например компания **A** (*BlockchainPay Inc.*): «*BlockchainPay Inc. provides decentralized payment solutions for global transactions. We use blockchain to ensure security and speed*»; компания **B** (*HealthTech Solutions*): «*HealthTech Solutions develops telemedicine platforms and AI-driven diagnostic tools for healthcare providers*»; компания **C** (*E-Commerce Pro*): «*E-Commerce Pro offers online shopping platforms with integrated digital payment solutions and advanced inventory management*».

Чтобы присвоить описание компании кластерам концептов необходимо:

*Шаг 5.* определить ключевые фразы в описании каждой компании (например, "blockchain-based payment solutions," "decentralized payments").

*Шаг 6.* Для каждой ключевой фразы вычисляется её мера косинусного сходства с центроидным вектором каждого кластера по формуле *CF-EDF* (26).

*Шаг 7.* Каждая компания представляется в виде вектора оценок сходства её фраз с каждым концептом. В таблице 4.12 представлены результаты процесса векторизации описаний компаний с использованием *BoWC*. Ненулевые значения обозначают веса концептов, присутствующих в тексте, и степень присутствия каждого из них.

*Шаг 8.* Мера косинусного сходства используется для сравнения вектора описания целевой компании с векторами описания других компаний и ранжирования результатов по их релевантности запросу пользователя (целевой компании).

Из таблицы 4.12 отметим, что компания А имеет высокую оценку сходства (0.96), потому что она также специализируется на *blockchain payment solutions* и

использует похожие технологии (например, быстрые и безопасные платежи). Компания *B* имеет очень низкий балл сходства (0.12), потому что она сосредоточена на *healthcare technology*, что выходит за рамки сферы деятельности целевой компании. Компания *C* имеет умеренную меру сходства (0.28), потому что она пересекается в таких областях, как *digital payment solutions*, но в основном сосредоточена на *e-commerce payments*.

Метод *BoWC* эффективно идентифицирует компании с похожей деятельностью, технологиями и рыночным фокусом, даже если сфера деятельности других компаний значительно отличается. Кроме того, использование алгоритма извлечения ключевых фраз на основе парсера позволяет идентифицировать понятия с общими корнями или синонимами и метками, основанными на наиболее частых составных существительных, что делает кластеры более связными и семантически значимыми.

Таблица 4.12 – Результаты векторизации описания компаний с помощью BoWC

| <b>Концепты (кластеры)</b>      | <b>Целевая компания</b> | <b>Компания</b> |          |             |
|---------------------------------|-------------------------|-----------------|----------|-------------|
|                                 |                         | <b>A</b>        | <b>B</b> | <b>C</b>    |
| Blockchain Payments             | <b>0.92</b>             | <u>0.88</u>     | 0.01     | 0.02        |
| Fast Payments                   | <b>0.75</b>             | <u>0.68</u>     | 0.02     | <u>0.42</u> |
| Payment Security                | <b>0.83</b>             | <u>0.75</u>     | 0.01     | 0.31        |
| Healthcare Technology           | 0.0                     | 0.0             | 0.91     | 0.01        |
| E-Commerce Payments             | 0.02                    | 0.02            | 0.03     | 0.85        |
| Social media                    | 0.01                    | 0.01            | 0.02     | 0.52        |
| <i>оценки схожести компаний</i> | –                       | <b>0.96</b>     | 0.12     | 0.28        |

Далее приведено подробное сравнение интерпретируемости векторов, созданных методом *BoWC*, с векторами, созданными аналогами, такими как *TF-IDF*, averaged *W2V*, *LSA* и *BERT*.

Векторы *BoWC* привязаны к понятным для человека концептам (например, "Blockchain Payments," "Fast Payments"). Каждое измерение в векторе отражает

определенный концепт, что делает легче осознание причин сходства между двумя компаниями. Метод *BoWC* явно группирует семантически связанные фразы (например, "blockchain payment solutions" и "decentralized payment systems") в один концепт "Blockchain Payments". Это гарантирует, что синонимы и связанные термины рассматриваются как часть одного кластера.

*TF-IDF* создает интерпретируемые векторы основанные на частоте терминов, поэтому каждое измерение соответствует конкретному слову или фразе. Однако *TF-IDF* не учитывает семантические связи, поэтому термины, такие как "blockchain" и "distributed ledger", рассматриваются как отдельные измерения. Вектор *TF-IDF* имеет следующий вид [blockchain: 0.5, payments: 0.4, secure: 0.3, fast: 0.2, healthcare: 0.01].

*W2V* создает плотные, низкоразмерные векторы для слов или документов. Эти векторы не являются непосредственно интерпретируемыми, так как представляют абстрактные семантические связи, а неявные концепты. Пример: "blockchain" и "distributed ledger" могут иметь похожие встраивания, но они неявно сгруппированы. Вектор *W2V* имеет следующий вид [0.12, -0.05, 0.34, ...].

*LSA* создает скрытые темы, которые частично интерпретируемые, но темы часто абстрактны и требуют ручной маркировки. Например, тема может включать "blockchain," "payments," и "security," но связь не является явной. Векторы не привязаны к явным концептам и не являются понятными для человека. Вектор *LSA* имеет следующий вид [0.02, -0.46, 0.54, ...].

*BERT* создает абстрактные векторы фиксированной длины (например, 768 измерений [0.45, -0.12, 0.78, ...]), которые захватывают семантическую информацию, но они не являются непосредственно интерпретируемыми без дополнительной постобработки (например, кластеризации или уменьшения размерности).

Метод *BoWC* предоставляет четкое, интерпретируемое и отраслево-специфичное представление, что делает его превосходным для задач, таких как поиск похожих компаний в нишевых отраслях. В дополнение к вышесказанному, согласно экспериментам в разделах 4.3–4.6, предложенный в данной работе метод

и алгоритмы достигают очень хороших результатов в задачах классификации и кластеризации, а также являются вычислительно эффективными, поскольку после создания словаря концептов векторизация является легковесной и эффективной. Это позволяет применять алгоритмы машинного обучения для организации компаний в отдельные группы с помощью кластеризации или относить их к заранее определенным концептуальным группам с помощью классификации.

В данном исследовании используется определение концепта, при котором не устанавливается тип отношений между концептами или фразами, формирующими эти концепты. Вместо этого акцент делается на оценке меры семантической близости, которая служит критерием для определения наличия связи между фразами, входящими в состав концепта. Особенность научной задачи, решаемой в этом исследовании, не требует явного определения отношений между концептами.

Тем не менее, это не исключает возможности расширения словаря концептов, чтобы включить явное определение типа отношений. Такое расширение позволит решать дополнительные задачи, такие как рекомендации, автоматизация построения и расширения онтологии и/или графа знаний [121, 120]. Каждый концепт в словаре может рассматриваться как элемент онтологии, а ключевые фразы внутри него формируют концепты, связанные отношением «часть – целое» или «является».

