ПОЛИТЕХ
Санкт-Петербургский
политехнический университет
Петра Великого

На правах рукописи

Ковалев Артем Дмитриевич

МЕТОДЫ АВТОМАТИЗАЦИИ СОПРОВОЖДЕНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА ОСНОВЕ ИНТЕЛЛЕКТУАЛЬНОЙ ОБРАБОТКИ ЗАПРОСОВ ЗАКАЗЧИКА

2.3.5. Математическое и программное обеспечение
вычислительных систем, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата технических наук

Санкт-Петербург
2023

Работа выполнена в федеральном государственном автономном образовательном учреждении высшего образования «Санкт-Петербургский политехнический университет Петра Великого».

Научный руководитель:

ДРОБИНЦЕВ Павел Дмитриевич, кандидат технических наук, доцент, директор высшей школы программной инженерии.

Официальные оппоненты:

ЗМЕЕВ Олег Алексеевич, доктор физико-математических наук, профессор, академический директор центра «Высшая IT-школа», ФГАОУ ВО «Национальный исследовательский Томский государственный университет».

КОЗНОВ Дмитрий Владимирович, доктор технических наук, профессор, профессор кафедры системного программирования, ФГБОУ ВО «Санкт-Петербургский государственный университет».

АБРАМСКИЙ Михаил Михайлович, кандидат технических наук, доцент, директор института информационных технологий и интеллектуальных систем, ФГАОУ ВО «Казанский (Приволжский) федеральный университет».

Защита состоится «16» марта 2023 г. в 16^{00} часов на заседании диссертационного совета У.2.3.5.38 при ФГАОУ ВО «Санкт-Петербургский политехнический университет Петра Великого» в удаленном интерактивном режиме на платформе MS Teams: http://bit.ly/3XJksXa.

С диссертацией можно ознакомиться в Фундаментальной библиотеке ФГАОУ ВО «Санкт-Петербургский политехнический университет Петра Великого» и на сайте https://www.spbstu.ru/science/the-department-of-doctoral-studies/defences-calendar/the-degree-of-candidate-of-sciences/kovalev_artem_dmitrievich/.

Автореферат разослан «___» _____ 2023 г.

Ученый секретарь
диссертационного совета
У.2.3.5.38, к.т.н., доцент Дробинцев Павел Дмитриевич

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования и степень ее разработанности. Сопровождение программного обеспечения (ПО) является одним из наиболее значимых и трудоемких этапов жизненного цикла разработки. По некоторым оценкам оно занимает до 67% всех затрат при разработке ПО. На данном этапе инженеры компании поставщика принимают и обрабатывают запросы от заказчиков программного продукта. Запрос, как правило, написан на естественном неформализованном языке и содержит описание ошибки или проблемы, возникшей в ходе эксплуатации программного продукта на стороне заказчика. Так как количество дефектов и недоработок многократно возрастает по мере развития и усложнения программных продуктов, то актуальной задачей исследований является снижение трудоемкости обработки таких запросов.

На сегодняшний день область автоматизации этапа сопровождения ПО активно исследуется и развивается в таких направлениях, как 1) поиск семантически похожих запросов об ошибках, 2) поиск и назначение подходящего инженера с необходимым уровнем экспертизы для решения запроса, 3) классификация/категоризация входящих запросов об ошибках и присвоение им определенных тематических меток, 4) извлечение краткого описания и ключевых слов из текстовых данных запросов, 5) определение участков кода, которые потенциально могут являться причиной возникшей ошибки в работе программы. В развитие данных направлений заметный вклад внесли следующие авторы: О.В. Кузина, А.Ю. Крайнов, А.А. Смагин, Н.А. Двойнишников, К.А. Щипанов, S. Rastkar, G.C. Murphy, G. Murray, X. Li, H. Jiang, D. Liu, Z. Ren, G. Li, M.I. Nawaz Tarar, F. Ahmed, W.H. Butt, L. Bo, J. Lu, H. Wang, X. Xia, D. Lo, J. Grundy, X. Wang, L. Zhang, T. Xie, J. Anvik, J. Sun, C. Sun, J. Jiang, S.-C. Khoo, H. Mahfoodh, M. Hammad, A. Al-Batlaa, M. Abdullah-Al-Wadud, M. Anwar, M. D’Ambros, M. Lanza, R. Robbes, R. Almhana, W. Mkaouer, M. Kessentini, A. Ouni и другие. Результаты диссертационной работы концентрируются на первых двух подходах к автоматизации этапа сопровождения ПО.

В своих работах F. Ebert, E. Aghajani, T. Dasgupta, M. Grechanik, E. Moritz, B. Dit, D. Poshyvanyk, D. Sondhi, A. Gupta, S. Purandare, A. Rana, D. Kaushal, R. Purandare предлагают методы, направленные на автоматическое обновление и совершенствование программной документации в процессе разработки ПО. Однако область эффективной обработки запросов заказчика имеет свои особенности и развитие этих методов не получило здесь достойного продолжения. Для успешного применения в сфере автоматизации этапа сопровождения упомянутые методы требуют существенных доработок. Поэтому автоматизация семантического поиска по локальной и удаленной документации программного продукта представляется весьма перспективной и помогает значительно снизить трудоемкость решения запросов.

Еще одним направлением повышения эффективности обработки запросов является автоматизация рутинных действий за счет использования механизмов исполнения правил. Здесь следует отметить работы авторов S. Juan-Song, H. Bin-Wang, L. Chen, X. Dong, Z. Yang, K. Qu, T. Gong, J. Shao, результаты которых показали свою продуктивность в различных задачах автоматизации процессов. Развитие этих методов в области снижения трудоемкости этапа сопровождения ПО также является многообещающим и помогает избежать

повторяющихся операций, сокращая время на исправление ошибок. Учитывая характерные особенности обработки запросов заказчиков, необходимо модифицировать существующие механизмы исполнения правил для эффективной автоматизации этапа сопровождения ПО.

Ни одно из существующих исследований в выбранной предметной области не рассматривает применение совокупности методов для снижения трудоемкости этапа сопровождения. Развитие этих методов в рамках единого подхода является предметом данной диссертации.

Важным элементом ряда этих методов является использование модифицированного алгоритма построения векторных пространств (Doc2VecC), который был выбран на основе всестороннего сравнительного анализа передовых техник и подходов к информационному поиску по неструктурированным текстовым данным. Он используется для создания векторных представлений текстовых документов и помогает получить новые результаты в работе трех выбранных для изучения подходов.

Цель и задачи диссертационной работы. Целью диссертационной работы является разработка методов автоматизированной обработки запросов заказчиков в системах отслеживания ошибок с применением подходов интеллектуального анализа данных, позволяющих сократить трудоемкость этапа сопровождения программного обеспечения.

