МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра дискретной математики и информационных технологий

ПРИМЕНЕНИЕ ВЕКТОРНЫХ МОДЕЛЕЙ ПРЕДСТАВЛЕНИЯ ТЕКСТОВ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Бикбулатова Рамиля Гиниятовича

Научный руководитель
доцент, к. ф.-м. н. _____ В. А. Поздняков

Заведующий кафедрой
к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2019

ВВЕДЕНИЕ

В настоящее время, в связи с большим объёмом документации и текстовых данных в электронном виде, встала острая необходимость в их упорядочивании и систематизации. Одними из самых перспективных направлений в решении проблем данной области являются технологии, которые основаны на машинном обучении. Главными задачами подобных технологий являются, в первую очередь, задачи поиска и извлечения необходимых информационных данных, фактов, ключевых словосочетаний и классификация текстов.

В задачах классификации и анализа текста существует необходимость представления текста с помощью специальных векторных моделей. В случаях, когда два документа близки по смыслу - их можно представить близкими векторами. Подобный способ структурирования текстовой информации используется во многих задачах анализа, например, в задачах информационного поиска, категоризации, классификации, анализа тональности, а так же генерации ответа в диалоговых системах.

В простом случае, векторная модель подразумевает сопоставление каждому тексту частотного спектра слов и вектора в лексическом пространстве. Задачи информационного поиска подразумевают вычисление оценок релевантности «строка-текст». В качестве текстов выступают те или иные документы, а в качестве строк - ключевые слова или словосочетания, заданные экспертом, извлечённые из документов по заданным принципам, либо элементы текста, состоящие из фиксированного набора символов.

Имея подобные данные для всех рассматриваемых документов, можно определять их схожесть по мерам релевантности, которые должны быть интуитивно простыми, независимыми от длины текста, а также предоставлять возможность эффективной вычислительной реализации. В случае поиска, запрос будет представлен в виде вектора, что поможет определить документы, соответсвующие данному запросу.

Существуют различные подходы к построению векторной модели. Актуальной задачей является выбор векторной модели для конкретного корпуса текстовых документов.

Целью настоящей работы является совершенствование процесса автоматизации обработки текстовых документов.

В процессе выполнения поставленной цели, необходимо решить следующие задачи:

— анализ и сравнение моделей представления текстов;
— применение технологий скрапинга для получения корпуса текстов;
— формирование выборки для решения задачи классификации;
— разработка интерфейса приложения;
— сравнение и выбор векторной модели, в задаче классификации текстов.

В первом разделе выпускной квалификационной работы рассмотрены несколько подходов представления текста. Рассматриваются основные этапы предварительной обработки текстов, а также методы машинного обучения. Во втором разделе представлен каждый этап разработки приложения для классификации сайтов ИКТ-компаний: технология скрапинга данных, используемые библиотеки для предобработки и представления данных, а также разработка интерфейса приложения. Третий раздел содержит сравнение полученных векторных моделей представления текста, включающее формирование обучающей выборки и оценку точности каждой из моделей с использованием различных метрик.

Методологические основы бакалаврской работы по теме «Применение векторных моделей представления текстов» представлены в работах Митчела P., Sarkar D., Russel S., официальной документации используемых библиотек для языка Python.

Практическая значимость заключается в разработке приложения, позволяющего осуществить выбор векторной модели для решения задачи классификации сайтов по результатам анализа их содержимого.

Структура и объём работы. Бакалаврская работа состоит из введения, трёх разделов, заключения, списка использованных источников и одного приложения. Общий объем работы – 45 страниц, из них 36 страниц – основное содержание, включая 11 рисунков и 3 таблицы, цифровой носитель в качестве приложения, список использованных источников информации – 22 наименования.

Первый раздел «Анализ методов представления текстов» посвящен классификации методов предсталвения текстов, основным этапам предварительной обработки текстов и методам машинного обучения для обработки текстовых данных.

Существует несколько основных методов представления текста: векторная модель, модели суффиксных деревьев, модели скрытых тем, теоретико-множественная модель, а так же языковая модель. Все эти модели основаны на одном принципе: текст состоит из термов, которые представляют собой нормализованные слова или их значимые части, например, основы. У каждой из этих моделей есть свои преимущества и недостатки. Например, в силу своих особенностей, векторная модель не учитывает порядок слов, что приводит к невозможности использования данной модели в задачах генерации текста или оценки его появления. Зачастую, в подобных случаях используют языковые модели, а так же некоторые из моделей скрытых тем.