Эти возможности придают особое значение предлагаемому в данном исследовании словарю эталонных концептов и позволяют применять его в более широком спектре задач анализа текстов на естественном языке в системах искусственного интеллекта и машинного обучения.

#### **4.8. Выводы по разделу**

В данной главе были описаны результаты вычислительных экспериментов, направленных на проверку эффективности предложенных метода и алгоритмов генерации векторных представлений текстов. Основная цель заключалась в повышении эффективности средств интеллектуального анализа текстов на основе

разработки методов и алгоритмов представления текстов, позволяющих построить низкоразмерные векторное представление с интерпретируемыми признаками.

Вычислительные эксперименты, описанные в этой главе, подтвердили эффективность метода векторизации текстов и предложенных алгоритмов обработки и анализа текста. Было продемонстрировано, что предложенный метод векторизации текста *BoWC* генерирует интерпретируемые низкоразмерные векторные представления и, в то же время, повышает точность и эффективность инструментов классификации и кластеризации документов. Метод *BoWC* показал хорошие результаты, превзойдя группу канонических методов векторизации текстов по сокращению количества признаков, используемых для представления векторов, и повышению точности результатов в задачах классификации и кластеризации текста. Данные результаты вносят вклад в методологическую область средств обработки естественного языка, предоставляя новое решение проблем, связанных с представлением текста в низкоразмерном пространстве.

Результаты также подтвердили, что использование алгоритма построения концептов при создании словаря эталонных концептов привело к повышению эффективности методов векторизации текстов, использующих этот словарь, таких как *BoWC* и *BoC*. Это связано с применением алгоритма фильтрации ключевых фраз на основе парсера, устранением неоднозначности ключевых фраз и выбором только релевантных фраз при построении словаря концептов. Кроме того, эффективность алгоритма извлечения ключевых фраз на основе фильтра при извлечении релевантных фраз была подтверждена в экспериментах, независимых от методов векторизации текстов.

Результаты вычислительного эксперимента показали, что метод *BoWC* лучше работает с более длинными текстами. Это объясняется возможностью более эффективного извлечения большего количества информации о каждом концепте из длинных документов по сравнению с короткими.

## ЗАКЛЮЧЕНИЕ

Диссертация посвящена решению научной задачи создания моделей, алгоритмов и методов обработки и анализа текстов на естественном языке, в условиях экспоненциального роста их объёмов, что позволяет повысить эффективность средств и инструментов систем искусственного интеллекта и машинного обучения. Одной из основных моделей пространства решений при классификации и кластеризации текстовых документов в системах искусственного интеллекта является векторное представление (векторизация текстовых данных). Эти методы необходимы для преобразования текстовых данных в числовые представления, понятные машинам, что позволяет алгоритмам машинного обучения эффективно обрабатывать и анализировать тексты. Для решения поставленной задачи автором были разработаны модель, алгоритмы и метод Data mining для интеллектуальной обработки и анализа текстов на естественном языке, позволяющие снизить частоту ошибок при классификации и кластеризации текстов. Основные результаты диссертационной работы перечислены в следующих пунктах:

Проведенные исследования существующих методов решения задачи векторизации документов показали, что данные методы не позволяют построить векторные представления документов малой размерности, которые можно интерпретировать без негативного влияния на эффективность алгоритмов классификации и кластеризации с точки зрения обобщаемости результатов и возможности интерпретации логики их работы. На основе проведенных исследований также было подтверждено, что для построения векторных представлений документов необходимо использовать концепт, чтобы каждое элемент вектора соответствовало концепту, а не слову. Это представление является линейным преобразованием пространства слов в пространство концептов и считается решением проблемы размерности векторов.

В результате были разработаны модель и модифицированный метод генерации векторных представлений текстов (векторизации документов) с использованием методов извлечения ключевых фраз и алгоритмов построения

концептов на основе технологии интеллектуального анализа данных. Метод позволяет снизить размерность векторного пространства и повысить дискриминационную способность результирующих векторов признаков. Кроме того, требования к памяти для хранения векторов, генерируемых методом, значительно ниже, чем у аналогичных методов. Метод позволяет построить более информативное векторное представление с интерпретируемыми признаками.

Серия вычислительных экспериментов была проведена для оценки эффективности метода BoWC в решении задач текстового анализа, классификации и кластеризации на пяти эталонных наборах данных. Производительность метода была сравнена с группой известных методов векторизации текста, с учетом используемых размерностей векторов и точности классификации и/или кластеризации. Предложенный метод *BoWC* превзошёл большинство канонических методов по минимальному количеству признаков и максимальной точности классификации и кластеризации. В среднем предложенный автором метод векторизации текстов снижает частоту ошибок алгоритмов классификации и кластеризации документов на 0,2 – 1 % и 2 – 6 % соответственно, а также позволяет уменьшить число признаков по сравнению с конкурирующими методами. Временная сложность предложенного метода векторизации текстов в худшем случае составляет  $O(n^2)$ . (решены задачи 1 и 2, с. 8, п. 1 и 2 научной новизны, с. 9).

Разработан алгоритм извлечения и фильтрации ключевых фраз на основе применения парсера, позволяющий извлекать ключевые фразы с правильной грамматической структурой. Алгоритм показал хорошие результаты и превзошел другие известные методы в среднем на 1% по метрике MAP@K. (решена задача 4, с.8, п. 3 научной новизны, с.9).

Разработан алгоритм построения концептов из текста путем применения методов кластеризации к n-граммам (фразам) вместо униграмм. Использование n-граммы мотивировано тем, что она подтверждает значение слова и раскрывает многозначность его смысла, расширяя соседними словами имеющийся контекст. Эффективность алгоритма была подтверждена повышением однородности построенных им кластеров концептов по сравнению с аналогами, что привело к

увеличению дискриминационной способности векторов признаков. (решена задача 4, с.8, п. 4 научной новизны, с.10).

Разработано программное приложение, позволяющее использовать разработанные модель, метод и алгоритмы обработки и анализа текстов на естественном языке в системах искусственного интеллекта для снижения частоты появления ошибок в алгоритмах классификации и кластеризации с учётом условий снижения размерности векторного представления текстов и сохранения его интерпретируемости.

В целом совокупность полученных в диссертации теоретических и практических результатов позволяет сделать вывод о том, что цель исследований достигнута, сформулированная научная задача решена. Перечисленные результаты получили высокую оценку научного сообщества при аprobации и положительные рекомендации для внедрения в информационные процессы предприятий, учреждений и организаций различного профиля деятельности.

