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

Институт радиоэлектроники и информационных технологий – РТФ
Школа профессионального и академического образования

ДОПУСТИТЬ К ЗАЩИТЕ ПЕРЕД ГЭК

РОП 09.04.01 Аксенов К.А.

_____ _____
(подпись) (Ф.И.О.)

«____» _____ 2023 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

ИССЛЕДОВАНИЕ И РАЗРАБОТКА ПРОТОТИПА ВОПРОСНО-ОТВЕТНОЙ СИСТЕМЫ

Научный руководитель
к.т.н., доцент _____ (подпись) Аксенов К.А.

Нормоконтролер _____ (подпись) Аксенов К.А.

Студент группы РИМ-210960 _____ (подпись) Алейникова А.А.

Екатеринбург

2023

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

Институт радиоэлектроники и информационных технологий - РТФ
Школа профессионального и академического образования
Направление подготовки: 09.04.01 Информатика и вычислительная техника
Образовательная программа: Информационно-управляющие системы

УТВЕРЖДАЮ
Директор ДИТиА Аксенов К.А.
«____» _____ 2023 г.

ЗАДАНИЕ

на выполнение магистерской диссертации

студента Алейниковой Алёны Алексеевны **группы** РИМ-210960
(фамилия, имя, отчество)

1. Тема магистерской диссертации Исследование и разработка прототипа вопросно-ответной системы
Утверждена распоряжением по ИРИТ-РТФ от 02.12.2022 г. № 33.02-05/286

2. Руководитель Аксенов Константин Александрович, доцент, к.т.н., директор департамента Информационных технологий и автоматики
(Ф.И.О., должность, ученое звание, ученая степень)

3. Исходные данные к работе Литература, информация из Интернета, учебные материалы, материалы практик и научные статьи

4. Перечень демонстрационных материалов Презентация в формате PowerPoint

5. Календарный план

№ п/п	Наименование этапов выполнения работы	Срок выполнения этапов работы	Отметка о выполнении
1.	*Сбор материалов, постановка задачи*	до 31.12.2022 г.	
2.	*Выполнение задания на ВКР*	до 30.04.2023 г.	
3.	*Магистерская диссертация в целом*	до 16.05.2023 г.	

Руководитель _____ Аксенов К.А.
(подпись) Ф.И.О.
Задание принял к исполнению _____ _____ Алейникова А.А.
дата (подпись)

6. Магистерская диссертация закончена «___» _____ 20__ г. считаю возможным допустить Алейникову Алёну Алексеевну к защите её магистерской диссертации в Государственной экзаменационной комиссии.
Руководитель _____ Аксенов К.А.
(подпись) Ф.И.О.

7. Допустить Алейникову Алёну Алексеевну к защите магистерской диссертации в Государственной экзаменационной комиссии (протокол заседания кафедры №___ от «___» _____ 20__г.).

РОП 09.04.01 _____ Аксенов К.А.
(подпись) Ф.И.О.

РЕФЕРАТ

Объем выпускной квалификационной работы: пояснительная записка объемом в 85 страниц содержит в себе 5 глав, 25 иллюстраций, 2 таблицы, 33 использованных источника, 2 приложения.

Ключевые слова: искусственный интеллект, машинное обучение, нейронная сеть, вопросно-ответная система, обработка естественного языка, алгоритмы поиска информации, семантический анализ.

Объектом исследования и разработки являются вопросно-ответные системы.

Цель работы: исследовать существующие методы анализа текста и существующие вопросно-ответные системы. Разработать прототип вопросно-ответной системы, протестировать его и оценить результаты.

Методы проведения работы: обзор и изучение статей, исследований, анализ информации и статистическое исследование, разработка и тестирование прототипа приложения, оценка и сравнение полученных результатов.

Результаты работы: в результате был проведено исследование существующих методов анализа текста и существующих вопросно-ответных систем. Был спроектирован и реализован прототип вопросно-ответной системы, было выполнено тестирование и оценка полученных результатов.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

NLP — обработка естественного языка (Natural Language Processing);

QA – Вопросно-ответная система (Question-answering system);

АОТ – Автоматическая обработка текста;

БД – База данных;

ВОС – вопросно-ответная система;

ВОП – вопросно-ответный поиск;

ЕЯ – Естественный язык;

ИИ – Искусственный интеллект;

ИП – Информационный поиск;

ИС – Информационная система;

МО – Машинное обучение;

НС – Нейронная сеть;

ОС – Операционная система;

ПС – Поисковая система;

СУБД – Система управления базами данных.