Для достижения поставленной цели в работе ставились и решались следующие задачи.

1. Формализовать существующий подход к обработке запросов заказчиков на этапе сопровождения ПО с выявлением узких мест, требующих автоматизации, а также определением недостатков существующих систем поддержки и реализованных в них методов интеллектуальной помощи на основе всестороннего сравнительного анализа программных инструментов.
2. Разработать алгоритм построения векторных пространств, решающий проблему эффективного представления текстовых данных из области программной инженерии в виде числовых векторов.
3. Предложить набор методов автоматизации сопровождения программного обеспечения на основе интеллектуальной обработки запросов заказчика.
4. Определить оптимальные гиперпараметры алгоритма построения векторных пространств для эффективной работы предлагаемых методов и сформулировать правила обработки запросов.
5. Разработать программный комплекс на языке Java, реализующий предложенные в работе методы.
6. Оценить эффективность предложенных методов по снижению трудоемкости этапа сопровождения в сравнении с существующим подходом на трех экспериментальных проектах: Apache Kafka, Apache ActiveMQ, InfoProject.
7. Оценить качество векторных моделей и точность применения правил обработки запросов.
8. Внедрить предлагаемые методы в этап сопровождения проекта InfoProject.

Постановка цели и задач исследования соответствует следующим пунктам паспорта специальности 2.3.5 «Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей»: 3. Модели, методы, алгоритмы, языки и программные

инструменты для организации взаимодействия программ и программных систем; 10. Оценка качества, стандартизация и сопровождение программных систем.

Предметом исследования являются методы и инструментальные средства автоматизации обработки запросов, написанных на естественном языке, в процессе этапа сопровождения программных продуктов.

Научные результаты и их новизна

1. Разработан алгоритм построения векторных пространств, являющийся развитием алгоритма Doc2VecC для случая работы с техническими текстами из области программной инженерии. Новизна заключается в использовании регулярных выражений вместо стохастического подхода для выборки слов при формировании глобального контекста слова во время обучения нейронной сети алгоритма.

2. Предложен подход к автоматизации и снижению трудоемкости этапа сопровождения ПО, который содержит следующий комплекс методов:

2.1. Поиск семантически похожих решенных запросов заказчика в системе отслеживания ошибок за счет подготовки и векторизации текстовых данных запросов с помощью предложенного алгоритма построения векторных пространств.

2.2. Нахождение семантически похожих на входящий запрос заказчика сегментов документации ПО. Метод включает в себя подготовку и векторизацию текстовых данных входящего запроса и документации с помощью нового алгоритма построения векторных пространств.

2.3. Автоматизация рутинных действий за счет применения разработанных правил анализа и обработки запроса, состоящих из условия и действия. Результатом работы метода является набор рекомендаций, который может быть применен к запросу в автоматизированном режиме.

2.4. Автоматизированный выбор сотрудников, имеющих соответствующую экспертизу в решении поступившего запроса заказчика. Данный метод основан на методе 1, его отличительной особенностью является извлечение имен инженеров из семантически похожих решенных запросов и их ранжирование на основе вероятностного принципа.

Теоретическая значимость работы

Результаты исследования связаны с областью информационного поиска и автоматизацией этапа сопровождения программного обеспечения. Теоретическую значимость работы составляют: предложенный алгоритм построения векторных пространств; разработанные методы снижения трудоемкости обработки запросов заказчиков; созданные интеллектуальные правила обработки запросов, которые могут быть применены в различных программных проектах.

Практическая значимость работы

Разработанный алгоритм построения векторных пространств реализован в виде программной библиотеки и может быть адаптирован к любой иной предметной области, где необходима обработка специализированных текстовых данных. Выигрыш в точности по сравнению с существующим алгоритмом Doc2VecC составляет 15,8% на примере текстов из области программной инженерии.

На базе полученных научных результатов разработан программный комплекс, реализующий предложенные в работе методы автоматизации этапа сопровождения ПО. Он

успешно интегрирован в процесс обработки запросов заказчиков на одном из крупных промышленных проектов (InfoProject), где заметно снизил нагрузку на сотрудников и затраты на сопровождение. Кроме того, данный программный комплекс дополнительно был апробирован еще на двух проектах с открытым исходных кодом: Apache Kafka, Apache ActiveMQ. Автоматизированный подход показал высокую эффективность при обработке входящих запросов заказчиков, позволив сократить трудоемкость этапа сопровождения в 2,4 раза по сравнению с принятым в настоящее время подходом. Применение результатов исследования показывает фактическую пользу в практической деятельности.

Отдельные разработки диссертации внедрены в учебный процесс первого курса магистратуры Высшей школы программной инженерии в Санкт-Петербургском политехническом университете Петра Великого по дисциплине «Системы анализа больших данных».

Методология и методы исследования. Для решения поставленных задач в работе использовались методы математического моделирования, теории множеств и вычислительной математики. Также в диссертации применяются методы планирования, измерения и обработки результатов эксперимента. Кроме того, исследование включает в себя методы оптимизации и оценки качества моделей машинного обучения, а также методы оценки качества ранжирования результатов поиска.

Положения, выносимые на защиту

1. Предложенный алгоритм построения векторных пространств, обеспечивающий эффективный и точный поиск семантически похожих запросов заказчиков и релевантных сегментов документации на этапе сопровождения ПО.
2. Методы автоматизации сопровождения ПО, позволяющие сократить трудоемкость обработки запросов заказчиков:
   2.1. Метод поиска семантически похожих решенных запросов для нерешенных запросов заказчика в системе отслеживания ошибок.
   2.2. Метод нахождения семантически похожих на входящий запрос заказчика сегментов документации ПО.
   2.3. Метод автоматизации рутинных действий, который заключается в применении разработанных правил анализа и обработки запроса.
   2.4. Метод автоматизированного выбора сотрудников, имеющих соответствующую экспертизу в решении поступившего запроса заказчика.

Степень достоверности и апробация работы. Обоснованность и достоверность полученных результатов определяется корректным использованием математического аппарата, обеспечивается положительными итогами применения разработанных методов в экспериментальных проектах и подтверждается результатами внедрения на производстве. Эффективность предложенных методов и алгоритмов подкрепляется сравнительным анализом со стандартными подходами к решению подобного рода задач.