Дальнейшее развитие полученных результатов диссертационного исследования возможно в следующих основных направлениях:

1. Разработанный в данном исследовании метод векторизации текстов более эффективно работает с длинными текстовыми документами благодаря возможности извлечения достаточного количества концептов, описывающих содержимое документа. Необходимо провести исследование для разработки алгоритмов и функций, которые обеспечат баланс между концептами коротких и длинных документов.
2. Применение разработанного алгоритма построения концептов в дополнительных областях, таких как разработка профилей пользователей на основе концептов, а также автоматизация процесса построения и расширения онтологии в системах искусственного интеллекта для извлечения знаний из текстов. В качестве функций присвоения нового содержания концептам профиля пользователя или онтологии применяются функции определения весов концептов CF-EDF, CF-EDF<sub>FBKE</sub>.
3. Проведение дополнительных исследований алгоритма извлечения ключевых фраз для достижения баланса между точностью и скоростью парсера, в связи с

важностью этой проблемы для повышения эффективности анализа текстов в реальном масштабе времени. Одним из возможных направлений улучшения предложенных решений является использование атомарных признаков, таких как униграммы слов и униграммы POS-тегов вместо использования большого количества признаков, созданных вручную.

4. В данном исследовании использовались исключительно текстовые источники для извлечения концептов и устранения их неоднозначности, но это не ограничивает возможности будущих исследований по расширению словаря эталонных концептов за счёт интеграции концептов из онтологий или базы знаний.
5. Модификация разработанных метода и алгоритмов для их применения при обработке и анализе текстов на других языках.

## СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

ИИ – Искусственный интеллект

NLP – Natural language processing

ML – Machine learning

ИНС – Искусственная нейронная сеть

K-NN – K nearest neighbors

TF-IDF – Term Frequency – Inverted Document Frequency;

BoW – Bag of words

BoC – Bag of Concepts

BoWC – Bag of Weighted Concepts

NP – noun phrases

VP – Verb phrases

KP – Key phrases

KW – Keyword

SVM – Support vector machine

W2V – word2vec, Word to vector

D2V – Doc2Vec, Document to vector

**СПИСОК ЛИТЕРАТУРЫ**

1. Jordan, M. I. Machine learning: Trends, perspectives, and prospects / M.I. Jordan, T.M. Mitchell // Science. – 2015. – Vol. 349. – no 6245. – P. 255-260.
2. Sarker, I. H. Machine Learning: Algorithms, Real-World Applications and Research Directions / I.H. Sarker// SN Computer Science. – 2021. – Vol. 2. – Machine Learning. – no 3. – P. 160.
3. Al-Aswadi, F. N. Extracting semantic concepts and relations from scientific publications by using deep learning / F.N. Al-Aswadi, H.Y. Chan, K.H. Gan // International Conference of Reliable Information and Communication Technology. Springer. – 2020. – P. 374-383.
4. Petrovic, S. A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters / S. Petrovic // Proceedings of the 11th Nordic workshop of secure IT systems. – Citeseer. – 2006. – Vol. 2006. – P. 53-64.
5. Fedorenko, V. I. Use of text vectorization methods in natural language to improve the quality of content recommendations of films / V.I. Fedorenko, V.S. Kireev// Modern high technologies. – 2018. – no 3. – P. 102.
6. Manning C. D. Introduction to information retrieval / C. D. Manning, H. Schütze, P. Raghavan // Cambridge: Cambridge University Press. – 2008. – Vol. 39. – P. 234-265.
7. Mladenic, D. Machine Learning on non-homogeneous, distributed text data / D. Mladenic // Computer Science, University of Ljubljana, Slovenia. – 1998.
8. Singh, V. Feature extraction techniques for handwritten text in various scripts: a survey / V. Singh, B. Kumar, T. Patnaik// International Journal of Soft Computing Engineering. – 2013. – Vol. 3. no 1. – P. 238-241.
9. Тюрин, В. В. Дискриминантный анализ в биологии / В.В. Тюрин, С.Н. Щеглов // монография. – Краснодар: Кубанский государственный университет. – 2015.

10. Das, M. A comparative study on tf-idf feature weighting method and its analysis using unstructured dataset / M. Das, S. Kamalanathan, P. Alphonse// CEUR Workshop Proceedings. – 2021. – Vol. 2870. – P. 98-107.
11. Мансур, А. Развитие кластерного поиска документов на основе разработки методов векторизации текстов/А. Мансур, Ж. Мохаммад, Ю.А. Кравченко// Труды II научно-методической конференции НПР «Современные компьютерные технологии» (ИКТИБ ЮФУ). – 2021. – С. 28-31.
12. Kim, H. K. Bag-of-concepts: Comprehending document representation through clustering words in distributed representation / H.K. Kim, H. Kim, Cho// Neurocomputing. – 2017. – Vol. 266. – P. 336-352.
13. Мансур, А. М. Метод генерации векторов низкой размерности для представления текстовых документов / А.М. Мансур, Ж.Х. Мохаммад// Труды XIX всероссийской научной конференции молодых ученых, аспирантов и студентов «Информационные технологии, системный анализ и управление» (ИТСАУ-2021). – 2021. – С. 199-203.
14. Мансур, А. М. Векторизация текста с использованием методов интеллектуального анализа данных / А.М. Мансур, Ж.Х. Мохаммад, Ю.А. Кравченко // Известия ЮФУ. Технические науки. – 2021. – № 2 (219). – С. 154-167.
15. Mikolov, T. Efficient estimation of word representations in vector space / T. Mikolov, K. Chen, G. Corrado, J. Dean// arXiv preprint arXiv:1301.3781. – 2013.
16. Bojanowski, P. Enriching word vectors with subword information / P. Bojanowski, E. Grave, A. Joulin, T. Mikolov// Transactions of the Association for Computational Linguistics. – 2017. – Vol. 5. – P. 135-146.
17. Le, Q. Distributed representations of sentences and documents / Q. Le, T. Mikolov// International conference on machine learning. 2014. – P. 1188-1196.
18. Liu, Z. Representation Learning for Natural Language Processing / Z. Liu, Y. Lin, M. Sun. – Springer Nature. – 2023.
19. Mitra, B. Learning to match using local and distributed representations of text for web search / B. Mitra, F. Diaz, N. Craswell, – 2017. – P. 1291-1299.

