Сравнительный анализ применяемых технологий обработки естественного языка для улучшения качества классификации цифровых документов

А.К. Марков, Д.О. Семёночкин, А.Г. Кравец, Т.А. Яновский

***Аннотация*— Тегирование цифровых электронных документов - это процесс назначения метаданных или меток (тегов) документам с целью упрощения их организации, поиска и управления. Этот процесс имеет большое значение для эффективного управления информацией и обеспечения доступности документов в цифровой среде. Выбор метода и технологии тегирования зависит от конкретных потребностей организации или пользователя. Часто используется комбинация различных методов для достижения наилучших результатов в управлении цифровыми электронными документами. В данной статье выполняется сравнительный анализ технологий обработки естественного языка для улучшения качества классификации цифровых документов на примере технических образовательных документов. В статье рассматриваются методы, используемые в предварительной обработке документов, способы улучшения предварительной обработки, а также проводится вычислительный эксперимент, по результатам которого определяются показатели улучшения полноты и точности классификации данных.**

***Ключевые слова*— Тегирование, Классификация, Метаданные, Обработка, Категоризация, Тэг.**

I. Введение

Обработка естественного языка (Natural Language Processing, NLP) - это технология, применяемая в машинном обучении позволяющая электронно - вычислительным машинам интерпретировать и понимать человеческий язык. В наши дни организации имеют большие объемы данных разных форматов, полученные из разных источников. Это могут быть электронные письма, оцифрованные служебные документы, новостные ленты социальных сетей и многое
другое. Для автоматической обработки таких данных они используют программное обеспечение NLP[1].

В начале 2010-х годов все большую популярность стали обретать обработчики естественного языка и чат-боты. Изначально работа с поисковыми системами напоминала лишь работу с предметным указателем — простым инструментом, для использования которого не требовалось особых навыков. Однако довольно быстро он стал более интеллектуальным и начал воспринимать поисковые запросы, все более близкие к естественному языку.

Далее еще больше усложнялась функциональность автозаполнения в смартфонах. Посередине поисковой строки зачастую указывалось конкретно интересующее пользователя слово[2].

Обработчик естественного языка также называют конвейером из-за того, что он включает в себя несколько этапов обработки, принимая текст на естественном языке в одном конце системы и выдавая преобразованный результат в другом ее конце[2].

Что же делает системы NLP многофункциональными и точными в решении ежедневных задач? В основе NLP - систем находятся алгоритмы глубокого обучения. Это нейронные сети, которые могут преобразовывать необработанные данные в нужный результат, не требуя какой-либо многообразной ручной настройки алгоритма. Например, в случае, когда условный пользователь напишет отзыв на естественном человеческом языке, а компьютер абсолютно точно ответит, «Является ли положительным этот отзыв?». В таких задачах, как машинный перевод или речевое распознавание, глубокое обучение уже превысило уровень человеческих возможностей[3].

Существует множество технологий обработки естественного языка, и выбор наиболее подходящей может быть сложным. В данной статье будет проведен сравнительный анализ применяемых технологий обработки естественного языка для улучшения качества классификации цифровых документов. Будут рассмотрены различные методы и алгоритмы и проанализированы их преимущества и недостатки. Результаты данного исследования помогут определить наиболее эффективные технологии обработки естественного языка для улучшения классификации цифровых документов.

Научная новизна работы представлена сравнительным анализом различных технологий обработки естественного языка, которые применяются для улучшения качества классификации цифровых документов не только путем рассмотрения отдельных методов и алгоритмов, но и их сравнением с точки зрения эффективности, точности и применимости к различным типам электронных цифровых документов. Исследование, представленное в статье основано на систематическом анализе существующих методов и технологий в области обработки естественного языка. В статье проведен сравнительный анализ различных подходов, оценены их эффективность и точность.

Статья содержит шесть разделов. В разделе 1 рассказывается о целях предварительной обработки документов при их классификации, описывается влияние предварительной обработки на точность классификации, а также выявляется необходимость улучшения предварительной обработки естественного языка при задаче классификации. Раздел 2 обозревает

использование OCR в задачах классификации цифровых документов: принципы работы, выбор движка, предобработку документов перед OCR, оценку точности, выводы о возможном улучшении. В разделе 3 приведено описание датасета для предварительной обработки. Датасет содержит четыре типа документов: описания образовательных программ, аннотации к учебным планам, программы практик, приказы. В четвертом разделе описаны методы, используемые в предварительной обработке, такие как токенизация и обработка знаков препинания, лемматизация и стемминг, удаление стоп-слов, обработка регистра, обработка синонимов и антонимов, обработка отрицания. Пятый раздел рассказывает о способах улучшения предварительной обработки с использованием NLP, такие как использование контекстных моделей для лемматизации и стемминга, использование эмбеддингов для учета семантической близости, использование моделей машинного обучения для определения стоп-слов, использование моделей глубокого обучения для обработки регистра. В шестом разделе проводится вычислительный эксперимент и сравниваются результаты улучшения обработки. Результаты, полученные в статье, планируется использовать при разработке механизма тегирования электронных цифровых документов, на базе АО «ВНИКТИнефтехимоборудование».

II. Обработка естественного языка (NLP) в задачах классификации цифровых документов

A. Цели предварительной обработки документов при задаче их классификации

Классификация цифровых документов играет ключевую роль в различных областях, таких как информационный поиск, анализ настроений и рекомендательные системы. Эффективная классификация цифровых документов способствует оптимизации управления информацией, повышает точность поисковых систем и улучшает качество работы пользователей. Однако качество классификации цифровых документов во многом зависит от подготовки и предварительной обработки данных, полученных в текстовом формате. Предварительная обработка текста напрямую влияет на точность передачи как содержания, так и стиля языка, что является основополагающим фактором для извлечения значимых выводов из текстовых данных[4]. Одной из основных проблем является наличие постороннего или нерелевантного текста в цифровых документах, что может нарушить точность обучения текстовых классификаторов. Эта проблема усугубляется распространенностью сканированных цифровых документов, которые часто содержат встроенные изображения или отсканированный текст, что требует применения надежных методов для точного извлечения и предварительной обработки текста.

B. Влияние предварительной обработки на точность классификации

Сфера предварительной обработки текста в обработке естественного языка тесно связана с его существенным влиянием на точность классификации, что подчеркивается в последних научных исследованиях. Стратегическое применение методов предварительной обработки, включая токенизацию, удаление стоп-слов, стемминг и лемматизацию, оказалось полезным для повышения общей эффективности моделей классификации. Например, анализ отзывов в Интернете [8] показал критическую роль этапов предварительной обработки, включая POS-тегирование, частоты n-грамм, стемминг и фильтрацию стоп-слов, в формировании эффективности моделей обнаружения спама, причем выбор соответствующих этапов предварительной обработки существенно повышает общую точность алгоритмов классификации с использованием методов SVM и NB.

Аналогичным образом, в области анализа медицинских текстов [9] выполнение таких этапов предварительной обработки, как токенизация, исправление ошибок и нормализация, оказалось ключевым фактором, влияющим на надежность процесса классификации, причем исследования выявляют существенное влияние таких этапов, как нормализация и исправление ошибок, на достижение значительной точности классификации при анализе историй болезни аллергических пациентов.