Векторная модель — самая популярная модель представления текстов. Текст, в этой модели, представляется вектором в многомерном пространстве слов, где каждому слову соответствует своя координата векторного пространства. Каждый из векторов представляет собой значение частоты нахождения слова в тексте.

Главными преимуществами векторной модели представления текстов является её простота и возможность использования линейно-алгебраических операций для определения сходства между текстами и их сортировки по конкретному запросу.

Большинство описательных текстовых данных создаются людьми, что непременно ведёт к, так называемому, «шуму» в данных. Поэтому, прежде чем приступить к анализу данных, следует совершить ряд операций по устранению этого «шума». Для этой цели будут использоваться: токенизация, отбрасывание стоп-слов и нормализация слов. Помимо этих операций, используют так же стемминг и семантическое ядро. Стемминг - это процесс нахождения основы для заданного слова.

Следующим этапом является отбрасывание стоп-слов. Стоп-слова представляют собой слова в тексте, которые не несут особой смысловой нагрузки при индивидуальном рассмотрении каждого. Однако, могут сильно повлиять на время анализа текста и на его точность. Поэтому, задача отбрасывания

стоп-слов включает задачу поиска ключевого слова.

Проблема понимания текстов алгоритмами, включает в себя проблему близости слов, а так же их склонения. Для машинного алгоритма слова «разработки» и «разработку» будут являться двумя отдельными словами, что приведёт к значительным последствиям при анализе корпуса текстов. Данная проблема свойственна всем языкам, русский язык не исключение, более того, из-за особенности склонений и окончаний, функция канонизации слов усложняется, а библиотек, которые могут корректно приводить слова в нормальную форму на русском языке не так много.

Рассмотрим основные методы обучения, которые применяются для решения задач классификации.

Одним из самых популярных алгоритмов для классификации данных является метод опорных векторов, известный в англоязычной литературе под названием Support Vector Machine (SVM).

Метод является дискриминантным классификатором, формально определённым разделяющей гиперплоскостью. Другими словами, классификатор выдаёт оптимальную гиперплоскость, построенную на обучающей выборке, которая классифицирует новые элементы поступающих данных. В двухмерном пространстве эта гиперплоскость представляет собой линию, разделяющую плоскость на две части, где каждая часть представляет собой класс.

Рассмотрим ещё один метод классификации данных.

Задача любого классификатора текста подразумевает под собой нахождение класса, в который будет занесён рассматриваемый документ. Соответственно, задача байесовской классификации состоит в нахождении самого вероятного класса c_m, который будет рассчитан по следующей формуле:

$$c_m = argmax_{c \in C} P(c|d) \tag{1}$$

где d – текст, c – класс, $argmax$ – это элемент, при котором находится максимальное значение.

Перейдём к рассмотрению другого алгоритма.

Суть алгоритма случайный лес заключается в построении ассамблеи деревьев, которые сами по себе дают не лучшие результаты классификации, однако в совокупности результат является достаточно высоким. Каждое отдельное дерево этой ассамблеи является методом, позволяющим предсказы-

вать принадлежность наблюдений или объектов к тому или иному классу категориальной зависимой переменной, в зависимости от соответствующих значений одной или нескольких предикторных переменных.

Исходя из вышеописанного, для классификации предлагается использовать метод наивного байесовского классификатора и алгоритм случайный лес в виду простоты их реализации и низких вычислительных затрат при обучении и классификации.

Второй раздел «Разработка приложения для классификации web-ресурсов» посвящен практической реализации выбранных методов классификации на языке Python, а также применению технологии скрапинга, инструментальных средств обработки и разработки интерфейса приложения.

Web-скрапинг - это обобщённый термин для различных методов, которые используются для сбора информации по всей сети Интернет. Сбор осуществляется при помощи различных программных решений, которые имитируют человеческие действия по сбору конкретных данных с различных сайтов. Люди, использующие программы для web-скрапинга, обычно, заинтересованы в получении определённой информации, для формирования баз данных, которые могут использоваться в различных целях. Например, реклама.

Получать необходимые данные, предполагается с помощью библиотеки с функцией скрапинга сайтов BeautifulSoup. BeautifulSoup представляет собой библиотеку для синтаксического разбора файлов html и xml, которая написана на языке Python. Работа библиотеки заключается в преобразовании разметки в дерево синтаксического разбора.

Рассмотрим используемые инструментальные средства на различных этапах предобработки и представления текста.