20. Mitra, B. An introduction to neural information retrieval / B. Mitra, N. Craswell// Foundations Trends® in Information Retrieval. – 2018. – Vol. 13. no 1. – P. 1-126.
21. Roy, D. Using word embeddings for automatic query expansion / D. Roy, D. Paul, M. Mitra, U. Garain// arXiv preprint arXiv:.07608. – 2016.
22. Grootendorst, M. Beyond Bag-of-Concepts: Vectors of Locally Aggregated Concepts / M. Grootendorst, J. Vanschoren// Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, – 2019. – P. 681-696.
23. Rücklé, A. Concatenated power mean word embeddings as universal cross-lingual sentence representations / A. Rücklé, S. Eger, M. Peyrard, I. Gurevych// arXiv preprint arXiv:.01400. – 2018.
24. Erbas, C. A General-Purpose Machine Reasoning Engine / C. Erbas// International Conference on Artificial General Intelligence. Springer. – 2022. – P. 3-13.
25. Devlin, J. Bert: Pre-training of deep bidirectional transformers for language understanding / J. Devlin, M.-W. Chang, K. Lee, K. Toutanova// arXiv preprint arXiv:.04805. – 2018.
26. Pappagari, R. Hierarchical transformers for long document classification / R. Pappagari, P. Zelasko, J. Villalba, Y. Carmiel, N. Dehak// 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). IEEE, – 2019. – P. 838-844.
27. Adhikari, A. Docbert: Bert for document classification / A. Adhikari, A. Ram, R. Tang, J. Lin// arXiv preprint arXiv:.08398. – 2019.
28. Мансур А. М. Алгоритм на основе трансформеров для классификации длинных текстов/А. М. Мансур //Известия ЮФУ. Технические науки. – 2024. – №. 3.
29. Boubekeur, F. Concept-based indexing in text information retrieval / F. Boubekeur, W. Azzoug // arXiv preprint arXiv:1303.1703. – 2013.
30. Мансур, А. Программный модуль оптимизации работы классификатора при векторизации текста на основе биоэвристик / А. Мансур, Ж. Мохаммад, Ю.А. Кравченко, Д. Ю. Кравченко // Свидетельство регистрации программы для ЭВМ. – 12.12.2023. – № 2023687185.

31. Мансур, А. Программный модуль оптимизации извлечения ключевых слов при обработке лингвистической экспертной информации / А. Мансур, Ж. Мохаммад, Ю.А. Кравченко, К. Н. Владимировна // Свидетельство регистрации программы для ЭВМ. – 14.12.2023. – № 2023687372.
32. Popov, A. Adaptive look-alike targeting in social networks advertising / A. Popov, D. Iakovleva // Procedia computer science. – 2018. – Vol. – 136. – P. 255-264.
33. Nahm, U. Y. Text mining with information extraction / U.Y. Nahm, R.J. Mooney// Proceedings of the AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge Bases. Stanford CA, – 2002. – P. 60-67.
34. Manning, C. Foundations of statistical natural language processing / C. Manning, H. Schutze. – MIT press. – 1999.
35. Hearst, M. A. Text Tiling: Segmenting text into multi-paragraph subtopic passages / M.A. Hearst// Computational linguistics. – 1997. – Vol. 23. no 1. – P. 33-64.
36. Panchenko, A. Russe: The first workshop on russian semantic similarity / A. Panchenko, N. Loukachevitch, D. Ustalov, D. Paperno, C. Meyer, N. Konstantinova// arXiv preprint arXiv:.05820. – 2018.
37. Poelmans, J. Formal concept analysis in knowledge processing: A survey on applications / J. Poelmans, D.I. Ignatov, S.O. Kuznetsov, G. Dedene// Expert Systems with Applications. – 2013. – Vol. 40. no 16. – P. 6538-6560.
38. Poelmans, J. Text mining scientific papers: a survey on FCA-based information retrieval research / J. Poelmans, D.I. Ignatov, S. Viaene, G. Dedene, S.O. Kuznetsov// Advances in Data Mining. Applications and Theoretical Aspects: 12th Industrial Conference, ICDM 2012, Berlin, Germany, July 13-20, 2012. Proceedings 12. Springer. – 2012. – P. 273-287.
39. Игнатов, Д. И. Анализ формальных понятий: от теории к практике / Д.И. Игнатов, Р.Э. Яворский// Доклады всероссийской научной конференции АИСТ. – 2012. – Том. 12. – Р. 3-15.
40. Чусовлянов, Д. С. Машинное обучение для определения тональности и классификации текстов на несколько классов / Д.С. Чусовлянов// Москва. – 2014.

41. Kutuzov, A. Word vectors, reuse, and replicability: Towards a community repository of large-text resources / A. Kutuzov, M. Fares, S. Oepen, E. Velldal// Proceedings of the 58th Conference on Simulation and Modelling. Linköping University Electronic Press. – 2017. – P. 271-276.
42. Craswell, N. ORCAS: 20 million clicked query-document pairs for analyzing search / N. Craswell, D. Campos, B. Mitra, E. Yilmaz, B. Billerbeck// Proceedings of the 29th ACM International Conference on Information & Knowledge Management. – 2020. – P. 2983-2989.
43. Blei, D. M. Latent dirichlet allocation / D.M. Blei, A.Y. Ng, M.I. Jordan// Journal of machine Learning research. – 2003. – Vol. 3. no Jan. – P. 993-1022.
44. Jurafsky, D. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition / D. Jurafsky, J. H. Martin. – 2000.
45. Marcheggiani, D. Encoding sentences with graph convolutional networks for semantic role labeling / D. Marcheggiani, I. Titov// arXiv preprint arXiv:1703.04826. – 2017.
46. Peters, M. E. Deep contextualized word representations / M.E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer// arXiv preprint arXiv:05365. – 2018.
47. Степанов, Р. Г. Технология Data Mining: интеллектуальный анализ данных / Р.Г. Степанов// Казань: КГУ. – 2008.
48. Фридман, О. В. Data Mining-методы и алгоритмы, краткий обзор / О.В. Фридман // Труды Кольского научного центра РАН. – 2021. – Том. 12. № 5 (12). – P. 91-103.
49. Han, J. Data mining: concepts and techniques / J. Han, J. Pei, H. Tong. – Morgan kaufmann. – 2022.
50. Мансур, А. Перспективы развития процессов информационного поиска на основе применения методов data mining/ А. Мансур, Ж. Мохаммад// Труды VI всероссийской научно-технической конференции молодых ученых, аспирантов, магистрантов и студентов «Фундаментальные и прикладные аспекты