Основные положения и результаты диссертационной работы опубликованы в ведущих научных изданиях. Кроме того, они докладывались и обсуждались на научных конференциях, в том числе и международных: «IDC 2019 (Intelligent Distributed Computing)» (СПб, НИУ ИТМО), «CPSC 2019 (Cyber-Physical Systems and Control)» (СПбПУ), «Современные

технологии в теории и практике программирования» (СПбПУ, 2020 - 3 место, 2019 - 3 место, 2018 - 1 место), «Неделя науки СПбПУ» (18-23 ноября 2019 года, 19-24 ноября 2018 года).

Публикации. Основные положения диссертации изложены в 8 печатных работах, из них 2 работы опубликованы в журналах из перечня ВАК и 2 работы проиндексированы в Scopus.

Внедрение. Разработанные методы внедрены в рабочий процесс этапа сопровождения ПО компании ООО «Оупен Текст Технолоджи» на проекте InfoProject (система архивного хранения данных).

Структура и объем работы. Диссертация состоит из введения, четырех глав, заключения, списка литературы и одного приложения. Объем диссертации – 154 страницы, объем приложений – 3 страницы; диссертация содержит 72 рисунка, 12 таблиц, список литературы включает 147 названий.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении показана актуальность темы диссертации, определены цель и задачи проведенных исследований, отражена научная новизна и практическая значимость полученных результатов, приведены сведения об апробации работы и публикациях.

В первой главе рассмотрены этапы жизненного цикла разработки программного обеспечения (ПО), подробно проанализирован этап сопровождения с точки зрения его трудоемкости, проведен обзор схемы взаимодействия всех участников рабочего процесса на этапе сопровождения, тщательно описана работа инженеров на 3-й линии поддержки и обоснована актуальность разработки автоматизированных методов для снижения трудоемкости на данном уровне цепочки сопровождения, проанализированы используемые в процессе сопровождения инструменты и технологии, выявлены их достоинства и недостатки. Помимо обзора на Service Desk системы и системы отслеживания ошибок было проведено исследование плагинов для системы отслеживания ошибок Atlassian JIRA.

Существующий подход к рассмотрению и обработке запросов заказчиков в системе отслеживания ошибок был изучен и формализован в виде алгоритма. В полученном алгоритме выявлены шаги, которые являются трудоемкими и могут быть автоматизированы с помощью интеллектуальных методов. Диссертация содержит анализ существующих работ в области автоматизации этапа сопровождения, которые решают задачи поиска похожих запросов и семантически близких сегментов документации, автоматизации рутинных действий и выбора подходящего инженера. Эти направления исследований являются актуальными на сегодняшний день и получили свое развитие в диссертационной работе.

В первой главе представлен обзор и исследование подходов к обработке естественного языка, которые используются в автоматизированном анализе текстовых данных запросов. Помимо этого, рассматриваются научные труды в области информационного поиска. В главе описываются различные методы построения семантических структур, таких как онтологии предметной области, а также векторные модели дистрибутивной семантики, такие как: SAE, SDAE, ParagraphVectors, Word2Vec, Doc2VecC, SkipThought, FastSent, SIF, Glove, FastText, InferSent, USE, BERT, SBERT, SRoBERTa, Sent2Vec, ELMo. Предлагается обоснование того, что подход дистрибутивной семантики на основе методов машинного обучения более предпочтителен для разработки методов автоматизации этапа сопровождения ПО.

На основе проанализированных исследований выполнен обзор и сравнение алгоритмов создания векторных пространств слов и документов. Приведено обоснование выбора алгоритма Doc2VecC (Document Vectors through corruption) в качестве оптимальной технологии создания векторов из текстовых данных, которая необходима для разработки предлагаемых методов. Делается вывод о том, что данный алгоритм обладает необходимой точностью и скоростью обучения. Выдвигается предположение о том, что алгоритм Doc2VecC может быть усовершенствован для более эффективной работы с техническими текстами.

Во второй главе приводится описание разработанного алгоритма построения векторных пространств. Он является развитием алгоритма Doc2VecC для случая работы с техническими текстами, которые содержат множество именованных сущностей из области программной инженерии (названия функций, интерфейсов, модулей, программ, операционных систем и так далее). Отличительная особенность предлагаемого подхода заключается в использовании регулярных выражений вместо стохастических методов для выборки слов при формировании глобального контекста слова во время обучения нейронной сети алгоритма. Данная идея получила свое развитие на основе предположения о том, что суть технического текста определяется набором программных сущностей, включенных в него.

В главе предложены методы, направленные на автоматизацию перечисленных в главе 1 шагов, которые выполняют инженеры-разработчики во время обработки поступающих запросов от заказчиков. Автоматизация выполняется с применением предлагаемого алгоритма векторизации текстовых данных. Схема предлагаемой автоматизированной обработки запросов, с учетом интеграции четырех разработанных методов, приведена на Рисунке 1.

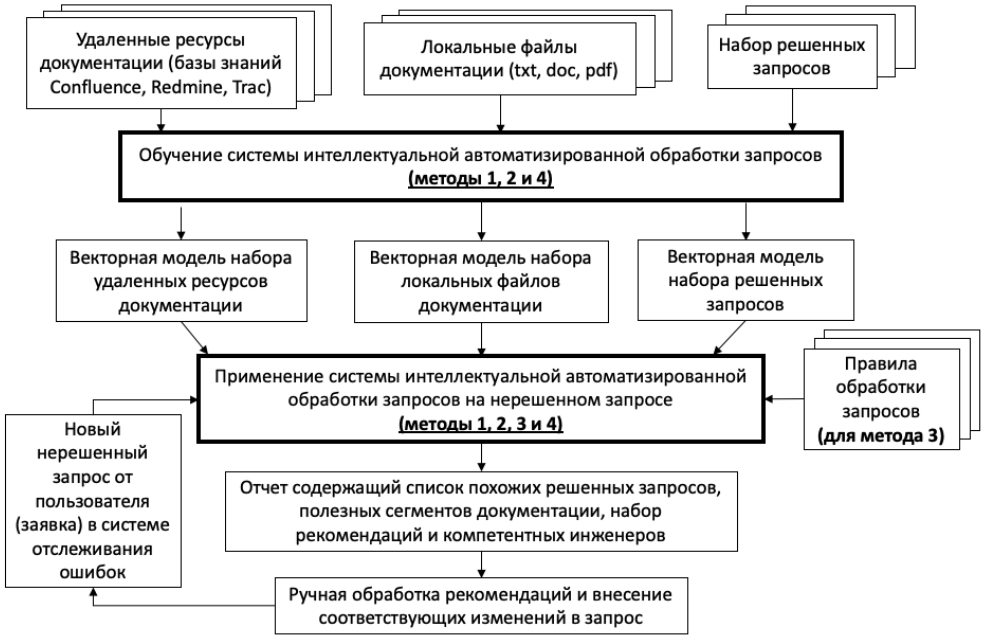

Рисунок 1 – Предлагаемая концептуальная схема обработки запросов заказчика

Ниже приведен список разработанных методов, а также особенностей их реализации.