Кроме того, в исследовании [10], посвященном идентификации автора, особое внимание уделяется комплексной оценке подходов к представлению текста и задачам предварительной обработки, подчеркивается значимость конкретных методов предварительной обработки, таких как включение стоп-слов и выборочное использование стемминга, для повышения эффективности моделей идентификации автора.

Эти научные работы в совокупности подчеркивают первостепенную роль предварительной обработки текста в повышении точности классификационных моделей, причем правильный выбор и реализация этапов предварительной обработки вносят существенный вклад в повышение точности классификации и общей эффективности задач обработки естественного языка.

C. Необходимость улучшения предварительной обработки естественного языка при задаче классификации

К сожалению, любой текст в его исходном виде совершенно непригоден для анализа — его необходимо предварительно обработать. Требования к препроцессингу и очистке данных могут сильно варьироваться в зависимости от задачи, которую пытаются решить с помощью NLP-моделей. В тексте могут присутствовать различные шумы, такие как пунктуация, числа, специальные символы и т.д. Их удаление позволяет повысить качество и точность классификации. Также текст документа необходимо стандартизировать, путем приведения различных форм одинаковых слов к одному виду, что значительно влияет на увеличение качества классификации. Векторизация текста позволяет представить его в виде числовых признаков, которые могут быть использованы алгоритмами машинного обучения. Однако, простая векторизация может упускать некоторую важную информацию о контексте слова. Например, модель Bag-

of-Words[12] не учитывает порядок слов в предложении. Улучшенные методы векторизации, такие как Word2Vec[13] или GloVe[14], учитывают контекст и могут помочь в задаче классификации. В некоторых случаях, тексты могут содержать неструктурированные данные, такие как URL-адреса, электронные письма или коды программ. Необходимо провести специальную обработку таких данных, чтобы они не повлияли на результат классификации. В результате улучшения предварительной обработки, модели машинного обучения могут получать более точные и эффективные результаты классификации текстов[11].

III. Описание датасета для предварительной обработки

A. Описание процесса сбора документов с ресурсов

В качестве исходных данных для обучения модели были отобраны цифровые документы, специфичные для технических документов, связанных с процессом обучения в высших учебных заведениях Российской Федерации. В соответствии с законом №273-ФЗ "Об образовании в Российской Федерации" от 29 декабря 2012 года, веб-сайты образовательных учреждений, включая вузы, обязаны содержать определенный перечень информации и копий документов, размещенных на официальных онлайн-платформах организаций. Согласно этому законодательному акту, веб-сайты обязаны иметь раздел "Сведения об образовательной организации", который включает в себя разделы, такие как "Основные сведения", "Структура и органы управления образовательной организацией", "Документы", "Образование", "Образовательные стандарты и требования", "Руководство педагогическим (научно-педагогическим) составом", "Материально-техническое обеспечение и оснащенность образовательного процесса", "Страницы со стипендиями и мерами поддержки для обучающихся", "Платные образовательные услуги", "Финансово-хозяйственная деятельность", "Вакантные места для приема (перевода) обучающихся", "Доступная среда" и "Международное сотрудничество".

В качестве источников данных были выбраны 8 российских ВУЗов. Из раздела "Образование" каждого учебного заведения, а конкретно из таблицы с основными сведениями о реализуемых образовательных программах, выбирались следующие документы: "Описание образовательной программы", "Аннотации к рабочим программам дисциплин" и "Программы практик". Также из раздела "Документы" были использованы различные приказы и приложения к приказам, связанные с обучающим процессом или иными аспектами деятельности вуза. Итоговая выборка документов составила 385 документов в формате PDF и DOCX.

B. Описание типов извлеченных документов

Рассмотрим подробнее 4 выбранные категории документов. Основная образовательная программа высшего образования" (ООП ВО) представляет собой документ, который определяет требования к освоению в рамках определенной образовательной программы высшего образования. Содержание ООП ВО индивидуально для конкретного учебного заведения, но обычно включает следующие ключевые элементы:

1. Наименование образовательной программы: указание на специализацию, направление, или профиль, к которому относится программа.
2. Цели и задачи программы
3. Описание структуры программы: информация о количестве и последовательности обязательных и факультативных курсов, практик, а также об их длительности и последовательности.
4. Перечень основных дисциплин, которые студенты должны изучить для успешного освоения программы, с указанием их краткого содержания и целей изучения.
5. Описание методов оценки знаний и умений обучающихся, а также оценки качества образовательной программы.

Некоторые ООП содержат большие объемы табличной информации (рисунок 1)

Таблица 1

Соответствие компетенций (ОПК и ПК) обобщенным трудовым функциям (ОТФ), трудовым функциям (ТФ), содержащихся в профессиональных стандартах (ПС)

№ п/п	Наименование ПС	ОТФ	ТФ	ОПК и ПК из ФГОС ВО	Тип задач профессиональной деятельности (ПД) из актуализированных ФГОС ВО
1	40.011 Специалист по научно-исследовательским и опытно-конструкторским разработкам	B Проведение научно-исследовательских и опытно-конструкторских разработок при исследовании самостоятельных тем	Проведение патентных исследований и определение характеристик продукции (услуг) (B/01.6)	ОПК-1 ОПК-2 ОПК-3 ПК-10 ПК-11 ПК-12 ПК-13	Научно-исследовательский
			Проведение работ по обработке и анализу научно-технической информации и результатов исследований (B/02.6)	ОПК-1 ОПК-2 ОПК-3 ПК-10 ПК-11 ПК-12 ПК-13	
		C Проведение научно-исследовательски	Осуществление научного руководства проведением	ОПК-1 ОПК-2 ОПК-3 ПК-10	

Рис. 1. Часть документа ООП

Аннотация к рабочей программе дисциплины содержит описание преподаваемых дисциплин: название дисциплины, основные задачи изучения дисциплины, потенциальные результаты и виды контроля.

АННОТАЦИЯ
РАБОЧЕЙ ПРОГРАММЫ ДИСЦИПЛИНЫ

История

Дисциплина «История» относится к базовой части Блока 1 «Дисциплины (модули)».

Основной целью освоения дисциплины «История» является гуманитарная подготовка специалистов, изучение политических, социально-экономических и культурных аспектов истории России с точки зрения современных подходов к анализу явлений и процессов.

Основными задачами изучения дисциплины являются:

– ознакомить студентов с основными этапами исторического развития;
– научить анализировать исторические документы, факты, события;
– научить использовать полученные знания для оценки современного политического и экономического развития России, решения практических задач.

Процесс изучения дисциплины направлен на формирование следующих результатов обучения:

Знать:

– закономерности и этапы исторического процесса, основные события и процессы мировой и отечественной истории (факты, даты, события, имена исторических деятелей и их место в истории).

Уметь:

– занимать активную гражданскую позицию, ориентироваться в мировом историческом процессе, анализировать процессы и явления, происходящие в обществе, применять понятийно-категориальный аппарат, основные законы гуманитарных наук в профессиональной деятельности, логически верно, аргументировано и ясно строить устную и письменную речь, реферировать научную литературу, применять полученные знания для интеллектуального развития, повышения культурного уровня, профессиональной

Рис. 2. Часть аннотации рабочей программой дисциплины

Программы практик содержат описание практических мероприятий, которые должны быть выполнены студентами в рамках программы, включая информацию об их продолжительности и основных целях.