- компьютерных технологий и информационной безопасности». – Таганрог. – 2020. – С. 317-320.
51. Акофф, Р. Л. Менеджмент в XXI веке. Преобразование корпорации / Р.Л. Акофф // Томск: Изд-во Том. ун-та. – 2006.
  52. West, D. M. Big data for education: Data mining, data analytics, and web dashboards / D.M. West// Governance studies at Brookings. – 2012. – Vol. 4. no 1. – P. 1-10.
  53. Aggarwal, C. C. Data mining: the textbook. Vol. 1 / C. C. Aggarwal. – New York: springer, 2015. – T. 1. – C. 1.
  54. Рукавицын, А. Н. Разработка модели классификации веб-страниц с использованием методов интеллектуального анализа данных / А.Н. Рукавицын// Известия СПбГЭТУ «ЛЭТИ». – 2016. – №. 4. – С. 12-20.
  55. Nikam, S. S. A comparative study of classification techniques in data mining algorithms / S.S. Nikam//Oriental Journal of Computer Science Technology. – 2015. – Vol. 8. no 1. – P. 13-19.
  56. Wu, X. Top 10 algorithms in data mining / X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P.S. Yu//Knowledge information systems. – 2008. – Vol. 14. – P. 1-37.
  57. Xue, H. SVM: Support vector machines / H. Xue, Q. Yang, S. Chen//The top ten algorithms in data mining. – 2009. Vol. 6. no 3. – P. 37-60.
  58. Chomboon, K. An empirical study of distance metrics for k-nearest neighbor algorithm / K. Chomboon, P. Chujai, P. Teerarassamee, K. Kerdprasop, N. Kerdprasop// Proceedings of the 3rd international conference on industrial application engineering. – 2015. – Vol. 2.
  59. Hartigan, J. A. Algorithm AS 136: A k-means clustering algorithm / J.A. Hartigan, M.A. Wong// Journal of the royal statistical society. series c. – 1979. – Vol. 28. no 1. – P. 100-108.
  60. Ершов, К. С. Анализ и классификация алгоритмов кластеризации / К.С. Ершов, Т.Н. Романова// Новые информационные технологии в автоматизированных системах. – 2016. – № 19. – P. 274-279.

61. Kodinariya, T. M. Review on determining number of Cluster in K-Means Clustering / T.M. Kodinariya, P.R. Makwana//International Journal. – 2013. – Vol. 1. no 6. – P. 90-95.
62. Arthur, D. K-means++ the advantages of careful seeding / D. Arthur, S. Vassilvitskii// Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. – 2007. – P. 1027-1035.
63. Schubert, E. Accelerating spherical k-means / E. Schubert, A. Lang, G. Feher//International Conference on Similarity Search and Applications. Springer. – 2021. – P. 217-231.
64. Hotho, A. A brief survey of text mining / A. Hotho, A. Nürnberg, G. Paß//Journal for Language Technology Computational Linguistics. – 2005. – Vol. 20. no 1. – P. 19-62.
65. Большаякова, Е. И. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика / Е.И. Большаякова, Э.С. Клышинский, Д.В. Ландэ, А.А. Носков, О.В. Пескова, Е.В. Ягунова. – 2011.
66. Бенгфорт, Б. Прикладной анализ текстовых данных на Python / Б. Бенгфорт, Р. Билбро, Т. Охеда//Машинное обучение и создание приложений обработки естественного языка. СПб.: Питер. – 2019.
67. Hassani, H. Text mining in big data analytics / H. Hassani, C. Beneki, S. Unger, M.T. Mazinani, M.R. Yeganegi//Big Data Cognitive Computing. – 2020. – Vol. 4. no 1. – P. 1.
68. Balakrishnan, V. Stemming and lemmatization: A comparison of retrieval performances / V. Balakrishnan, E. Lloyd-Yemoh. – 2014.
69. Bengforth, B. Applied text analysis with Python: Enabling language-aware data products with machine learning/ B. Bengforth, R. Bilbro, T. Ojeda// O'Reilly Media, Inc. – 2018.
70. Řehůřek, R. Gensim—statistical semantics in python / R. Řehůřek, P. Sojka//Retrieved from genism. org. – 2011.

71. Srinivasa-Desikan, B. Natural Language Processing and Computational Linguistics: A practical guide to text analysis with Python, Gensim, spaCy, and Keras / B. Srinivasa-Desikan // Packt Publishing Ltd. – 2018.
72. Vasiliev, Y. Natural language processing with Python and spaCy: A practical introduction / Y. Vasiliev // No Starch Press. – 2020.
73. Bird, S. NLTK: the natural language toolkit / S. Bird//Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions. – 2006. – P. 69-72.
74. Qi, Y. Salient context-based semantic matching for information retrieval / Y. Qi, J. Zhang, W. Xu, J. Guo // EURASIP Journal on Advances in Signal Processing. – 2020. – Vol. 2020. – P. 1-17.
75. Porter, M. F. An algorithm for suffix stripping / M.F. Porter//Program. – 1980. – Vol. 14. no 3. – P. 130-137.
76. Willett, P. The Porter stemming algorithm: then and now / P. Willett//Program. – 2006. – Vol. 40. no 3. – P. 219-223.
77. Aggarwal, C. C. Mining text data / C. C. Aggarwal, C. Zhai Springer Science & Business Media. – 2012. P.– ISBN 1-4614-3223-5.
78. Turney, P. D. From frequency to meaning: Vector space models of semantics / P.D. Turney, P. Pantel//Journal of artificial intelligence research. – 2010. – Vol. 37. – P. 141-188.
79. Lavrenko, V. Relevance-based language models / V. Lavrenko, W.B. Croft//ACM SIGIR Forum. ACM New York, NY, USA. – 2001. – Vol. 51. – P. 260-267.
80. Abubakar, H. D. Sentiment classification: Review of text vectorization methods: Bag of words, Tf-Idf, Word2vec and Doc2vec / H.D. Abubakar, M. Umar, M.A. Bakale//SLU Journal of Science Technology. – 2022. – Vol. 4. no 1 & 2. – P. 27-33.
81. Singh, R. Text similarity measures in news articles by vector space model using NLP / R. Singh, S. Singh // Journal of The Institution of Engineers (India): Series B. – 2021. – Vol. 102. – P. 329-338.
82. Harris, Z. S. Distributional structure / Z.S. Harris//Word. – 1954. – Vol. 10. no 2-3. – P. 146-162.

83. Zhang, Y. Understanding bag-of-words model: a statistical framework / Y. Zhang, R. Jin, Z.-H. Zhou//International Journal of Machine Learning Cybernetics. – 2010. – Vol. 1. no 1-4. – P. 43-52.
84. Aizawa, A. An information-theoretic perspective of tf–idf measures / A. Aizawa//Information processing management. – 2003. – Vol. 39. no 1. – P. 45-65.
85. Salton, G. Term-weighting approaches in automatic text retrieval / G. Salton, C. Buckley//Information processing management. – 1988. – Vol. 24. no 5. – P. 513-523.
86. Eminagaoglu, M. A new similarity measure for vector space models in text classification and information retrieval / M. Eminagaoglu//Journal of Information Science. – 2022. – Vol. 48. no 4. – P. 463-476.
87. Sebastiani, F. Text categorization / F. Sebastiani//Encyclopedia of database technologies and applications. IGI Global. – 2005. – P. 683-687.
88. Kobayashi, M. Vector space models for search and cluster mining / M. Kobayashi, M. Aono//Survey of Text Mining II: Clustering, Classification, Retrieval. – 2008. – P. 109-127.
89. Kadhim, A. I. Survey on supervised machine learning techniques for automatic text classification / A.I. Kadhim//Artificial Intelligence Review. – 2019. – Vol. 52. no 1. – P. 273-292.
90. Sebastiani, F. Machine learning in automated text categorization / F. Sebastiani//ACM computing surveys. – 2002. – Vol. 34. no 1. – P. 1-47.
91. Raghavan, V. V. A critical analysis of vector space model for information retrieval / V.V. Raghavan, S.M. Wong // Journal of the American Society for Information Science. – 1986. – Vol. 37. no 5. – P. 279-287.
92. Shahmirzadi, O. Text similarity in vector space models: a comparative study / O. Shahmirzadi, A. Lugowski, K. Younge//2019 18th IEEE international conference on machine learning and applications (ICMLA). IEEE. – 2019. – P. 659-666.
93. Alghamdi, R. A survey of topic modeling in text mining / R. Alghamdi, K. Alfalqi//Int. J. Adv. Comput. Sci. Appl. – 2015. – Vol. 6. no 1.