1. Метод снижения трудоемкости этапа сопровождения за счет поиска семантически похожих решенных запросов для нерешенных запросов заказчика в системе отслеживания ошибок, который заключается в подготовке и векторизации текстовых данных запросов с помощью предложенного алгоритма построения векторных пространств и последующим нахождением таких решенных запросов, у которых критерий косинусной

близости между их векторными представлениями и векторным представлением нерешенного запроса больше порогового значения K и стремится к 1.

Этап поиска уже решенных запросов по имеющемуся нерешенному запросу в системах отслеживания ошибок является неотъемлемой частью рабочего процесса на этапе сопровождения ПО. Существующий классический подход к решению этой задачи заключается в поиске таких запросов с помощью ключевых слов в поисковом движке, который встроен в систему отслеживания ошибок.

Порой инженеру-разработчику очень сложно подобрать в поисковом запросе подходящую последовательность слов, которая бы в полной мере описывала искомую заявку. Кроме того, поисковые движки в системах отслеживания ошибок, как правило, не проводят предварительную обработку поисковых запросов (лемматизация, стемминг), из-за чего значительная часть заявок может быть не найдена по причине того, что в них содержатся слова с теми же основами, но другими окончаниями. Эта проблема особенно остро стоит в таких языках как русский, где окончание слова меняется в зависимости от многих факторов: падежа, склонения, числа, спряжения и т. д.

Для решения перечисленных проблем предлагается использовать семантический поиск по запросам в системе отслеживания ошибок на основе подходов дистрибутивной семантики. Суть данных подходов заключается в том, что каждому слову и документу сопоставляется определенный численный вектор, который определяет семантику слова или документа, и над которым можно производить математические операции сравнения. В качестве технологии построения векторных пространств предлагается использовать усовершенствованный вариант алгоритма Doc2VecC.

На Рисунке 2 изображена концептуальная схема метода. Работа с этим методом делится на 3 части: тренировка, распознавание и формирование результата. На этапе тренировки из исходных данных, которые представлены решенными запросами в системе отслеживания ошибок, извлекаются текстовые поля, несущие семантический смысл запроса (заголовок, описание, комментарии). Формируется структура данных, в которой каждая строка соответствует извлеченному решенному запросу. В начале строки стоит идентификатор запроса, а затем после символа разделителя находятся текстовые данные запроса. После этого происходит обработка текстов: удаление стоп-слов, удаление очень редких и часто встречаемых слов, стемминг. Предварительно обработанные тексты отправляются на вход предложенного алгоритма, который создает векторную модель слов из всего корпуса запросов. На основе векторов слов с помощью метода усреднения формируются вектора запросов. Структура данных, состоящая из векторов запросов и соответствующих им идентификаторов, называется векторной моделью запросов.

На этапе распознавания из нерешенного запроса извлекаются те же текстовые данные, что и на этапе тренировки. Затем каждое предварительно обработанное слово ищется в векторной модели слов решенных запросов. Из всех найденных векторов, путем усреднения, создается вектор нерешенного запроса. После этого наступает этап ранжирования и формирования результата, на котором с помощью косинусной меры путем полного перебора всех комбинаций вектора нерешенного запроса со всеми векторами решенных запросов, находятся и сортируются по убыванию все вектора решенных запросов, у которых косинусный коэффициент больше порогового значения K. Пороговое значение задается в

конфигурации программного средства. Далее, по имеющимся векторам можно получить идентификаторы запросов, после чего данные запросы извлекаются из системы отслеживания ошибок и добавляются в отчет с рекомендациями по решению входящего запроса.

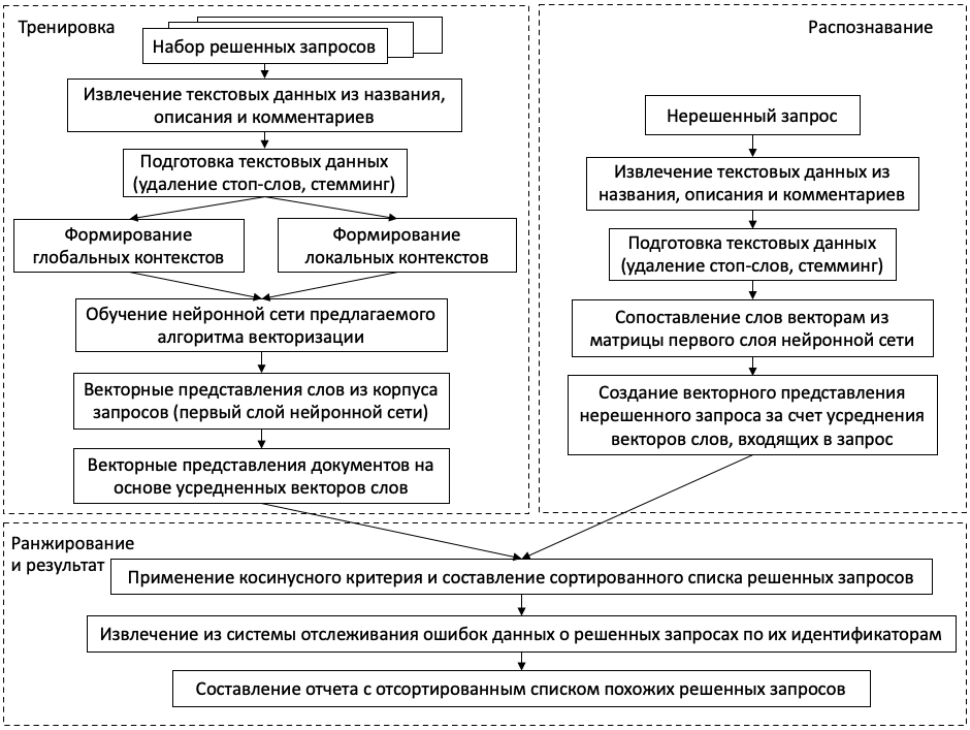

Рисунок 2 – Концептуальная схема метода 1

Предлагаемый метод может быть представлен в виде следующей формулы:

$$\forall u, R \in \mathbb{C}, \exists T \in \mathbb{N},\ F_1(u, V(R)) = \{r_1, r_2 \dots r_T\}, \qquad (1)$$

где u – нерешенный запрос, R – множество решенных запросов, $\mathbb{C}$ – множество всех запросов программного проекта, T – количество семантически близких решенных запросов, $F_1(u, V(R))$ – функция предлагаемого метода 1, $V(R)$ – функция построения векторной модели на множестве R, r_i – семантически похожий на u решенный i-ый запрос.