В решении задачи очистки текста от шума, предлагается использовать список стоп-слов, импортированный из библиотеки NLTK. Несмотря на то, что библиотека располагает обширной базой стоп-слов для нескольких языков, в ней отсутствует функция лемматизации (приведения в нормальную форму) слов русского языка. Поэтому на данном этапе нормализации слов, предполагается использование другой библиотеки, позволяющей воспользоваться приведением слов в нормальную форму. Pymystem3 может быть как внутрипрограммной библиотекой, используемой для задач токенизации, нор-

мализации и гипотетического разбора слов, так и отдельной программой для морфологического анализа текста на русском языке. Главной функцией рассматриваемой библиотеки является приведение слов русского языка в нормальную форму для улучшения точности работы алгоритмов классификации на следующих этапах работы.

Одной из главных проблем работы алгоритмов — невозможность работы с текстом, представленным в привычном для нас всех виде. Поэтому, полученный нами корпус документов должен быть преобразован в матрицу числовых значений для каждого из термов, с которой алгоритму будет работать привычнее. Подобные матрицы называются методами представления векторной модели и в нашей работе будут использоваться 3 разных метода, результаты работы которых можно будет оценить.

Представлены эти модели будут тремя классами библиотеки Sklearn: TfIdfVectorizer, CountVectorizer и HashingVectorizer. Данные классы предполагают процедуры предобработки и построения матриц, заложенные внутри этих векторизаторов.

Класс TfIdfVectorizer позволит нам представить корпус документов в виде математической матрицы $tf\text{-}idf$, в которой описаны частоты каждого из термов в конкретном документе. Векторизатор имеет функции токенизации, составления словаря и получения обратной частоты каждого терма. Возвращённая векторизатором терм-документная матрица будет использована на следующем этапе для обучения используемых классификаторов.

В отличие от предыдущего векторизатора, CountVectorizer производит лишь подсчёт частоты каждого слова, без пропорционального компенсирования логарифмической частью обратной частоты.

Целочисленные значения частоты и весов термов являются полезными и необходимыми метриками в задачах классификации текста. Но есть один существенный недостаток подобных методов подсчёта - размерность словаря может достигать очень больших объёмов. В таких случаях потребуются большие значения векторных выражений для кодировки документов, что влечёт увеличенные требования к оперативной памяти компьютера, а также замедлению работы алгоритмов.

Оптимальным решением подобной проблемы является использование односторонней функции хэширования для преобразования слов в целочислен-

ные значения. Представлена эта функция будет векторизатором HashingVectorizer, благодаря которому отпадает необходимость в словаре и пользователь может выбирать вектор произвольной длины. Единственным недостатком этого решения является невозможность обратного преобразования векторного значения терма в изначальный вид.

Используемая нами библиотека Sklearn также имеет функцию наивного байесовского классификатора GaussianNB, которая позволяет определить, к какому из классов будет относиться конкретный текст. Другим классификатором на основе выбранной векторной модели будет классификатор RandomForestClassifier. Для оценки результатов точности каждой из моделей с помощью проверки кросс-валидации, воспользуемся уже реализованным инструментом библиотеки Sklearn.

Для удобного представления элементов взаимодействия с программным кодом и результатами его выполнения, было решено разработать интерфейс. В разработке интерфейса приложения была использована библиотека PySide2, работающая на языке Python версии 3.5 и выше.

Третий раздел «Сравнение векторных моделей представления текстовых документов» включает формирование обучающей выборки для классификации и оценку векторных моделей.

Обучающая выборка - это первоначальный набор данных, который используется для обучения алгоритма классификации в работе с подобными элементами выборки и на их основании получать необходимые результаты. Выборка в данной работе будет представлять собой список сайтов различных ИКТ-компаний, где в роли классов выборки будут выступать род деятельности компаний. Данная обучающая выборка подбирается экспертным путём, с применением анализа каждой из предметных областей. Для наглядности представим гистограмму

Для оценки точности обоих классификаторов с использованием нескольких методов представления текста, предполагается использовать перекрёстную проверку.

Данная проверка позволяет представить классы обучающего множества примерно в одинаковой пропорции, что помогает избежать доминации одного класса над другим, как в случаях неудачного формирования множества. Неравномерное распределение элементов в классах может привести к «перевесу» в процессе обучения, и доминирующий класс будет рассматриваться как самый вероятный. Поэтому перекрёстная проверка является решением проблем подобного рода.

В ходе перекрёстной проверки обучающего корпуса текстов, полученных с сайтов ИКТ-компаний, были сформированы результаты для каждого из используемых классификаторов и метрик. Были выбраны самый точный метод представления для данной обучающей выборки, а именно модель tf-idf, и классификатор случайный лес.