Вопросно-ответная система — информационная система, которая позволяет пользователям задавать вопросы и получать на них ответы на естественном языке.

Естественный язык — это язык, который используется людьми для общения между собой. Он отличается от формальных языков, таких как математические символы или язык программирования, и имеет сложную структуру, включающую грамматику, лексику и синтаксис. Естественный

язык может быть говоренным или письменным и имеет множество диалектов и вариантов в зависимости от региона и культуры.

Искусственный интеллект — это область компьютерных наук, которая занимается созданием и разработкой программ и систем, способных выполнять задачи, которые ранее могли выполнять только люди (например, творческие). Используя методы и алгоритмы машинного обучения, искусственный интеллект может учиться и принимать решения на основе данных, что позволяет ему решать сложные задачи в различных областях

Машинное обучение — это подход в области искусственного интеллекта, который позволяет компьютерным системам учиться и принимать решения на основе опыта и данных, без явного программирования. С помощью алгоритмов машинного обучения, компьютер может анализировать большие объемы данных, выявлять закономерности и обучаться на основе этой информации. Для построения методов машинного обучения используются средства математического анализа, математической статистики, теории графов, численных методов, методов оптимизации, различные техники работы с данными в цифровой форме, теории вероятностей.

Нейронная сеть — это алгоритм машинного обучения, который моделирует работу человеческого мозга. Она состоит из множества взаимосвязанных "нейронов", которые принимают на вход данные, обрабатывают их и передают результат дальше по сети. Каждый нейрон имеет свой вес, который определяет важность входных данных для решения задачи. Нейронные сети используются для решения различных задач, таких как распознавание образов, классификация данных, прогнозирование результатов и т.д. Они могут обучаться как методом наблюдения за результатами, так и методом обратного распространения ошибки.

Сниппет — это краткое описание контента веб-страницы, которое отображается в результатах поисковой выдачи. Сниппет содержит заголовок страницы, краткое описание ее содержания и URL-адрес. Он помогает пользователям быстро понять, насколько релевантна страница их запросу и принять решение о ее посещении.

Фреймворк (framework) — это набор инструментов, библиотек и правил, которые используются для разработки программного обеспечения. Он предоставляет разработчикам готовые решения для ряда задач, таких как обработка данных, управление базами данных, создание пользовательских интерфейсов и многое другое. Фреймворки упрощают процесс разработки, ускоряют время создания приложений и позволяют сосредоточиться на бизнес-логике приложения, а не на технических деталях. Они также помогают обеспечить совместимость и безопасность приложений, что делает их более надежными и стабильными.

ВВЕДЕНИЕ

Одним из направлений в области искусственного интеллекта сегодня является обработка текстов на естественном языке. На анализе естественного языка основана разработка вопросно-ответных систем (от англ. Question Answering Systems) – компьютерных программ, предназначенных для ответа на вопросы. В отличие от ПС, которые выдают список релевантных документов, ВОС пытаются найти конкретный ответ на заданный вопрос. Для этого они используют различные методы и алгоритмы обработки ЕЯ, семантического анализа и поиска информации. ВОС могут быть использованы в различных областях, таких как медицина, право, финансы, образование и другие.

Исследование ВОС может включать в себя изучение различных методов и алгоритмов, используемых для поиска и обработки информации. Например, это может быть анализ текстовых данных, поиск по базам знаний, МО и НС. Также исследование может включать в себя оценку качества работы ВОС, например, путем проведения тестов и сравнения результатов с ответами экспертов в соответствующей области.

Другой важный аспект исследования вопросно-ответных систем — это разработка новых методов и алгоритмов для улучшения их работы. Например, это может быть разработка новых моделей МО или улучшение алгоритмов ИП.

Актуальность исследования ВОС обусловлена несколькими причинами. Во-первых, с развитием Интернета и цифровых технологий количество информации, доступной пользователям, растет в геометрической прогрессии. В этой связи возрастает необходимость в эффективных инструментах поиска и обработки информации.