94. Barde, B. V. An overview of topic modeling methods and tools / B.V. Barde, A.M. Bainwad//2017 International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE. – 2017. – P. 745-750.
95. Hofmann, T. Probabilistic latent semantic indexing / T. Hofmann//Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. – 1999. – P. 50-57.
96. Brand, M. Incremental singular value decomposition of uncertain data with missing values / M. Brand//Computer Vision–ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31. 2002 Proceedings, Part I 7. Springer. – 2002. – P. 707-720.
97. Almeida, F. Word embeddings: A survey / F. Almeida, G. Xexéo//arXiv preprint arXiv:.09069. – 2019.
98. Mikolov, T. Efficient estimation of word representations in vector space / T. Mikolov, K. Chen, G. Corrado, J. Dean//arXiv preprint arXiv:.1301.3781. – 2013.
99. Pennington, J. Glove: Global vectors for word representation / J. Pennington, R. Socher, C.D. Manning//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). – 2014. – P. 1532-1543.
100. Kusner, M. From word embeddings to document distances / M. Kusner, Y. Sun, N. Kolkin, K. Weinberger//International conference on machine learning. – 2015. – P. 957-966.
101. Savigny, J. Emotion classification on youtube comments using word embedding / J. Savigny, A. Purwarianti//2017 international conference on advanced informatics, concepts, theory, and applications (ICAICTA). IEEE. – 2017. – P. 1-5.
102. Kang, B.-Y. Document indexing: a concept-based approach to term weight estimation / B.-Y. Kang, S.-J. Lee//Information processing management. – 2005. – Vol. 41. no 5. – P. 1065-1080.
103. Hu, X. Exploiting wikipedia as external knowledge for document clustering / X. Hu, X. Zhang, C. Lu, E.K. Park, X. Zhou//Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. – 2009. – P. 389-396.

104. Mehanna, Y. S. A semantic conceptualization using tagged bag-of-concepts for sentiment analysis / Y.S. Mehanna, M.B. Mahmuddin//IEEE Access. – 2021. – Vol. 9. – P. 118736-118756.
105. Täckström, O. An evaluation of bag-of-concepts representations in automatic text classification / O. Täckström//Recall. – 2005.
106. Huang, E. H. Improving word representations via global context and multiple word prototypes / E.H. Huang, R. Socher, C.D. Manning, A.Y. Ng//Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). – 2012. – P. 873-882.
107. Mahalakshmi, P. An art of review on Conceptual based Information Retrieval / P. Mahalakshmi, N.S. Fatima//Webology Journal. – 2021. – Vol. 18. – P. 51-61.
108. Musat, C. Concept-based topic model improvement / C. Musat, J. Velcin, M.-A. Rizoiu, S. Trausan-Matu//Emerging intelligent technologies in industry. Springer. – 2011. – P. 133-142.
109. Voorhees, E. M. Using WordNet to disambiguate word senses for text retrieval / E.M. Voorhees//Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval. – 1993. – P. 171-180.
110. Boughanem, M. A new factor for computing the relevance of a document to a query / M. Boughanem, I. Mallak, H. Prade//International Conference on Fuzzy Systems. IEEE. – 2010. – P. 1-6.
111. Pelevina, M. Making sense of word embeddings / M. Pelevina, N. Arefyev, C. Biemann, A. Panchenko//arXiv preprint arXiv:03390. – 2017.
112. Li, Y. Sentence similarity based on semantic nets and corpus statistics / Y. Li, D. McLean, Z. Bandar, J. O’Shea, K. Crockett, Data Engineering // IEEE Transactions on Knowledge. – 2006. – Vol. 18. – P. 1138-1150.
113. Yatsko, V. Methods for dictionary generation / V. Yatsko//Automatic Documentation Mathematical Linguistics. – 2012. – Vol. 46. – P. 195-201.
114. Rose, S. Automatic keyword extraction from individual documents / S. Rose, D. Engel, N. Cramer, W. Cowley//Text mining: applications theory. – 2010. – Vol. 1. – P. 1-20.

115. Campos, R. YAKE! Keyword extraction from single documents using multiple local features / R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, A. Jatowt//Information Sciences. – 2020. – Vol. 509. – P. 257-289.
116. Qingyun, Z. Keyword extraction method for complex nodes based on TextRank algorithm / Z. Qingyun, F. Yuansheng, S. Zhenlei, Z. Wanli//2020 International Conference on Computer Engineering and Application (ICCEA). IEEE. – 2020. – P. 359-363.
117. Mansour, A. Generating Conceptual Semantic Vectors Based on Key Phrase Extraction Techniques / A. Mansour, J. Mohammad, Y. Kravchenko//2023 International Russian Automation Conference (RusAutoCon). IEEE. – 2023. – C. 374-379.
118. Mansour, A. Text vectorization method based on concept mining using clustering techniques / A. Mansour, J. Mohammad, Y. Kravchenko//2022 VI International Conference on Information Technologies in Engineering Education (Inforino). IEEE. – 2022. – C. 1-10.
119. Мансур, А. М. Модифицированный метод построения семантического представления текста на основе методов кластеризации и взвешивания терминов / А.М. Мансур, Ж.Х. Мохаммад, Д.Ю. Кравченко, Ю.А. Кравченко // Труды XII международной научно-технической конференции «Технологии разработки информационных систем (ТРИС-2022)». – Таганрог: 2022. – С. 94-100.
120. Мансур, А. Перспективы развития процессов автоматического построения онтологий на основе применения методов оценки семантической близости/ А. Мансур, Ж. Мохаммад // Труды XVIII всероссийской научной конференции молодых ученых, аспирантов и студентов «Информационные технологии, системный анализ и управление (ИТСАУ-2020). – Таганрог. – 2020. – С. 121-124.
121. Мансур, А. Использование методов веб-майнинга в автоматическом построении онтологий/ А. Мансур, Ж. Мохаммад// Труды VI всероссийской научно-технической конференции молодых ученых, аспирантов, магистрантов и студентов «Фундаментальные и прикладные аспекты компьютерных технологий и информационной безопасности». – Таганрог. – 2020. – С. 321-324.