При этом в качестве метрики ранжирования набора решенных запросов выступает косинусная мера:

$$\forall M, S \in \mathbb{N}, \exists U, R_x, \{x \in \mathbb{N} | 1 \le x \le S\}, \{K \in \mathbb{Q} | 0 < K < 1\},$$

$$similarity(D_x, U) = \cos(\theta) = \frac{R_x * U}{\|R_x\| * \|U\|} =$$

$$= \frac{\sum_{i=1}^{M} R_{xi} U_i}{\sqrt{\sum_{i=1}^{M} R_{xi}^2} \sqrt{\sum_{i=1}^{M} U_i^2}} > K \to 1, \qquad (2)$$

где M – размер векторов, S – количество семантически похожих решенных запросов, U – вектор нерешенного запроса, R_x – вектор решенного запроса x, K – настраиваемый параметр, θ – угол между векторами решенного и нерешенного запроса.

2. Метод снижения трудоемкости этапа сопровождения за счет поиска семантически похожих на входящий запрос заказчика сегментов документации ПО, представленной в виде файлов и удаленных баз знаний. Метод заключается в подготовке и векторизации текстовых данных входящего запроса и документации с помощью предложенного алгоритма построения векторных пространств и последующим нахождением

таких сегментов документации, у которых критерий косинусной близости между их векторными представлениями и векторным представлением нерешенного запроса больше порогового значения K и стремится к 1.

На этапе сопровождения ПО инженерам-разработчикам часто приходится искать участки документации по имеющимся данным из входящего нерешенного запроса. Этот процесс особенно трудоемок, если программный продукт имеет очень богатую и запутанную документационную базу. Существующий подход к решению этой проблемы подразумевает подбор определенных ключевых слов и поиск по ним в программах просмотра локальных документов или в поисковых движках удаленных баз знаний.

При поиске полезных участков документации для решенных запросов инженеры-разработчики сталкиваются с теми же самыми проблемами, что и при поиске похожих уже решенных запросов. Этот трудоемкий и неэффективный процесс может быть значительно улучшен благодаря предлагаемому методу. Для того, чтобы решить проблемы, связанные с ключевым поиском по документам, предлагается делить исходные документы на смысловые сегменты (главы, параграфы и т. д.) и использовать семантический поиск по данным сегментам на основе подходов дистрибутивной семантики.

На Рисунке 3 изображена концептуальная схема метода 2. По тому же принципу, что и в методе 1, необходимо обучить предложенный алгоритм построения векторных пространств для получения двух новых векторных моделей: локальных файлов документации и удаленных ресурсов баз знаний. Используя эти векторные модели, формируются 2 соответствующих вектора нерешенного запроса. Благодаря косинусной мере происходит ранжирование результатов и на выходе алгоритма формируется 2 отсортированных по релевантности списка сегментов документации, которые семантически близки к исходному нерешенному запросу: по локальному и по удаленному корпусу сегментов.

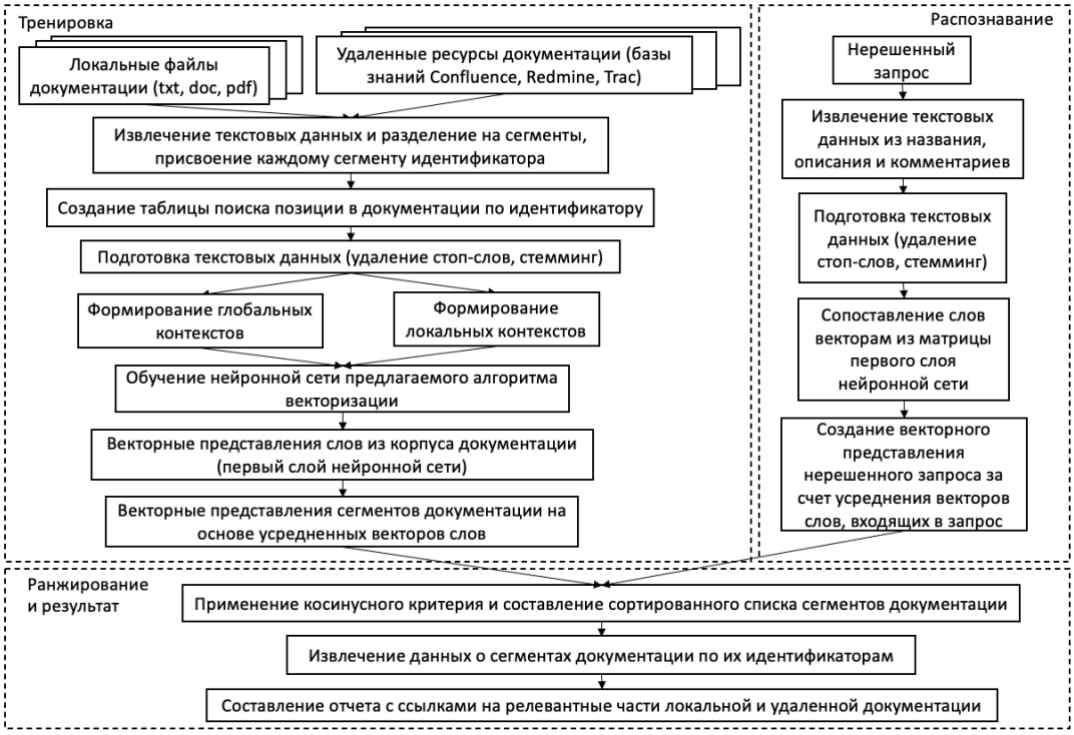

Рисунок 3 – Концептуальная схема метода 2

Предлагаемый метод может быть представлен в виде следующей формулы:

$$\forall u \in \mathbb{C}, \forall D, H \in \Omega, \exists T, S \in \mathbb{N},\ F_2\big(u, V(D), V(H)\big) = \\ = \{d_1, d_2 \dots d_T\} \cup \{h_1, h_2 \dots h_S\}, \qquad (3)$$

где u – нерешенный запрос, $\mathbb{C}$ – множество всех запросов программного проекта, D и H – множества сегментов локальной и удаленной документации соответственно, Ω – множество всех сегментов программной документации, T и S – количество семантически близких сегментов локальной и удаленной документации соответственно, $F_2\big(u, V(D), V(H)\big)$ – функция предлагаемого метода 2, $V(D)$ и $V(H)$ – функции построения векторной модели на множествах D и H соответственно, d_i – семантически похожий на u i-ый сегмент локальной документации, h_i – семантически похожий на u i-ый сегмент удаленной документации.