ПРОГРАММА ПРАКТИКИ

Шифр	Наименование практики
Б2.О.01(У)	Учебная изыскательская геодезическая практика

Код направления подготовки/ специальности	08.03.01
Направление подготовки/ специальность	Строительство
Наименование ОПОП (направленность/профиль)	Промышленное и гражданское строительство
Год начала реализации ОПОП	2021
Уровень образования	бакалавриат
Форма обучения	Очная, заочная
Год разработки/обновления	2021

Рис. 3. Часть программы практики

Приказы, размещаемые на сайте образовательной организации, представляются с заголовком, номером и датой публикации, содержащим краткое описание изменений, цели и задачи приказа, а также сроки и порядок их выполнения. Они также содержат инструкции для студентов и сотрудников, контактную информацию ответственных лиц, сводную таблицу изменений и ссылки на соответствующие правовые акты и так далее.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

«31» 10 2020 г. **П Р И К А З** Казань № 01-03/ 948а

Об утверждении типовой формы договора об оказании платных образовательных услуг по основным образовательным программам высшего образования иностранным гражданам

В целях установления единого подхода при заключении с иностранными гражданами договоров об оказании платных образовательных услуг, совершенствования договорной работы в КФУ, руководствуясь Уставом КФУ, **п р и к а з ы в а ю:**

1. Утвердить типовую форму договора об оказании платных образовательных услуг по основным образовательным программам высшего образования иностранным гражданам в соответствии с приложением 1 к настоящему приказу (далее – Договор).

Рис. 4. Часть приказа с сайта образовательной организации

C. Описание процесса извлечения текстовых данных из полученных документов

Для дальнейшего использования полученных документов в экспериментах необходимо извлечь из них текст для формирования корпуса текстов, который затем можно подготовить для загрузки в модель. Результирующий корпус документов содержит два типа файлов: word (docx, doc) и pdf. Подход к извлечению текста из обоих типов файлов будет индивидуальным.
Для извлечения текста из word файлов была использована библиотека python-docx для языка python. Извлеченный текстовый контент был добавлен к результирующему датасету.

PDF-файлы, если рассматривать внутреннее строение, состоят из различных компонентов, включая элементы LTFigure и LTTextContainer. LTFigure представляет собой изображение, а LTTextContainer и его производные содержат текстовые метки. Для взаимодействия с структурой PDF-файла использовалась библиотека pdfminer для языка python.

Алгоритм извлечения включает в себя несколько основных этапов, начиная с получения содержимого страницы. Если страница состоит из одного элемента LTFigure, что свидетельствует о наличии отсканированной страницы, то алгоритм конвертирует страницу в формат JPG, а затем использует инструмент оптического распознавания символов Tesseract OCR (Optical Character Recognition) для извлечения текста. Являясь OCR-движком с открытым исходным кодом, Tesseract обеспечивает эффективное извлечение текстового содержимого из изображений, предоставляя надежный механизм для управления отсканированными PDF-файлами в процессе извлечения. На рисунке 5 представлена часть отсканированного pdf документа.

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего профессионального образования
«Уральский федеральный университет
имени первого Президента России Б.Н.Ельцина»

12.05.2011 **ПРИКАЗ** 330/03
Екатеринбург

О присоединении УрГУ к УрФУ

Во исполнение приказа Министерства образования и науки Российской Федерации от 02.02.2011 г. № 155 «О реорганизации федерального государственного автономного образовательного учреждения высшего профессионального образования «Уральский федеральный университет имени первого Президента России Б.Н.Ельцина» и Государственного образовательного учреждения высшего профессионального образования «Уральский государственный университет им. А.М. Горького», на основании свидетельства Инспекции Федеральной налоговой службы России по Кировскому району г. Екатеринбурга от 12.05.2011 г. серия 66 № 006935444 о внесении записи в Единый государственный реестр юридических лиц о прекращении деятельности присоединенного юридического лица – УрГУ, свидетельства Инспекции Федеральной налоговой службы России по Кировскому району г. Екатеринбурга от 12.05.2011 г. серия 66 № 006935445 о внесении записи в Единый государственный реестр юридических лиц о реорганизации юридического лица в форме присоединения – УрФУ, в связи с утверждением новой редакции Устава УрФУ приказом Министерства образования и науки Российской Федерации от 04.05.2011 г. № 1585 и в соответствии с передаточным актом о

Рис. 5 Часть отсканированного приказа

Для страниц, содержащих объекты LTTextContainer, алгоритм напрямую извлекает текстовое содержимое. Извлеченный текстовый контент был добавлен к результирующему датасету.

После успешного извлечения текста из различных цифровых документов создается структурированный набор данных, включающий извлеченный текст и соответствующие метки классов документов. Результирующий датасет имеет формат csv файла и размер 53 мегабайта.

IV. МЕТОДЫ ИСПОЛЬЗУЕМЫЕ В ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКЕ

A. Токенизация и обработка знаков препинания

Для извлечения токенов из документа необходимо выполнить ряд сложных операций со строками, которые не могут быть выполнены с помощью базовых функций языков программирования. Необходимо разделить знаки препинания от слов, такие как кавычки, и расшифровать сокращения, например, we’ll. После этого нужно использовать регулярные выражения для объединения слов с похожим значением во время стемминга[2]. Кроме того, слова можно разделить на более мелкие части, такие как слоги, приставки, суффиксы и графемы, которые имеют свой собственный смысл и тональность[15]. Однако, эти подходы не всегда сохраняют всю информацию входных данных, и NLP-специалистам нужно уметь настраивать токенизатор для извлечения максимального объема информации из текста для конкретного приложения[2].

Токенизация в NLP является частным случаем сегментирования текста, где документ разбивается на более мелкие куски - токены или слова, а также знаки препинания. Этот процесс включает в себя сложные операции со строками, такие как разделение знаков препинания от слов, расшифровка сокращений и использование регулярных выражений для объединения слов с похожим значением. Несмотря на то, что некоторая информация может быть потеряна при использовании этого подхода, NLP-специалистам

необходимо уметь настраивать токенизатор для максимального извлечения информации из текста для конкретного приложения[2][3].

Токенизация является первым шагом в конвейере NLP и может оказать значительное влияние на остальные этапы. Полученные в результате токенизации числовые векторы могут быть использованы для машинного обучения, а также для более сложных решений и поведения. Одним из наиболее распространенных способов использования векторов, созданных при помощи токенизации, является поиск документов[2][16].