Во-вторых, ВОС имеют широкое применение в различных областях, от медицины до бизнеса. Они помогают экономить время и снижать затраты на поиск и анализ информации.

В-третьих, развитие ИИ и МО позволяет создавать все более точные и эффективные ВОС. Исследование в этой области может привести к созданию новых методов и алгоритмов, которые улучшат работу таких систем.

В-четвертых, ВОС русского языка недостаточно развиты. Разработка ВОС русского языка имеет свои особенности и вызовы, связанные с особенностями русского языка и его грамматикой. Например, в русском языке существует склонение и спряжение, что усложняет задачу обработки запросов и выдачи наиболее релевантных ответов. В отличие от английского языка, для которого существует большое количество текстов и баз знаний, русский язык не имеет такого широкого охвата. Это создает проблемы при обучении моделей и требует большего количества времени и усилий для сбора и обработки данных. Русский язык имеет множество диалектов и говоров, которые могут отличаться по лексике, грамматике и произношению, что создает проблемы при разработке ВОС, так как они должны быть способны понимать различные варианты выражений и ответов. Русский язык часто использует многозначные слова и выражения, которые могут иметь различные значения в зависимости от контекста, что требует от разработчиков ВОС на русском языке учета контекста и создания алгоритмов, которые могут понимать смысл выражений в контексте.

Таким образом, исследование вопросно-ответных систем является актуальным и важным для различных областей науки и технологий.

Целью научно-исследовательской работы является исследование существующих методов анализа текста и существующих ВОС. Разработка прототипа ВОС для русского языка, его тестирование и оценка результатов.

Новизна научных и практических результатов в исследовании и разработке прототипа ВОС заключается в исследовании языковых моделей и практическом применении одной их них, являющейся перспективной.

Гипотеза исследования – ВОС применяются во многих областях, но на данный момент не всегда могут заменить человека. Для исключения человеческого фактора предполагается, что необходимо приложить больше сил к обучению НС.

Задачи исследования:

– исследовать существующие методы анализа текста;

– исследовать существующие ВОС;

– исследовать этапы работы ВОС;

– разработать прототип ВОС;

– протестировать разработанный прототип ВОС;

– оценить результаты тестирования.

Методы исследования, которые использованы в данной работе:

– обзор и изучение научных статей, исследований, посвящённых НС и ИИ;

– анализ информации, статистическое исследование;

– разработка прототипа приложения;

– тестирование прототипа приложения и оценка результатов.

Объект исследования – искусственный интеллект, вопросно-ответные системы, языковые модели.

Предмет исследования – методы и алгоритмы обработки ЕЯ, семантического анализа и ИП, методы извлечения ответа.

Теоретическая основа исследования – научные статьи, исследования, разработки, доклады, книги.

1 ИССЛЕДОВАНИЕ ВОПРОСНО-ОТВЕТНЫХ СИСТЕМ

1.1 Вопросно-ответный поиск

Вопросно-ответный поиск – это вид ИП, при котором на заданный вопрос пользователем на ЕЯ ПС находит ответ на веб-страницах или в БД. В отличие от современных систем ИП, позволяющих получить список документов, которые могут содержать информацию, интересующую пользователя, ВОС получают вопросительное предложение на ЕЯ (русском, английском и т.д.), а не ключевые слова, и возвращают пользователю краткий ответ, а не множество ссылок и документов. Например, пользователь задает вопросительное предложение: «Какой океан самый большой на планете Земля?», и в качестве ответа получает наименование океана, а не список ссылок, которые релевантны запросу, на текстовые документы. Тем не менее ПС браузера пользуются большой популярностью среди пользователей [31].

Рассмотрим ПС Яндекс. Согласно исследованию Яндекс [8] каждый день пользователи, говорящие на русском языке, задают в среднем 100 миллионов запросов в день, из которых приблизительно 2,5% запросов сформулированы как вопрос. Слово, с которого чаще всего начинаются вопросительные запросы – «Какой?», реже – «Сколько?» и «Кто?». Многие пользователи не знают про существование ВОС и предпочитают обращаться к ПС и находить ответ самостоятельно. Пользователи могли бы сократить время поиска, обратившись к обученным моделям.

Таким образом, подготовка ответа на вопрос, путем извлечения необходимого отрывка текста из документа, содержащего сам ответ непосредственно, является совершенно другой задачей в отличие от ИП. Ответ на вопрос, сгенерированный ВОС, должен быть актуальным, точным и кратким.