122. Mansour, A. Harnessing Key Phrases in Constructing a Concept-Based Semantic Representation of Text Using Clustering Techniques / A. Mansour, J. Mohammad, Y. Kravchenko, D. Kravchenko, N. Silega// Lecture Notes in Computer Science. – LNCS. – Vol. 14335. – 2023. – P. 190-201.
123. Мансур А. М., Метод извлечения ключевых фраз на основе новой функции ранжирования / А.М. Мансур, Ж.Х. Мохаммад, Ю.А. Кравченко, В.В. Бова // Информационные технологии. – 2022. – Том. 28. № 9. – С. 465-474.
124. Мансур, А. М. Метод автоматического извлечения ключевых слов / А.М. Мансур, Ж.Х. Мохаммад, Д.Ю. Кравченко, Ю.А. Кравченко // Труды международного научно-технического конгресса «Интеллектуальные системы и информационные технологии – 2022» («ИС & ИТ-2022», «IS&IT'22»). Научное издание. – Таганрог: Изд-во Ступина С.А., Т.1. – 2022. – С. 90-97.
125. Мансур, А. М. Перспективы применения метода извлечения ключевых фраз FBKE в задачах персонализации веб-контента / А.М. Мансур, Ж.Х. Мохаммад, Ю.А. Кравченко// Труды XX всероссийской научной конференции молодых ученых, аспирантов и студентов «Информационные технологии, системный анализ и управление (ИТСАУ-2022)». – Таганрог. – 2022. – С. 206.
126. Мансур, А. М. Модифицированный метод устранения неоднозначности смысла слов, основанный на методах распределенного представления / А.М. Мансур, Ю.А. Кравченко, Ж.Х. Мохаммад//Известия Южного федерального университета. Технические науки. – 2021. № 3 (220). – С. 92-101.
127. Mansour, A. Algorithm for Optimization of Keyword Extraction Based on the Application of a Linguistic Parser / A. Mansour, J. Mohammad, D. Kravchenko, Y. Kravchenko & N. Pavlov // Informatics and Automation. – Vol. 23. – 2024. – no. 2. – P. 467-494.
128. Barker, K. Using noun phrase heads to extract document keyphrases / K. Barker, N. Cornacchia//Advances in Artificial Intelligence: 13th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, AI 2000 Montéal, Quebec, Canada, May 14–17, 2000 Proceedings 13. Springer. – 2000. – P. 40-52.

129. Kaur, J. Effective approaches for extraction of keywords / J. Kaur, V. Gupta//International Journal of Computer Science Issues. – 2010. – Vol. 7. no 6. – P. 144.
130. Siddiqi, S. Keyword and keyphrase extraction techniques: a literature review / S. Siddiqi, A. Sharan//International Journal of Computer Applications. – 2015. – Vol. 109. no 2.
131. Richards, T. Getting Started with Streamlit for Data Science: Create and deploy Streamlit web applications from scratch in Python / T. Richards// Packt Publishing Ltd, – 2021. p.– ISBN 1-80056-320-5.
132. Voron, F. Building Data Science Applications with FastAPI: Develop, manage, and deploy efficient machine learning applications with Python. Building Data Science Applications with FastAPI // F. Voron // Packt Publishing Ltd, – 2023.
133. Flanagan, D. JavaScript / D. Flanagan, P. Matilainen // Anaya Multimedia, – 2007. p.– ISBN 84-415-2202-2.
134. Adeshina, A. A. Building Python Web APIs with FastAPI: A fast-paced guide to building high-performance, robust web APIs with very little boilerplate code. Building Python Web APIs with FastAPI / A. A. Adeshina // Packt Publishing Ltd, – 2022.
135. Somasundar, A. MongoDB integration with Python and Node. js, Express. js / A. Somasundar, M. Chilakarao, B.R.K. Raju, S.K. Behera, C.V. Ramana, P.K. Sethy// 2024 Fourth International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT). IEEE, – 2024. – P. 1-5.
136. Greene, D. Practical solutions to the problem of diagonal dominance in kernel document clustering / D. Greene, P. Cunningham//Proceedings of the 23rd international conference on Machine learning. – 2006. – P. 377-384.
137. Sabbah, T. Modified frequency-based term weighting schemes for text classification / T. Sabbah, A. Selamat, M.H. Selamat, F.S. Al-Anzi, E.H. Viedma, O. Krejcar, H. Fujita//Applied Soft Computing. – 2017. – Vol. 58. – P. 193-206.
138. Lan, M. Supervised and traditional term weighting methods for automatic text categorization / M. Lan, C.L. Tan, J. Su, Y. Lu//IEEE transactions on pattern analysis and machine intelligence. – 2008. – Vol. 31. no 4. – P. 721-735.

139. Соколов, П. В. Сравнительный анализ методов кластеризации текстовой информации / П.В. Соколов, Е.Н. Каруна//Вестник Тюменского государственного университета. – 2019. № 7. – Р. 180.
140. Van Rijsbergen, C. Information retrieval: theory and practice / C. Van Rijsbergen// Proceedings of the joint IBM/University of Newcastle upon tyne seminar on data base systems. – 1979. – Vol. 79. – P. 1-14.
141. Rosenberg, A. V-measure: A conditional entropy-based external cluster evaluation measure / A. Rosenberg, J. Hirschberg// Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL). – 2007. – P. 410-420.
142. Zhang, X. ConceptEVA: Concept-based interactive exploration and customization of document summaries / X. Zhang, J. Li, P.-W. Chi, S. Chandrasegaran, K.-L. Ma // Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. – 2023. – P. 1-16.
143. Nivre, J. Pseudo-Projective Dependency Parsing / J. Nivre, J. Nilsson//Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05) / Citation Key: nivre-nilsson-2005-pseudo. Association for Computational Linguistics. – 2005. – P. 99-106.
144. Honnibal, M. An Improved Non-monotonic Transition System for Dependency Parsing / M. Honnibal, M. Johnson//Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing / Citation Key: honnibal-johnson-2015-improved. Association for Computational Linguistics. – 2015. – P. 1373-1378.
145. Dozat, T. Deep biaffine attention for neural dependency parsing / T. Dozat, C.D. Manning//arXiv preprint arXiv:01734. – 2016.
146. Qi, P. Stanza: A Python natural language processing toolkit for many human languages / P. Qi, Y. Zhang, Y. Zhang, J. Bolton, C.D. Manning//arXiv preprint arXiv:07082. – 2020.
147. Hulth, A. Improved automatic keyword extraction given more linguistic knowledge / A. Hulth// Proceedings of the 2003 conference on Empirical methods in natural language processing. – 2003. – P. 216-223.