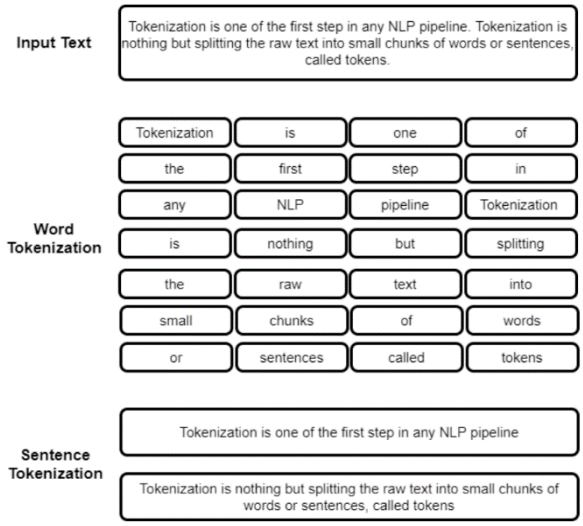

Рис. 6. Пример токенизации текста на слова и предложения

Таким образом, анализируя слова, присутствующие в тексте, можно легко интерпретировать смысл текста. Получив список слов, также можно использовать статистические инструменты и методы, чтобы получить больше информации о тексте. Например, использовать количество слов и частоту слов, чтобы определить важное слово в этом предложении или документе[16][17].

B. Лемматизация и стемминг

Леммы (lemmas) представляют собой корневые формы слов. Лемматизация (lemmatization) - это процесс сворачивания токенов до соответствующих лемм, который может быть полезен для уменьшения размерности векторного представления[18].

Стемминг - это один из методов лемматизации, который заключается в обрезании окончаний слов по заданным правилам. Существуют различные стеммеры, такие как стеммер Портера[19] и SnowBall, которые часто используются в пакетах с открытым исходным кодом.

Стеммеры работают быстрее и требуют меньших наборов данных, но они более склонны к ошибкам и обрабатывают больше слов, сокращая информационное содержание текста значительно сильнее, чем лемматизаторы. Обе технологии уменьшают словарный запас и увеличивают неоднозначность текста, но лемматизаторы работают лучше, сохраняя максимально возможную полезную информацию на основе использования слова в контексте и его предполагаемого значения.

Стеммеры обрезают слова, в то время как лемматизаторы проводят более глубокий анализ морфологии слов. Лемматизация возвращает базовую форму слова, сохраняя при этом максимально возможную полезную информацию. Для создания словарей и поиска правильной формы слова требуются глубокие лингвистические знания. Лемматизация является более интеллектуальной операцией, чем стемминг, и помогает формированию лучших возможностей машинного обучения.

Original	Stemming	Lemmatization
New	New	New
York	York	York
is	is	be
the	the	the
most	most	most
densely	dens	densely
populated	popul	populated
city	citi	city
in	in	in
the	the	the
United	Unite	United
States	State	States

Рис. 7. Сравнение лемматизации и стемминга

На рисунке 7 представлены сравнительные результаты лемматизации и стемминга, которые описал в своей статье Francisco Elia[21].

Преимущества стемминга заключаются в простоте реализации и скорости работы. Компромисс здесь заключается в том, что выходные данные могут содержать неточности, хотя они могут быть несущественными для некоторых задач, таких как индексирование текста. Вместо этого лемматизация дает лучшие результаты, выполняя анализ, который зависит от части речи слова, и создавая настоящие словарные слова. В результате лемматизацию сложнее реализовать, и она медленнее по сравнению с стеммингом.

C. Обработка стоп-слов

Стоп-слова являются общими словами на любом языке, которые встречаются часто, но не несут в себе глубокого смысла. Подвыборка, также известная как игнорирование стоп-слов, улучшает точность анализа. Игнорирование неинформативных слов, таких как "это", помогает избежать избыточности в корпусе. Математически это достигается путем игнорирования слова w_i в последовательности слов в корпусе с вероятностью:

$$1 - \sqrt{\frac{t}{f(w_i)}} \qquad (1)$$

В данной формуле t – это константа, определяющая верхний порог частоты слова, а $f(w_i)$ – частота w_i в корпусе. Подвыборка эффективно уменьшает частоту стоп-слов, улучшая баланс набора данных[3].

Крупнейшей Open-Source базой стоп - слов является

база StopWords ISO[22], собранная Gene Diaz[23]. Набор стоп-слов на русском языке, содержит 559 стоп-слов[24]. В рамках исследования проводимого в разделе 5, этот набор будет частично преобразован, под тематику обрабатываемых документов.

D. Обработка регистра

Обработка регистра (case folding) заключается в объединении разных вариантов написания слова, отличающихся только регистром букв. В тексте могут встречаться слова, написанные с прописной буквы в начале предложения или целиком прописными буквами для выделения.[2] Нормализация регистра слов и букв помогает уменьшить размер словаря и обобщить конвейер NLP. Этот метод объединяет слова, имеющие одинаковое значение и написанные одинаково, в один токен.

Однако, иногда регистр слов имеет значение. Часто прописные буквы указывают на имена собственные, имена людей, места или предметы. Если распознавание именованных объектов важно, то необходимо различать имена собственные от других слов. Ненормализованный регистр увеличит размер словаря, объем памяти и время обработки в два раза. Может потребоваться больше тренировочных данных для конвейера NLP, чтобы достичь точного общего решения. Как и в любом конвейере машинного обучения, маркированный набор данных должен быть «представительным» для всех возможных векторов признаков, включая варианты с различным регистром[25].

V. Способы улучшения предварительной обработки с использованием NLP

A. Использование контекстных моделей для лемматизации и стемминга Large References

Грамматические признаки, такие как части речи, могут содержать дополнительную информацию о языке. Внедрение грамматик и парсеров для создания облегченных синтаксических структур может значительно улучшить качество модели. Для получения информации о языке, на котором написано предложение, необходим набор грамматических правил, определяющих компоненты верно оформленных предложений на этом языке. Грамматика представляет собой набор правил, описывающих разделение синтаксических единиц на составляющие их элементы в языке[26]. Несколько примеров таких синтаксических категорий указаны на рисунке 8:

Символ	Синтаксическая категория
S	Предложение (Sentence)
NP	Именное словосочетание (Noun Phrase)
VP	Глагольное словосочетание (Verb Phrase)
PP	Предложное словосочетание (Prepositional Phrase)
DT	Определяющее слово (Determiner)
N	Существительное (Noun)
V	Глагол (Verb)
ADJ	Прилагательное (Adjective)
P	Предлог (Preposition)
TV	Переходный глагол (Transitive Verb)
IV	Непереходный глагол (Intransitive Verb)

Рис. 8. Примеры синтаксических категорий

Грамматика играет важную роль в определении правил синтаксической структуры предложений на определенном языке. Например, контекстно-свободная грамматика позволяет определить правила объединения частей речи в осмысленные фразы. Это позволяет анализировать и понимать структуру предложений на различных уровнях, от словосочетаний до целых предложений. Важно учитывать грамматические признаки, такие как части речи, при анализе текста, чтобы обеспечить точность и полноту обработки языковых данных.

B. Использование эмбеддингов для учета семантической близости

За последние годы термин «эмбеддинг» стал широко используемым, но его первоначальное появление связано с обработкой текстов на естественных языках. В контексте NLP эмбеддинг описывает процесс или результат преобразования языковых элементов, таких как слова, предложения, параграфы или целые тексты, в числовые векторы[27].

Томаш Миколов и его команда в 2013 году разработали гипотезу локальности[28], в этой гипотезе слова, используемые в идентичных контекстах, получают более приближенные значения. Для получения этих свойств, слова нужно представить в виде числовых векторов в высоко размерном векторном пространстве. Эта идея изображена на известной картинке, которая часто используется в публикациях и лекциях об эмбеддингах.[27].