В системах ВОП используются технологии обработки естественных языков (от английского «natural language processing», NLP) [1, 2]. Обработка ЕЯ – это область компьютерной науки, которая занимается разработкой технологий для анализа, понимания и генерации ЕЯ. В NLP используются различные алгоритмы и методы, включая машинное обучение, статистический анализ, семантический анализ и другие. Некоторые из технологий включают в себя распознавание речи, машинный перевод, анализ тональности, ВОП, анализ сентиментов, генерация текста и извлечение информации.

Типичный алгоритм работы ВОП можно представить следующим образом (рисунок 1):

Рисунок 1 – Алгоритм работы ВОП

ВОС между собой могут отличаться как видом ИП, так и предметной областью вопросов, которые они могут обрабатывать. ВОС могут быть как узкоспециализированые, так и всесторонние. Узкоспециализированные ВОС обычно используются для определенной области знаний, например, медицина, юриспруденция и т.д. [28], для таких систем используются локальные хранилища данных, содержащие знания данной области.

Интеллектуальный ИП в настоящее время широко исследуется. Разработка алгоритмов поиска и обработки информации одни из актуальных тем научных конференций в мире.

Некоторые научные конференции, которые содержали темы интеллектуального поиска информации:

– ACM SIGIR (Association for Computing Machinery Special Interest Group on Information Retrieval) – это ежегодная конференция, посвященная исследованию и разработке технологий поиска информации [22];

– CIKM (Conference on Information and Knowledge Management) – это ежегодная конференция, которая объединяет исследователей и практиков в области управления информацией и поиска [4];

– ECIR (European Conference on Information Retrieval) – это ежегодная конференция, которая объединяет исследователей и профессионалов в области ИП и извлечения информации [26];

– WSDM (Web Search and Data Mining) – это ежегодная конференция, которая объединяет исследователей и профессионалов в области поиска веб-страниц и анализа данных [27];

– ICTIR (International Conference on Theory on Information Retrieval) – это конференция, которая сосредоточена на теоретических аспектах ИП и извлечения информации [14].

1.2 Исследование существующих вопросно-ответных систем

1.2.1 Google Assistant

Google Assistant [3] – виртуальный помощник, который использует технологии голосового управления и ВОП для предоставления пользователю информации и выполнения задач. Он работает с ОС Android и iOS, а также на устройствах Google Home.

Google Assistant получает запрос от пользователя через голосовой интерфейс или текстовый ввод, далее анализирует запрос с помощью NLP, включая распознавание речи и машинный вод. Следом производит поиск подходящих ответов в БД Google и на веб-страницах с использованием ключевых слов и фраз, затем производит ранжирование найденных ответов по степени соответствия вопросу и другим параметрам, таким как релевантность и авторитетность источника. Виртуальный помощник предоставляет пользователю наиболее подходящий ответ пользователю в виде голосового ответа или текстового сообщения (рисунок 2).

Google Assistant также может выполнять другие задачи, такие как управление устройствами умного дома, отправка сообщений и выполнение расчетов. Он может запрашивать у пользователя дополнительную информацию, чтобы лучше понимать запрос и предоставить более точный ответ.

Рисунок 2 – Пример ответа Google Assistant

1.2.2 Apple Siri

Apple Siri [5] – это облачный виртуальный помощник и ВОС, которая использует голосовое управление для ответа на вопросы и выполнения задач на устройствах Apple, таких как iPhone, iPad, Apple Watch и Mac.

Siri может ответить на широкий спектр вопросов, от простых запросов на поиск информации до более сложных задач, таких как создание напоминаний, отправка сообщений и управление устройствами умного дома.

Пример ответа Siri на вопрос: «Какой океан самый большой в мире?» изображен на рисунке 3.

Siri использует МО и ИИ для улучшения и адаптации к предпочтениям пользователя. Она может запоминать предыдущие запросы и контекст, чтобы предложить более точный и персонализированный ответ на будущие запросы.

Siri — это разработка Международного центра искусственного интеллекта SRI. Для Siri Apple использовала результат 40-летних исследований «Центра Искусственного Интеллекта» (отдела SRI International).