Использовались различные алгоритмы классификации, и были проанализированы три выбранных векторных модели представления текста. Результаты с оценками точности не позволяют выбрать единственную лучшую векторную модель для любого метода, так как точность модели зависит и от классификатора.

В качестве методики выбора векторной модели предлагается встроить в приложение по автоматической классификации текстов дополнительный модуль. Данный модуль будет использовать различные векторные модели, в частности модель: tf-idf, модель частотной матрицы и модель хэш-функции, и выбирать тот метод представления, который обеспечит большую точность классификации.

ЗАКЛЮЧЕНИЕ

Методы текстового анализа широко применяются для решения практических задач обработки неструктурированного текстового контента. Text Mining предполагает многоэтапный процесс, включающий предварительную обработку текста, использование векторных моделей представления, построение модели на основе методов машинного обучения, оценку точности модели и её применение.

В выпускной квалификационной работе представлена задача анализа и сравнения моделей представления текстов. Анализ моделей представления текста позволил выбрать ряд наиболее часто применяемых подходов в решении подобного рода задач, а именно модель $tf\text{-}idf$, модель частотной матрицы и модель хэш-функции. Были реализованы основные этапы предварительной обработки текстов, а именно: токенизация, нормализация и фильтрация стоп-слов. На этапе построения модели были рассмотрены основные методы обучения, которые применяются для решения задач классификации (модель опорных векторов, модель наивного Байеса, случайный лес).

Для получения корпуса текстовых документов использовались технологии скрапинга, обеспечивающие сбор текстовых данных с web-ресурсов. Применение технологий скрапинга для получения обучающего выборки, состоящей из набора классифицированных экспертным способом текстов, осуществлено с помощью одной из самых популярных библиотек BeautifulSoup, используемой для задач подобного рода.

В качестве практической задачи классификации рассмотрена задача автоматического определения категории компании, занятой в сфере информационных технологий с учетом основного направления деятельности. В ходе формирования выборки для решения задач классификации были отобраны и классифицированы сайты российских ИКТ-компаний.

При разработке интерфейса приложения был использован графический редактор пользовательских интерфейсов QT Designer, позволяющий использовать разработанный интерфейс в программах языка Python. Виджеты и формы, созданные в среде разработки QT Designer были встроены с использованием управляющего кода, позволяющего определять поведение графических элементов благодаря связке слотов и сигналов. Все свойства управляющих элементов и форм ввода и вывода информации были настроены дина-

мически внутри кода.

Были также произведены сравнение и анализ получившихся моделей представления текста и сформулирован вывод на основе оценок точности, который показал, что не существует единственной лучшей векторной модели для рассматриваемых методов классификации и выбор наиболее точной из них является оптимальным решением.

ОСНОВНЫЕ ИСТОЧНИКИ ИНФОРМАЦИИ

1. Скрапинг веб-сайтов с помощью Python / Райан Митчелл; пер.: А. В. Груздев. — М.: ДМК Пресс, 2016. 280с.
2. Sarkar D., Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from Your Data. — Apress, 2016. 377c.
3. NLTK 3.4.1 documentation [Электронный ресурс]. URL: https://www.nltk.org (дата обращения 15.05.2019). Загл. с экрана. Яз. англ.
4. Russel S., Norvig P., Artificial Intelligence: A Modern Approach, Pearson there's a complete analytical explanation. — 2016. 588c.
5. Beautiful Soup Documentation [Электронный ресурс]. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (дата обращения 15.05.2019). Загл. с экрана. Яз. англ.
6. sklearn.featureextraction.text.TfidfVectorizer — scikit-learn 0.21.2 documentation [Электронный ресурс]. URL:https://scikit-learn.org/stable/modules/generated/sklearn.featureextraction.text.TfidfVectorizer.html (дата обращения 15.05.2019). Загл. с экрана. Яз. англ.
7. sklearn.featureextraction.text.CountVectorizer — scikit-learn 0.21.2 documentation [Электронный ресурс]. URL: https://scikit-learn.org/stable/modules/generated/sklearn.featureextraction.text.CountVectorizer.html (дата обращения 15.05.2019). Загл. с экрана. Яз. англ.
8. sklearn.featureextraction.text.HashingVectorizer — scikit-learn 0.21.2 documentatio [Электронный ресурс]. URL:https://scikit-learn.org/stable/modules/generated/sklearn.featureextraction.text.HashingVectorizer.html (дата обращения 15.05.2019 Загл. с экрана. Яз. англ.