(Миколов, NAACL HLT, 2013)

Рис. 9. Гипотеза Т. Милкова

Мы соберем набор данных для обучения, который будет использоваться в автоэнкодере. Этот тип модели называется Skip Gram, где на входе центральное слово, а на выходе - его контекст. В отличие от этого, CBOW использует контекст для предсказания центрального слова[29].

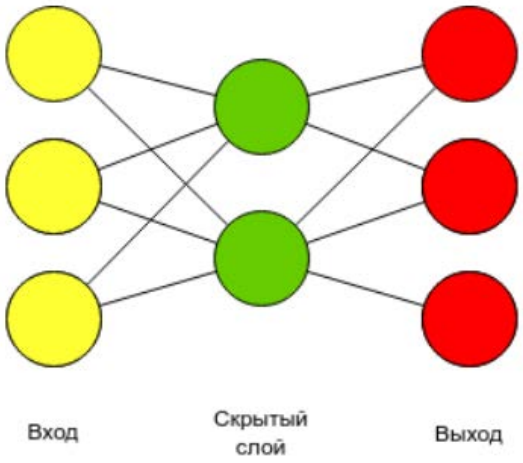

Рис. 10. Модель Skip Gram

Вместо использования Bag of words, можно передать эмбеддинг в классификатор. Для этого каждому слову в строке присваивается индекс из словаря, и затем для каждого слова ставится его эмбеддинг по индексу. Получается двумерный вектор размером [длина строки] X [размер эмбединга]. Этот вектор передается в сверточные сети, которые уже используют двумерные слои. Таким образом, получается новая нейронная сеть, которая принимает на вход индексы слов и отправляет результат в классификатор. Эту сеть можно обучать, включая обучение эмбеддинга, который можно использовать готовый или обучить самостоятельно.

C. Использование моделей машинного обучения для определения стоп-слов

Модели машинного обучения могут быть использованы для определения стоп-слов, которые являются наиболее часто встречающимися и малозначимыми словами в тексте.

Одним из подходов к определению стоп-слов с использованием моделей машинного обучения является обучение классификатора на размеченных данных. Для этого необходимо иметь набор текстовых документов, в которых стоп-слова уже помечены.

VI. Экспериментальные результаты

A. Описание классификатора

Для того чтобы оценить влияние методов предварительной обработки на качество классификации, был проведен эксперимент. В этом эксперименте качество классификации оценивалось на двух различных наборах данных: один содержал оригинальные тексты цифровой технической документации, а другой - тексты, прошедшие предварительную обработку. В таких экспериментах выбор подходящего классификатора является ключевым моментом, поскольку он существенно влияет на выявление эффектов предварительной обработки.

Многие современные исследователи сходятся во мнении, что модели типа трансформер является самыми продвинутыми в области обработки естественного языка [30-33]. Такие модели стали популярным выбором для решения различных задач обработки естественного языка, в том числе и для классификации текстов. Эти модели обладают рядом преимуществ как для классификации текстов, так и для области обработки естественного языка в целом. Они эффективно отражают контекстные связи между словами в предложении, что позволяет улавливать нюансы смысла. Кроме того, трансформеры эффективно работают с дальними зависимостями и способны изучать сложные закономерности на больших объемах данных.

История моделей - трансформеров берет свое начало с представления архитектуры типа трансформер в 2017 году в работе [34]. До появления трансформеров для решения задач обработки языки обычно использовались рекуррентные нейронные сети (РНС). Однако трансформеры вытеснили РНС благодаря их способности параллельно обрабатывать длинные последовательности, что привело к сокращению времени обучения.

Ярким примером трансформаторной модели является BERT (Bidirectional Encoder Representations from Transformers) [35]. Модель BERT, выпущенная компанией Google в 2018 году, позволила существенно продвинуться вперед в решении широкого спектра задач обработки естественного языка, включая классификацию текстов. В процессе обучения BERT использует модель маскированного языка, которая учится предсказывать маскированные слова в предложении на основе окружающего контекста.

Несмотря на значительные успехи в области обработки естественного языка, одним из основных недостатков BERT является ограниченный размер входных данных - 512 токенов. Это ограничение становится особенно критичным для задачи классификации текста [36], так как текст при изучении набора данных было обнаружено, что среднее количество токенов в одном документе из обоих наборов превышает значение в 512. Так как сокращение каждого документа до длины в 512 токенов может свести на нет усилия по предварительной обработке, в качестве классификатора был выбран Longformer, который построен на архитектуре трансформера и поддерживает входной контент длиной до 4096 токенов.

Longformer - это современная модель обработки длинных текстов, разработанная исследователями из Facebook AI[37]. Она специально разработана для решения задач, требующих понимания и классификации длинных документов. Являясь новым расширением BERT, Longformer решает эту проблему за счет использования нового механизма внимания, называемого вниманием "скользящего окна". Он заменяет стандартный механизм самовнимания, используемый в BERT, на скользящее окно для эффективной работы с длинными контекстами. Благодаря использованию скользящего окна модель фокусирует свое внимание на подмножестве лексем за один раз, а не на всей входной последовательности. Такой подход значительно снижает вычислительные затраты при работе с длинными текстами. Кроме того, в Longformer добавлены два механизма глобального внимания: "глобальное внимание" и "глобально-локальное внимание". Глобальное внимание охватывает все лексемы в последовательности, обеспечивая глобальный контекст для модели. Глобально-локальное внимание, с другой стороны, обращает внимание на подмножество лексем в зависимости от их важности,

позволяя модели фокусироваться на критически важных частях документа.

Longformer используется в различных областях NLP, где требуется обработка больших текстов. Например, в медицине Longformer используется для анализа больших объемов клинических текстов[40], что позволяет улучшить точность диагностики и лечения. В юридической области Longformer используется для обработки больших документов [41][42], таких как судебные решения и договоры, что позволяет ускорить процесс анализа и принятия решений. Кроме того, Longformer используется для обработки данных из социальных сетей, что позволяет анализировать большие объемы текстовых сообщений и выявлять тренды и паттерны в общении пользователей [43].

После выбора классификатора, было проведено 2 эксперимента, которые включали в себя обучение модели и получение результатов для каждого из двух датасетов. Все полученные текстовые документы были оптимально обрезаны таким образом, чтобы последовательность конкатенированных лексем не превышала 4096 лексем. Предварительно обученный Longformer настраивается с помощью модели kazzand/ru-longformer-base-4096[38] для поддержки русского языка с потерями по перекрестной энтропии, при этом используются следующие параметры: batch size 16, оптимизатор Адама с начальной скоростью обучения 2,5×10-5, уменьшающейся в 0,5 раза каждую эпоху. Общее количество эпох - 5. Для обучения использовался pytorch. и пакет transformers из HuggingFace для реализации Longformer.

В качестве метрик качества модели использовались показатель F1, Precision и Recall[39]. Precision - это показатель того, сколько правильных положительных предсказаний было сделано моделью из всех положительных предсказаний. Она сосредоточена на точности положительных предсказаний модели и показывает, насколько надежен классификатор при определении положительного класса. Он рассчитывается как отношение истинно положительных результатов (TP) к сумме истинно положительных и ложноположительных результатов (FP).