При этом в качестве метрики ранжирования набора сегментов документации выступает косинусная мера:

$$\forall M, S \in \mathbb{N}, \exists U, D_x, \{x \in \mathbb{N} | 1 \le x \le S\}, \{K \in \mathbb{Q} | 0 < K < 1\}, \\ similarity(D_x, U) = \cos(\theta) = \frac{D_x * U}{\|D_x\| * \|U\|} = \\ = \frac{\sum_{i=1}^{M} D_{xi} U_i}{\sqrt{\sum_{i=1}^{M} D_{xi}^2} \sqrt{\sum_{i=1}^{M} U_i^2}} > K \to 1, \qquad (4)$$

где M – размер векторов, S – количество семантически похожих сегментов документации, U – вектор нерешенного запроса, D_x – вектор сегмента документации x, K – настраиваемый параметр, θ – угол между векторами сегмента документации и нерешенного запроса.

В работе показано, что использование предложенного алгоритма построения векторных пространств в качестве подхода дистрибутивной семантики позволяет избежать недостатков поиска по ключевым словам и значительно снижает трудоемкость поиска похожих запросов и подходящей документации на этапе сопровождения ПО.

3. Метод сокращения времени обработки запросов заказчика в системе отслеживания ошибок за счет автоматизации рутинных действий, который заключается в применении правил анализа и обработки запросов, состоящих из условия и рекомендации. Если условие истинно для конкретного запроса, значит к нему может быть применима рекомендация. В качестве условия может быть состояние или содержимое любого из полей запроса в момент запуска механизма исполнения правил. Результатом работы метода является набор рекомендаций, который может быть применен в автоматизированном режиме. Примененная рекомендация является действием над запросом. В качестве действия может выступать любое разрешенное изменение полей запроса, например, изменение статуса запроса или написание комментария с вопросом инженеру технической поддержки.

Рисунок 4 содержит концептуальную схему работы метода. Ключевым объектом на схеме является набор интерпретируемых написанных человеком правил. На каждом нерешенном запросе выполняются все правила анализа из набора. Результатом работы метода является список рекомендаций, каждая из которых может быть предоставлена в виде текстового описания или в виде автоматического действия над запросом. Только определенный тип рекомендаций может быть применен в автоматическом режиме.

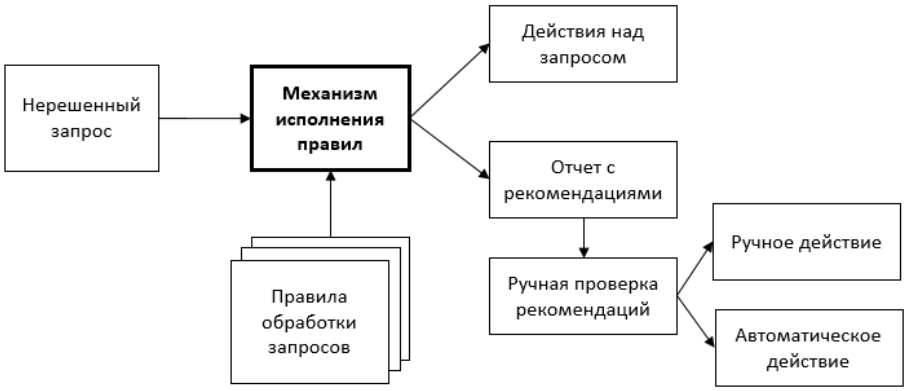

Рисунок 4 – Концептуальная схема метода 3

Пример правила с действием: «Если последний комментарий написан 5 дней назад от исполнителя запроса и содержал слова «ошибка исправлена», то перевести запрос в статус закрыто». Пример правила с рекомендацией без действия: «Если последний комментарий был 5 дней назад от создателя запроса, то необходимо ответить на вопрос».

Предлагаемый метод может быть представлен в виде следующей формулы:

$$\forall u \in \mathbb{C}, \forall L, \exists T \in \mathbb{N},\ F_3(u, L) = \{p_1, p_2 \dots p_T\}, \qquad (5)$$

где u – нерешенный запрос, $\mathbb{C}$ – множество всех запросов программного проекта, L – множество правил обработки запросов, T – количество рекомендаций на выходе, $F_3(u, L)$ – функция предлагаемого метода 3, p_i – i-ая рекомендация.

Разработанный в диссертации метод основан на механизме исполнения правил. На основе формальных правил анализа и обработки запроса становится возможным снижение трудоемкости работы над запросом, за счет избавления инженера-разработчика от рутинных действий.

4. Метод снижения трудоемкости этапа сопровождения за счет автоматизированного выбора сотрудников, имеющих опыт и соответствующую экспертизу в решении поступившего запроса заказчика. Данный метод основан на методе 1 и его отличительной особенностью является извлечение имен инженеров из семантически похожих решенных запросов заказчика в системе отслеживания ошибок и суммирование их вхождений в эти запросы, при этом вероятность того, что инженер сможет принести пользу рассчитывается с помощью функции softmax. Имена инженеров извлекаются из следующих информационных полей запроса: Исполнитель, Создатель, Автор комментария.

Рисунок 5 содержит концептуальную схему работы метода, который является логичным продолжением метода 1. Имея набор отсортированных решенных запросов с коэффициентом косинусной близости больше порога K, каждому инженеру можно сопоставить количество его вхождений в запросы. Под вхождением подразумевается факт того, что инженер является создателем, исполнителем или автором комментариев. За каждое вхождение счетчик инженера увеличивается на единицу. В итоге получается вектор, где каждая позиция соответствует определенному инженеру, а значение в этой позиции равно количеству его вхождений. Для получения вероятности того, что инженер сможет помочь в решении текущего нерешенного запроса, необходимо применить функцию softmax к этому вектору и получить распределение вероятности того, что инженер сможет помочь.

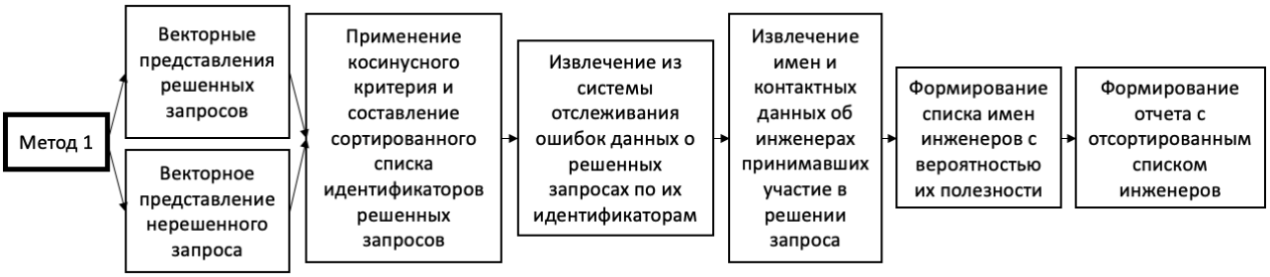

Рисунок 5 – Концептуальная схема метода 4

Предлагаемый метод может быть представлен в виде следующей формулы:

$$\forall u \in \mathbb{C}, \exists T \in \mathbb{N},\ F_4(u, F_1) = \{w_1, w_2 \dots w_T\}, \qquad (6)$$

где u – нерешенный запрос, $\mathbb{C}$ – множество всех запросов программного проекта, T – количество найденных инженеров по решенным запросам, F_1 – результат функции предлагаемого метода 1, w_i – i-ый инженер, обладающий необходимой экспертизой.