148. Мансур, А. Релевантность контента как основной критерий ранжирования при поиске информации / А. Мансур, Ж. Мохаммад // Труды XVIII всероссийской научной конференции молодых ученых, аспирантов и студентов «Информационные технологии, системный анализ и управление (ИТСАУ-2020)». – Таганрог. – 2020. – С. 118-121.

**ПРИЛОЖЕНИЕ №1**

**АКТЫ О ВНЕДРЕНИИ РЕЗУЛЬТАТОВ РАБОТЫ**



АКТ

об использовании в учебном процессе

Института компьютерных технологий и информационной безопасности

Южного федерального университета

результатов кандидатской диссертации А.М. Мансур «Модель, метод и алгоритмы Data mining для интеллектуальной обработки и анализа текстов на естественном языке»

Я, нижеподписавшийся, руководитель образовательной программы (ОП) «Разработка информационных систем и web-приложений» 09.04.01 Информатика и вычислительная техника Кулиев Э.В., к.т.н., доцент кафедры САПР имени В.М. Курейчика, составил настоящий акт о том, что в учебном процессе кафедры САПР имени В.М. Курейчика, Института компьютерных технологий и информационной безопасности ЮФУ используются следующие результаты, полученные в кандидатской диссертации Мансур А.М.:

- Методы и алгоритмы обработки и анализа текстов на естественном языке;
- Математическая модель векторного представления текстовых документов;
- Метод генерации векторных представлений для решения задач классификации и кластеризации текстов;
- Алгоритмы извлечения и фильтрации ключевых фраз на основе применения функций парсера для разметки частей речи;
- Алгоритм построения концептов из семантически близких слов и фраз с использованием функций парсера для решения задач автоматического построения онтологии и графа знаний.
- Программное приложение для решения задач представления и классификации текстов и выделения из них ключевых фраз.

Указанные результаты используются при проведении следующих курсов в ИКТИБ: «Технологии Big Data», «Методы машинного обучения при построении информационных систем», «Онтологические модели в информационных системах».

Внедрение в учебный процесс ряда теоретических и практических результатов диссертационной работы Мансур А.М. позволило повысить качество подготовки магистров.

Руководитель ОП «Разработка информационных систем и web-приложений»,  
к.т.н., доцент

Э. В. Кулиев

**ОБЩЕСТВО С ОГРАНИЧЕННОЙ  
ОТВЕТСТВЕННОСТЬЮ ИТ-ЭФФЕКТ**

ИНН 7729594500, КПП 77290100, ОГРН 1087746189893  
119517, Москва, Нежинская ул., д.8, корп.2, пом.6  
(985) 997-5591

УТВЕРЖДАЮ  
Генеральный директор  
ООО «ИТ-ЭФФЕКТ»

С.А. Сафонов

13 декабря 2024 г.



Запрос о переносе д/с в рамках лицевого счёта

**АКТ**  
**о внедрении результатов диссертационной работы на соискание ученой степени кандидата технических наук Мансур Али Махмуд в обществе с ограниченной ответственностью «ИТ-ЭФФЕКТ» (ООО «ИТ-ЭФФЕКТ»)**

Комиссия в составе:

председатель комиссии

- генеральный директор С.А. Сафонов;

члены комиссии:

- Технический директор Ю.А. Ермилов;

- Начальник отдела маркетинга А.И. Бурдаев.

составили настоящий акт о том, что научные результаты диссертационной работы аспиранта Южного федерального университета А. Мансур по теме «Модель, метод и алгоритмы Data mining для интеллектуальной обработки и анализа текстов на естественном языке», представленной на соискание ученой степени кандидата технических наук, использованы в ООО «ИТ-ЭФФЕКТ» при построении системы семантического поиска на основе правил путем их применения при улучшении представления текстов в векторном пространстве и решении задач автоматического построения онтологии и графа знаний.

В частности, были использованы следующие конкретные научные результаты кандидатской диссертации А. Мансур:

- Методы и алгоритмы обработки и анализа текстов на естественном языке;

- Математическая модель векторного представления текстовых документов;

- Метод генерации векторных представлений для решения задач

классификации и кластеризации текстов;

- Алгоритмы извлечения и фильтрации ключевых фраз на основе применения функций парсера для разметки частей речи;

- Алгоритм построения концептов из семантически близких слов и фраз с использованием функций парсера для решения задач автоматического построения онтологии и графа знаний.

Внедренные результаты диссертационного исследования позволили повысить качество и унификацию проектных решений.

Председатель комиссии:

Генеральный директор

С.А. Сафонов

Члены комиссии:

Технический директор

Начальник отдела маркетинга

Ю.А.Ермилов;  
А.И.Бурдаев.

**ПРИЛОЖЕНИЕ №2**

**СВИДЕТЕЛЬСТВА О ГОСУДАРСТВЕННОЙ РЕГИСТРАЦИИ ПРОГРАММ  
ДЛЯ ЭВМ**

## РОССИЙСКАЯ ФЕДЕРАЦИЯ



## СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2023687372

Программный модуль оптимизации извлечения  
ключевых слов при обработке лингвистической  
экспертной информацииПравообладатель: *федеральное государственное автономное  
образовательное учреждение высшего образования  
«Южный федеральный университет» (RU)*Авторы: *Кравченко Юрий Алексеевич (RU), Мухаммад  
Жуман (AB), Мансур Али (RU), Кулиева Нина  
Владимировна (RU)*

Заявка № 2023686104

Дата поступления **30 ноября 2023 г.**

Дата государственной регистрации

в Реестре программ для ЭВМ **14 декабря 2023 г.**Руководитель Федеральной службы  
по интеллектуальной собственности

Ю.С. Зубов

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ  
Сертификат 429bbafe3853164ba96f83b73b4aa7  
Владелец Зубов Юрий Сергеевич  
Действителен с 10.05.2023 по 02.08.2024

РОССИЙСКАЯ ФЕДЕРАЦИЯ



# СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

**№ 2023687185**

**Программный модуль оптимизации работы  
классификатора при векторизации текста на основе  
биоэвристик**

Правообладатель: *федеральное государственное автономное  
образовательное учреждение высшего образования  
«Южный федеральный университет» (RU)*

Авторы: *Кравченко Даниил Юрьевич (RU), Кравченко Юрий  
Алексеевич (RU), Мансур Али (RU), Мухаммад Жуман  
(RU)*

Заявка № **2023686057**

Дата поступления **30 ноября 2023 г.**

Дата государственной регистрации

в Реестре программ для ЭВМ **12 декабря 2023 г.**

Руководитель Федеральной службы  
по интеллектуальной собственности

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ  
Сертификат 429bb6afef3853164ba96f83b73b4aa7  
Владелец Зубов Юрий Сергеевич  
Действителен с 10.05.2023 по 02.08.2024

*Ю.С. Зубов*