Recall, также известный как чувствительность или частота истинных положительных результатов, измеряет, насколько хорошо классификатор способен идентифицировать все положительные экземпляры из числа действительно положительных. Он рассчитывается как отношение истинно положительных результатов к сумме истинно положительных и ложноотрицательных результатов (FN).

Показатель F1 представляет собой среднее гармоническое значение между показателями Precision и Recall. Он представляет собой единую метрику, которая уравновешивает точность и отзыв. Показатель F1 рассчитывается как 2 * ((Precision * Recall) / (Precision + Recall)) и находится в диапазоне от 0 до 1, причем 1 означает наилучшую возможную производительность. Показатель F1 полезен в ситуациях, когда наблюдается дисбаланс классов, т.е. один класс имеет значительно больше экземпляров, чем другие.

B. Описание действий над улучшенным датасетом

В рамках работы по улучшению качества датасета, были частично произведены операции описанные в разделе 4. Первичным этапом улучшения стало использование контекстной модели для стеммера. В качестве стеммера рассматривались два основных варианта PorterStemmer[19] и SnowballStemmer[20], решающим фактором в выборе стеммера стало его качество работы с русскоязычными входными данными. Был проведен отдельный эксперимент, на котором SnowballStemmer показал более качественные результаты по сравнению с PorterStemmer. В качестве контекстной модели в стеммер был добавлен метод listen, определяющий унифицирующую абстрактую функциональность. Результатом этого метода является ответ, если он необходим, или параметр None, если нет, а также оценка достоверности - число с плавающей запятой в диапазоне от 0 до 1.

Далее было выполнено преобразование текста в численную форму(embedding).Использовался модуль CountVectorizer, который входит в пакет scikit-learn и преобразовывает входной текст в матрицу, значениями которой является количество вхождений q данного ключа(слова) в текст. Таким образом, мы получили матрицу, размерность которой будет равна количеству всех слов, умноженных на количество документов. И элементами матрицы будут числа, которые означают, сколько раз всего слово встретилось в тексте.

Для получения набора стоп слов, было решено отказаться от обучения собственной модели, и взять уже готовый набор, полученный также при помощи глубокого обучения[22][24]. Причиной послужила невозможность получения данных для обучения в сравнимом количестве с количеством данных на которых обучались, уже существующие модели определения стоп - слов.

C. Результаты эксперимента

Обучение/тестирование модели проводилось на графическом процессоре с 16 гб видеопамяти. Полученные в результате эксперимента данные представлены в таблице 1.

Таблица 1. Экспериментальные оценки качества классификации

Датасет	F1 score	Precision	Recall
Исходный	0.9055	0.9261	0.9048
Обработанные	0.9524	0.9524	0.9524

Источник: Составлено авторами

D. Выводы по результатам эксперимента.

Оценка F1 — это метрика оценки машинного обучения, которая сочетает в себе оценки точности и полноты. По результатам эксперимента видно, что эта метрика улучшилась на 0,069 или на 5,179%. Точность

увеличилась на 0,0263 или на 2,839%. Recall изменилась также в большую сторону на 0,0476 или на 5,26%. Также стоит отметить, что все оценки качества перешли порог в 0.95, таким образом, до абсолютного значения 1 остается менее 5%. Таким образом можно однозначно утверждать, что использование улучшенной предварительной обработки с использованием NLP значительно повышает оценки качества классификации цифровых документов.

VII. Заключение

Для решения актуальной задачи повышения качества классификации цифровых документов в данном исследовании использован набор инструментов обработки естественного языка. Был проведен краткий обзор литературы, для выяснения актуальных методов и средств для обработки текстовых данных. Для оценки качества классификации был собран набор данных, состоящий из технической документации и включающий 4 класса документов. В качестве классификатора для этой цели был выбран Longformer - модель на основе архитектуры трансформера, специально предназначенная для обработки объемных документов. Экспериментальная схема включала в себя оценку качества многоклассовой классификации с использованием двух наборов данных: исходного и обработанного методами NLP. Анализ результатов эксперимента однозначно показал, что применение средств обработки языка к набору данных значительно повысило точность предсказания классов. Данные средства обработки языка планируется использовать при тегировании цифровых документов института АО “ВНИКТИнефтехимоборудование”, при создании единого электронного хранилища цифровых документов.

Библиография

[1] Amazon URL: https://aws.amazon.com/ru/what-is/nlp/ (дата обращения: 20.10.2023).

[2] Хобсон Лейн, Ханнес Хапке, Коул Ховард Обработка естественного языка в действии. — СПб.: Питер, 2020. — С. 68-140.

[3] Ганегедара Т. Обработка естественного языка с TensorFlow / пер. с анг. В. С. Яценкова. – М.:ДМК Пресс, 2020. – С. 74-102.

[4] Hickman L. et al. Text preprocessing for text mining in organizational research: Review and recommendations //Organizational Research Methods. – 2022. – Т. 25. – №. 1. – С. 114-146.

[5] Kadhim A. I. An evaluation of preprocessing techniques for text classification //International Journal of Computer Science and Information Security (IJCSIS). – 2018. – Т. 16. – №. 6. – С. 22-32.

[6] Denny M. J., Spirling A. Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it //Political Analysis. – 2018. – Т. 26. – №. 2. – С. 168-189.

[7] Tabassum A., Patil R. R. A survey on text pre-processing & feature extraction techniques in natural language processing //International Research Journal of Engineering and Technology (IRJET). – 2020. – Т. 7. – №. 06. – С. 4864-4867.

[8] Etaiwi W., Naymat G. The impact of applying different preprocessing steps on review spam detection //Procedia computer science. – 2017. – Т. 113. – С. 273-279.

[9] Kashina M., Lenivtceva I. D., Kopanitsa G. D. Preprocessing of unstructured medical data: the impact of each preprocessing stage on classification //Procedia Computer Science. – 2020. – Т. 178. – С. 284-290.

[10] Pak M. Y., GUNAL S. The impact of text representation and preprocessing on author identification //Anadolu University Journal of Science and Technology A-Applied Sciences and Engineering. – 2017. – Т. 18. – №. 1. – С. 218-224.

[11] Идеальный препроцессинговый пайплайн для NLP-моделей // Temofeev.ru URL: https://temofeev.ru/info/articles/idealnyy-preprotsessingovyy-payplayn-dlya-nlp-modeley/ (дата обращения: 23.10.2023).

[12] A Gentle Introduction to the Bag-of-Words Model // Machine Learning Mastery URL: https://machinelearningmastery.com/gentle-introduction-bag-words-model/ (дата обращения: 28.10.2023).

[13] Gensim Word2Vec Tutorial // Kaggle URL: https://www.kaggle.com/code/pierremegret/gensim-word2vec-tutorial (дата обращения: 28.10.2023).

[14] Jeffrey Pennington, Richard Socher, Christopher D. Manning // GloVe: Global Vectors for Word Representation URL: https://www-nlp.stanford.edu/projects/glove/ (дата обращения: 02.11.2023).