В качестве классификатора используется функция softmax (обобщение логистической функции для многомерного случая); каждому инженеру w_i сопоставляется вероятность его полезности, полученная с помощью этой функции:

$$\forall T, M \in \mathbb{N},\ F_1 = \{r_1, r_2 \dots r_T\}, z = \sum_{i=1}^{T} Q(r_i),$$

$$\sigma(z) = softmax(z), \sigma(z)_i = \frac{e^{z_i}}{\sum_{k=1}^{M} e^{z_k}}, \qquad (7)$$

где T – количество семантически близких решенных запросов, M – общее количество инженеров, принимавших участие в решении T запросов, r_i – i-ый семантически близкий решенный запрос, $Q(x) = [q_1, q_2, \dots, q_M]$ – бинарный вектор инженеров из запроса x, где $q_i(x) = \begin{cases} 0, \text{если } i - \text{ый инженер не участвовал в запросе } x \\ 1, \text{если } i - \text{ый инженер участвовал в запросе } x \end{cases}$, $\sigma(z)_i$ – координата вектора σ.

Применение метода позволяет автоматизировать процесс выбора инженера с необходимым уровнем экспертизы для помощи в решении проблемы, тем самым снижая трудоемкость обработки запросов на этапе сопровождения ПО.

В третьей главе представлено описание программного комплекса, который реализует предложенные методы. Глава содержит модульную архитектуру системы, состав дистрибутива, библиотеки и технологии, использовавшиеся при разработке. Помимо этого, приводится список конфигурационных параметров для корректной настройки инструмента.

В четвертой главе представлено описание оптимизации гиперпарамеров предложенного алгоритма построения векторных пространств, внутри которого используется трехслойная нейронная сеть. Использованный метод оптимизации – случайный поиск. Для оценки качества алгоритма использовался подход кросс-валидации на специально подготовленном размеченном наборе данных.

Кроме того, в главе 4 представлены результаты внедрения разработанных методов в процесс обработки запросов заказчика на стадии сопровождения ПО. Здесь проанализированы показатели эффективности применения в трех экспериментальных проектах: Apache Kafka – распределенный программный брокер сообщений, проект с открытым исходным кодом; Apache ActiveMQ – брокер сообщений с открытым исходным кодом, реализующий спецификацию JMS 1.1; InfoProject – проприетарная система архивного хранения данных.

Подготовка к эксперименту заключалась в извлечении решенных запросов из системы отслеживания ошибок, извлечении документации из удаленных баз знаний, обучении моделей и написании правил обработки запросов для механизма исполнения правил.

Задача проведения эксперимента заключалась в том, чтобы выяснить насколько применение разработанного автоматизированного подхода позволяет снизить трудоемкость решения запросов на этапе сопровождения ПО. В главе также определяются критерии, по которым запрос считается решенным.

В решении запросов проектов Apache Kafka и Apache ActiveMQ участвовала группа студентов СПбПУ в составе 10 человек. В решении запросов проекта InfoProject участвовали разработчики компании «Оупен Текст Технолоджи» в составе 10 человек. Программное средство было внедрено в реальный рабочий процесс на предприятии в проекте InfoProject. Студенты и инженеры, участвовавшие в эксперименте, обладают разной квалификацией и разной степенью экспертизы. Каждая группа из 10 инженеров (студентов) была поделена на 2 равные части по 5 человек: 1) инженеры, решающие запросы с помощью ручного подхода, 2) инженеры, решающие запросы с помощью автоматизированного подхода.

Для каждого проекта было выбрано по 30 нерешенных запросов. Чтобы избежать неоднозначности, перед началом эксперимента и на его протяжении все наборы входных данных (решенные запросы, корпус документации и правила обработки) были зафиксированы. Таймер включался, когда инженер (или студент) начинал работу над новым запросом, и выключался, когда инженер временно прерывал или заканчивал решение запроса.

Результаты, полученные в рамках применения разработанных методов, свидетельствуют о сокращении трудоемкости обработки запросов. Рисунок 6 иллюстрирует показатели сокращения трудоемкости в трех экспериментальных проектах. Сравнительные диаграммы показывают, что автоматизированный подход требует значительно меньше времени, чем существующий подход, для анализа 30 запросов.

Рисунок 6 – Диаграммы, показывающие затраченное в часах время каждым инженером с помощью ручного и автоматизированного подхода

Для понимания на сколько процентов в среднем снизилась трудоемкость при автоматизированном подходе, необходимо применить формулу расчета среднего времени, потраченного на решение одного запроса:

$$avg = \frac{\sum_{i=1}^{5} t_i^{Kafka} + \sum_{i=1}^{5} t_i^{ActiveMQ} + \sum_{t=1}^{5} t_i^{InfoProject}}{N * M * T}, \qquad (8)$$

где t_i – суммарное время затраченное инженером на решение 30 запросов в одном проекте, $N = 5$ – количество инженеров, обрабатывающих запросы проекта ручным или автоматизированным способом, $M = 30$ – количество решенных запросов в одном проекте, $T = 3$ – количество экспериментальных проектов.

В итоге, среднее время, потраченное на решение одного запроса по всем проектам и инженерам в ручном подходе составляет 4,61 часов, а в автоматизированном – 1,89 часов. Результаты, полученные в рамках применения предлагаемого автоматизированного подхода на трех экспериментальных проектах, свидетельствуют о выигрыше во времени (сокращении трудоемкости) в среднем в 2,4 раза по сравнению с существовавшим ручным подходом. Данный результат является существенным для промышленного сопровождения программного продукта.