[15] Графема // Википедия URL: https://ru.wikipedia.org/wiki/Графема (дата обращения: 03.11.2023).

[16] Satish Gunjal. Tokenization in NLP [Электронный ресурс]. – Режим доступа: https://www.kaggle.com/code/satishgunjal/tokenization-in-nlp (дата обращения: 04.11.2013).

[17] Machine Learning Mastery. How to Prepare Text Data for Deep Learning with Keras [Электронный ресурс]. – Режим доступа: https://machinelearningmastery.com/prepare-text-data-deep-learning-keras/ (дата обращения: 04.11.2023).

[18] Макмахан Брайан, Рао Делип. Знакомство с PyTorch. — СПб.: Питер, 2020. - С. 88-101

[19] Stemmer Porter. In: Wikipedia: the free encyclopedia [Электронный ресурс]. – Available at: https://ru.wikipedia.org/wiki/Стеммер_Портера (Дата обращения: 04.11.2023).

[20] Porter Stemmer. В: Snowball: Язык для алгоритмов стемминга [Электронный ресурс]. – Режим доступа: https://snowballstem.org/algorithms/porter/stemmer.html (дата обращения: 04.11.2023).

[21] Baeldung. Stemming vs Lemmatization [Электронный ресурс] // Baeldung.com. - Режим доступа: https://www.baeldung.com/cs/stemming-vs-lemmatization (дата обращения: 08.11.2023).

[22] Stopwords-iso repository on GitHub [Электронный ресурс] // GitHub.com. - Режим доступа: https://github.com/stopwords-iso (дата обращения: 08.11.2023).

[23] GitHub.com. Stopwords-iso repository on GitHub [Электронный ресурс]. Режим доступа: https://github.com/stopwords-iso (дата обращения: 08.11.2023).

[24] Stopwords-iso. Список стоп-слов для русского языка [Электронный ресурс]. Режим доступа: https://github.com/stopwords-iso/stopwords-ru/blob/master/stopwords-ru.txt (дата обращения: 08.11.2023).

[25] Kaggle. NLP Preprocessing [Электронный ресурс]. Режим доступа: https://www.kaggle.com/code/abdallahwagih/nlp-preprocessing (дата обращения: 08.11.2023).

[26] Макмахан Брайан, Рао Делип. Глубокое обучение при обработке естественного языка. — СПб.: Питер, 2020. - С. 46-92

[27] НКЖ. Открытый доступ к научным публикациям [Электронный ресурс]. Режим доступа: https://www.nkj.ru/open/36052/ (дата обращения: 08.11.2023).

[28] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural Architectures for Named Entity Recognition. arXiv preprint arXiv:1603.01360.

[29] NLP Эмбеддинги [Электронный ресурс]. Режим доступа: https://blog.bayrell.org/ru/iskusstvennyj-intellekt/495-nlp-embeddingi.html (дата обращения: 08.11.2023).

[30] Soyalp G. et al. Improving Text Classification with Transformer //2021 6th International Conference on Computer Science and Engineering (UBMK). – IEEE, 2021. – С. 707-712.

[31] Wang C., Banko M. Practical transformer-based multilingual text classification //Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers. – 2021. – С. 121-129.

[32] Shaheen Z., Wohlgenannt G., Filtz E. Large scale legal text classification using transformer models //arXiv preprint arXiv:2010.12871. – 2020.

[33] Tezgider M., Yildiz B., Aydin G. Text classification using improved bidirectional transformer //Concurrency and Computation: Practice and Experience. – 2022. – Т. 34. – №. 9. – С. e6486.

[34] Vaswani A. et al. Attention is all you need //Advances in neural information processing systems. – 2017. – Т. 30.

[35] Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding //arXiv preprint arXiv:1810.04805. – 2018.

[36] Sun C. et al. How to fine-tune bert for text classification? //Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18. – Springer International Publishing, 2019. – С. 194-206.

[37] Beltagy I., Peters M. E., Cohan A. Longformer: The long-document transformer //arXiv preprint arXiv:2004.05150. – 2020.

[38] Longformer model designed for Russian language [Электронный ресурс]. Режим доступа: https://huggingface.co/kazzand/ru-longformer-base-4096 (дата обращения: 08.11.2023).

[39] Hossin M., Sulaiman M. N. A review on evaluation metrics for data classification evaluations //International journal of data mining & knowledge management process. – 2015. – Т. 5. – №. 2. – С. 1.

[40] Li Y. et al. A comparative study of pretrained language models for long clinical text // J. Am. Med. Inform. Assoc. 2023. Vol. 30, № 2.

[41] Wei F. et al. An Empirical Comparison of DistilBERT, Longformer and Logistic Regression for Predictive Coding // Proceedings - 2022 IEEE International Conference on Big Data, Big Data 2022. 2022.

[42] Mamakas D. et al. Processing Long Legal Documents with Pre-trained Transformers: Modding LegalBERT and Longformer // NLLP 2022 - Natural Legal Language Processing Workshop 2022, Proceedings of the Workshop. 2022.

[43] Khandelwal A. Fine-Tune Longformer for Jointly Predicting Rumor Stance and Veracity // ACM International Conference Proceeding Series. 2020.

Статья получена 14 декабря 2023

А.К. Марков, АО «ВНИКТИнефтехимоборудование», инженер 2 категории, г. Волгоград, Россия (e-mail: akmarkov@vnikti.rosneft.ru).

Д.О. Семёночкин, Волгоградский государственный технический университет, Россия (e-mail: semm0202@yandex.ru).

А.Г. Кравец, Волгоградский государственный технический университет, Россия, д.т.н., профессор (e-mail: allagk@yandex.ru).

Т.А. Яновский, АО «ВНИКТИнефтехимоборудование», заведующий лабораторией моделирования и анализа данных, г. Волгоград, Россия (e-mail: TAYanovsy@vnikti.rosneft.ru).

Comparative analysis of applied natural language processing technologies for improving the quality of digital document classification

A.K. Markov, D.O. Semenochkin, A.G. Kravets, T.A. Yanovskiy

***Abstract*—Tagging digital electronic documents is the process of assigning metadata or labels (tags) to documents in order to simplify their organization, retrieval, and management. This process is essential for effective information management and document accessibility in a digital environment. The choice of tagging method and technology depends on the specific needs of the organization or user. Often a combination of different methods is used to achieve the best results in managing digital electronic documents. This paper performs a comparative analysis of natural language processing techniques to improve the quality of digital document classification, using technical educational documents as an example. The paper discusses the methods used in document preprocessing and the use of NLP, ways to improve preprocessing, and a computational experiment is conducted to determine the improvement in the completeness and accuracy of data classification.**

***Keywords*—Tagging, NLP, Classification, Metadata, Processing, Categorization, Tag.**

REFERENCES

[1] URL: https://aws.amazon.com/ru/what-is/nlp/ (accessed 20.10.2023).What is Natural Language Processing (NLP) // Amazon.

[2] Hobson Lane, Hannes Hapke, Cole Howard Natural Language Processing in Action. - SPb.: Peter, 2020. - C. 68-140.

[3] Ganegedara T. Natural Language Processing with TensorFlow. V. S. Yatsenkov. - Moscow: DMK Press, 2020. - C. 74-102.

[4] Hickman L. et al. Text preprocessing for text mining in organizational research: Review and recommendations //Organizational Research Methods. – 2022. – Т. 25. – №. 1. – С. 114-146.

[5] Kadhim A. I. An evaluation of preprocessing techniques for text classification //International Journal of Computer Science and Information Security (IJCSIS). – 2018. – Т. 16. – №. 6. – С. 22-32.

[6] Denny M. J., Spirling A. Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it //Political Analysis. – 2018. – Т. 26. – №. 2. – С. 168-189.

[7] Tabassum A., Patil R. R. A survey on text pre-processing & feature extraction techniques in natural language processing //International Research Journal of Engineering and Technology (IRJET). – 2020. – Т. 7. – №. 06. – С. 4864-4867.

[8] Etaiwi W., Naymat G. The impact of applying different preprocessing steps on review spam detection //Procedia computer science. – 2017. – Т. 113. – С. 273-279.

[9] Kashina M., Lenivtceva I. D., Kopanitsa G. D. Preprocessing of unstructured medical data: the impact of each preprocessing stage on classification //Procedia Computer Science. – 2020. – Т. 178. – С. 284-290.

[10] Pak M. Y., GUNAL S. The impact of text representation and preprocessing on author identification //Anadolu University Journal of Science and Technology A-Applied Sciences and Engineering. – 2017. – Т. 18. – №. 1. – С. 218-224.

[11] Ideal preprocessing pipelines for NLP models // Temofeev.ru URL: https://temofeev.ru/info/articles/idealnyy-preprotsessingovyy-payplayn-dlya-nlp-modeley/ (date of access: 23.10.2023).

[12] A Gentle Introduction to the Bag-of-Words Model // Machine Learning Mastery URL: https://machinelearningmastery.com/gentle-introduction-bag-words-model/ (date of access: 28.10.2023).

[13] Gensim Word2Vec Tutorial // Kaggle URL: https://www.kaggle.com/code/pierremegret/gensim-word2vec-tutorial (date of access: 28.10.2023).

[14] Jeffrey Pennington, Richard Socher, Christopher D. Manning // GloVe: Global Vectors for Word Representation URL: https://www-nlp.stanford.edu/projects/glove/ (date of access: 02.11.2023).

[15] Grapheme // Wikipedia URL: https://ru.wikipedia.org/wiki/Графема (date of access 03.11.2023).

[16] Satish Gunjal. Tokenization in NLP [Electronic resource]. – Access mode: https://www.kaggle.com/code/satishgunjal/tokenization-in-nlp (date of access: 04.11.2013).

[17] Machine Learning Mastery. How to Prepare Text Data for Deep Learning with Keras [Electronic resource]. – Access mode: https://machinelearningmastery.com/prepare-text-data-deep-learning-keras/ (date of access: 04.11.2023).

[18] McMahan Brian, Rao Delip. Getting to know PyTorch. - SPb.: Peter, 2020. - С. 88-101

[19] Stemmer Porter. In: Wikipedia: the free encyclopedia [Electronic resource]. - Available at: https://ru.wikipedia.org/wiki/Стеммер_Портера (date of access: 04.11.2023).

[20] Porter Stemmer. In: Snowball: A language for stemming algorithms [Electronic resource]. - Access mode: https://snowballstem.org/algorithms/porter/stemmer.html (date of access: 04.11.2023).

[21] Baeldung. Stemming vs Lemmatization [Electronic resource] // Baeldung.com. - Access mode: https://www.baeldung.com/cs/stemming-vs-lemmatization (date of access: 08.11.2023).

[22] Stopwords-iso repository on GitHub [Electronic resource] // GitHub.com. - Access mode: https://github.com/stopwords-iso (date of access: 08.11.2023).

[23] GitHub.com. Stopwords-iso repository on GitHub [Electronic resource]. Access mode: https://github.com/stopwords-iso (date of access: 08.11.2023).

[24] Stopwords-iso. List of stopwords for Russian language [Electronic resource]. Access mode: https://github.com/stopwords-iso/stopwords-ru/blob/master/stopwords-ru.txt (date of access: 08.11.2023).

[25] Kaggle. NLP Preprocessing [Electronic resource]. Access mode: https://www.kaggle.com/code/abdallahwagih/nlp-preprocessing (date of access: 08.11.2023).

[26] McMahan Brian, Rao Delip. Deep learning in natural language processing. - SPb.: Peter, 2020. - С. 46-92

[27] NCW. Open access to scientific publications [Electronic resource]. Access mode: https://www.nkj.ru/open/36052/ (date of access: 08.11.2023).

[28] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural Architectures for Named Entity Recognition. arXiv preprint arXiv:1603.01360.

[29] NLP Эмбеддинги [Electronic resource]. Access mode: https://blog.bayrell.org/ru/iskusstvennyj-intellekt/495-nlp-embeddingi.html (date of access: 08.11.2023).

[30] Soyalp G. et al. Improving Text Classification with Transformer //2021 6th International Conference on Computer Science and Engineering (UBMK). – IEEE, 2021. – С. 707-712.

[31] Wang C., Banko M. Practical transformer-based multilingual text classification //Proceedings of the 2021 Conference of the North

American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers. – 2021. – С. 121-129.

[32] Shaheen Z., Wohlgenannt G., Filtz E. Large scale legal text classification using transformer models //arXiv preprint arXiv:2010.12871. – 2020.

[33] Tezgider M., Yildiz B., Aydin G. Text classification using improved bidirectional transformer //Concurrency and Computation: Practice and Experience. – 2022. – Т. 34. – №. 9. – С. e6486.

[34] Vaswani A. et al. Attention is all you need //Advances in neural information processing systems. – 2017. – Т. 30.

[35] Devlin J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding //arXiv preprint arXiv:1810.04805. – 2018.

[36] Sun C. et al. How to fine-tune bert for text classification? //Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18. – Springer International Publishing, 2019. – С. 194-206.

[37] Beltagy I., Peters M. E., Cohan A. Longformer: The long-document transformer //arXiv preprint arXiv:2004.05150. – 2020.

[38] Longformer model designed for Russian language [Electronic resource]. Access mode: https://huggingface.co/kazzand/ru-longformer-base-4096 (date of access: 08.11.2023).

[39] Hossin M., Sulaiman M. N. A review on evaluation metrics for data classification evaluations //International journal of data mining & knowledge management process. – 2015. – Т. 5. – №. 2. – С. 1.

[40] Li Y. et al. A comparative study of pretrained language models for long clinical text // J. Am. Med. Inform. Assoc. 2023. Vol. 30, № 2.

[41] Wei F. et al. An Empirical Comparison of DistilBERT, Longformer and Logistic Regression for Predictive Coding // Proceedings - 2022 IEEE International Conference on Big Data, Big Data 2022. 2022.

[42] Mamakas D. et al. Processing Long Legal Documents with Pre-trained Transformers: Modding LegalBERT and Longformer // NLLP 2022 - Natural Legal Language Processing Workshop 2022, Proceedings of the Workshop. 2022.

[43] Khandelwal A. Fine-Tune Longformer for Jointly Predicting Rumor Stance and Veracity // ACM International Conference Proceeding Series. 2020.