Кроме расчета снижения трудоемкости, в работе также было вычислено качество работы векторных моделей и эффективность применения правил обработки запросов. Для оценки качества методов 1, 2 и 4 была использована метрика ранжирования mAP@K (mean average precision at K), которая рассчитывается с помощью следующих формул:

$$precision@K = p@K = \frac{|\{relevant\ pages\} \cup \{retrieved\ pages\}|}{|\{retrieved\ pages\}|} = \frac{\sum_{k=1}^{K} rel(k)}{K}, \quad (9)$$

$$ap@K = \frac{\sum_{k=1}^{K}(rel(k) * p@K)}{K}, mAP@K = \frac{\sum_{j=1}^{N} ap@K_j}{N}. \quad (10)$$

На Рисунках 7 и 8 изображены диаграммы с оценкой качества ранжирования в разработанных методах, основанных на работе алгоритма Doc2VecC и основанных на предложенном алгоритме построения векторных пространств, соответственно. Выигрыш в точности составил 15,8%.

Рисунок 7 – Оценка качества работы методов 1, 2 и 4 с алгоритмом Doc2VecC

Рисунок 8 – Оценка качества работы методов 1, 2 и 4 с предложенным алгоритмом векторизации

Для оценки качества метода 3 были использованы метрики бинарной классификации (в качестве классов выступают правила). Для каждого правила обработки запросов в каждом из трех проектов считались метрики precision, recall и F1-score по 30 запросам с помощью формул:

$$precision = \frac{TP}{TP+FP}, recall = \frac{TP}{TP+FP}, F1 = 2\frac{precision * recall}{precision + recall}. \quad (11)$$

Затем по каждому из трех проектов считались средние значения precision, recall и F1-score по всем 14 правилам. На Рисунке 9 изображены диаграммы с оценкой качества работы метода 3, который основан на механизме применения правил к нерешенным запросам.

	Kafka	ActiveMQ	InfoProject
Avg. Precision	0,8179	0,8615	0,8734
Avg. Recall	0,8879	0,8725	0,8897
Avg. F1-score	0,8515	0,8670	0,8815

Рисунок 9 – Оценка качества работы метода 3

Графики показывают, что не все предложенные решенные запросы, сегменты документации, правила обработки, выбранные инженеры оказались полезными для решения входящих нерешенных запросов, при этом качество ранжирования и точность применения правил находятся на достаточно высоком уровне.

ЗАКЛЮЧЕНИЕ

Поставленная цель диссертационной работы, которая заключалась в сокращении трудоемкости обработки запросов заказчиков в системах отслеживания ошибок за счет методов автоматизации этапа сопровождения ПО, достигнута.

Основные результаты, полученные в ходе выполнения работы, перечислены ниже.

1. Разработан алгоритм построения векторных пространств для случая работы с техническими текстами из области программной инженерии.

2. Предложен комплекс методов по автоматизации сопровождения программного обеспечения на основе интеллектуальной обработки запросов заказчика:

2.1. Метод поиска семантически похожих решенных запросов заказчика в системе отслеживания ошибок.

2.2. Метод нахождения семантически похожих на входящий запрос заказчика сегментов документации ПО.

2.3. Метод автоматизации рутинных действий за счет применения разработанных правил анализа и обработки запроса, состоящих из условия и действия.

2.4. Метод автоматизированного выбора сотрудников, имеющих соответствующую экспертизу в решении поступившего запроса заказчика.

3. Определены оптимальные гиперпараметры алгоритма построения векторных пространств для эффективной работы предлагаемых методов и сформулированы правила обработки запросов.

4. Разработан программный комплекс на языке Java, реализующий предложенные в работе методы.

5. Снижена трудоемкость этапа сопровождения за счет совместного применения реализованных в работе методов в среднем в 2,4 раза по сравнению с существующим в настоящее время подходом на трех экспериментальных проектах: Apache Kafka, Apache ActiveMQ, InfoProject.

6. Обосновано преимущество использования разработанного алгоритма построения векторных пространств, показавшего выигрыш в точности на 15,8% по сравнению с существующим алгоритмом Doc2VecC на примере текстов из области программной инженерии.

7. Оценена точность применения правил анализа и обработки запросов.

8. Предлагаемые методы внедрены в процесс обработки запросов заказчиков на этапе сопровождения промышленного проекта InfoProject, за счет чего была заметно снижена нагрузка на сотрудников и затраты на сопровождение.

9. Отдельные разработки диссертации внедрены в учебный процесс Санкт-Петербургского политехнического университета Петра Великого.

Практические результаты, полученные в ходе работы, позволяют сделать вывод о том, что поставленные исследованием задачи решены, и констатировать достижение общей цели работы.

СПИСОК РАБОТ, ОПУБЛИКОВАННЫХ АВТОРОМ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Ковалев А.Д., Никифоров И.В., Дробинцев П.Д. Автоматизированный подход к обнаружению семантически близких запросов заказчика в системе отслеживания ошибок Jira // Современная наука: актуальные проблемы теории и практики. Серия: Естественные и технические науки. – 2021. – №. 5-2. – С. 61-67. (**Издание из перечня ВАК**)

2. Ковалев А.Д., Никифоров И.В., Дробинцев П.Д. Автоматизированный подход к семантическому поиску по программной документации на основе алгоритма Doc2Vec // Информационно-управляющие системы. – 2021. – №. 1. – С. 17-27. (**Издание из перечня ВАК, статья проиндексирована в Scopus**)

3. Kovalev A., Voinov N., Nikiforov I. Using the Doc2Vec algorithm to detect semantically similar jira issues in the process of resolving customer requests // Studies In Computational Intelligence. – Springer, Cham, 2020. – Т. 868, С. 96-101. (**Статья проиндексирована в Scopus**)

4. Sokolova A., Safronov D., Stonozhenko K., Solomonov M., Nikiforov I., Kovalev A. Artificial Intelligence Methods for Services and Product Sustaining Phase // Conference of Open Innovations Association, FRUCT. – 2021. – №. 28. – С. 645-651.

5. Соколова А.Е., Соломонов М.С., Ковалев А.Д., Никифоров И.В. Использование методов машинного обучения на этапе обслуживания программного продукта // Современные технологии в теории и практике программирования. – 2020. – С. 129-131.

6. Kovalev A.D., Sheremetov S.D., Nikiforov I.V. Algorithm for Searching Semantically Related Jira Issues to Automate Customer Requests Resolving // Cyber-Physical Systems and Control. – 2020. – С. 83-94.

7. Соколова А.Е., Соломонов М.С., Ковалев А.Д., Никифоров И.В. Поиск по документации в системе автоматизации решения запросов заказчика // Неделя Науки СПбПУ. – 2019. – С. 87-90.

8. Ковалев А.Д., Никифоров И.В., Дробинцев П.Д. Интеллектуальная обработка запросов заказчика на этапе сопровождения программного продукта // Неделя Науки СПбПУ. – 2019. – С. 165-168.