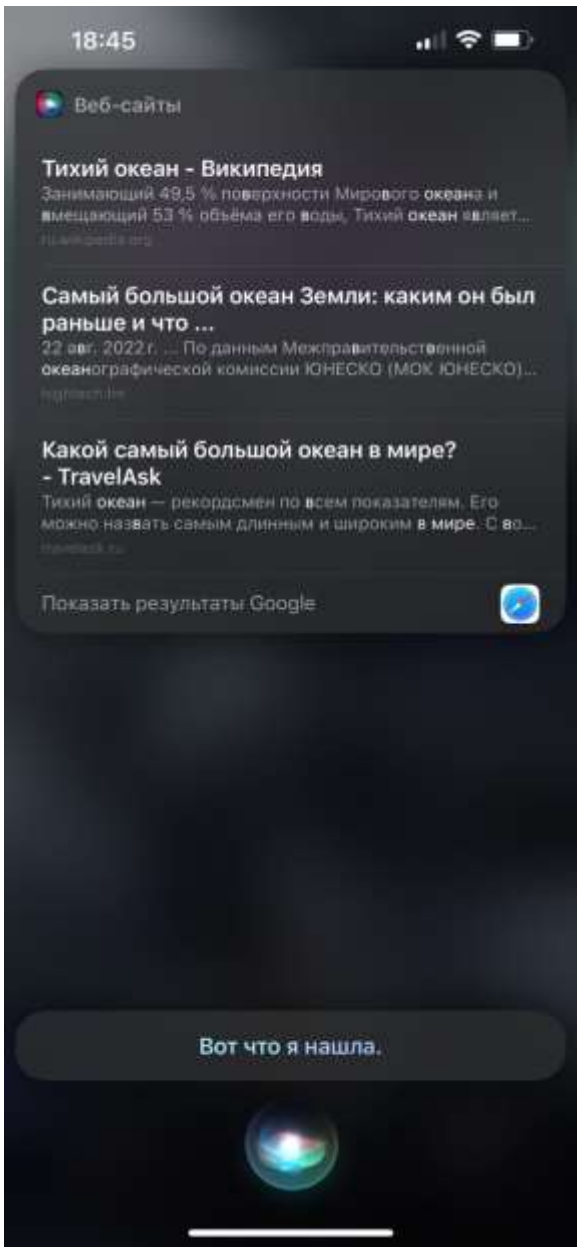

Рисунок 3 – Пример ответа Siri

1.2.3 IBM Watson

IBM Watson является одной из наиболее продвинутых ВОС в мире. Watson — суперкомпьютер, состоящий из множества серверов IBM p750 (на момент 2011 года их было 90), оснащённых процессорами архитектуры POWER7 (на момент 2011 года каждый сервер содержал 4 восмиядерных процессора). Объём оперативной памяти Watson в сумме — более 15 терабайт [7]. Он использует ИИ и МО для анализа и обработки большого объема данных и поиска ответов на сложные вопросы.

Одним из основных преимуществ IBM Watson является способность обрабатывать ЕЯ, что позволяет пользователям задавать вопросы в обычной форме, без необходимости использовать специальные ключевые слова или фразы.

IBM Watson также использует методы МО для постоянного улучшения своей производительности. Он может анализировать большие объемы текстовой информации, чтобы выявлять связи между словами и понимать контекст вопросов, что помогает предоставлять более точные и полезные ответы.

Доступ к ВОС закрытый, протестировать работу IBM Watson не удалось.

1.2.4 Wolfram Alpha

Wolfram Alpha [15] – это программа, ВОС, которая использует знания и алгоритмы математики, физики, химии, биологии и других наук для ответа на вопросы. Она может рассчитывать математические выражения, строить графики, выполнять статистический анализ, а также предоставлять информацию о фактах и концепциях.

WolframAlpha является одним из ярких примеров ВОС экспертного типа. Данная система не является интеллектуальной ПС, имея схожий интерфейс. Алгоритм WolframAlpha основан на NLP и поддерживает исключительно английский язык. Система находит ответ, используя собственную большую библиотеку алгоритмов и NKS-подход (рисунок 5).

Исходный код рассматриваемой системы написан на языке Mathematica, он состоит примерно из пяти миллионов строк и исполняется в настоящее время примерно на десяти тысячах процессорах.

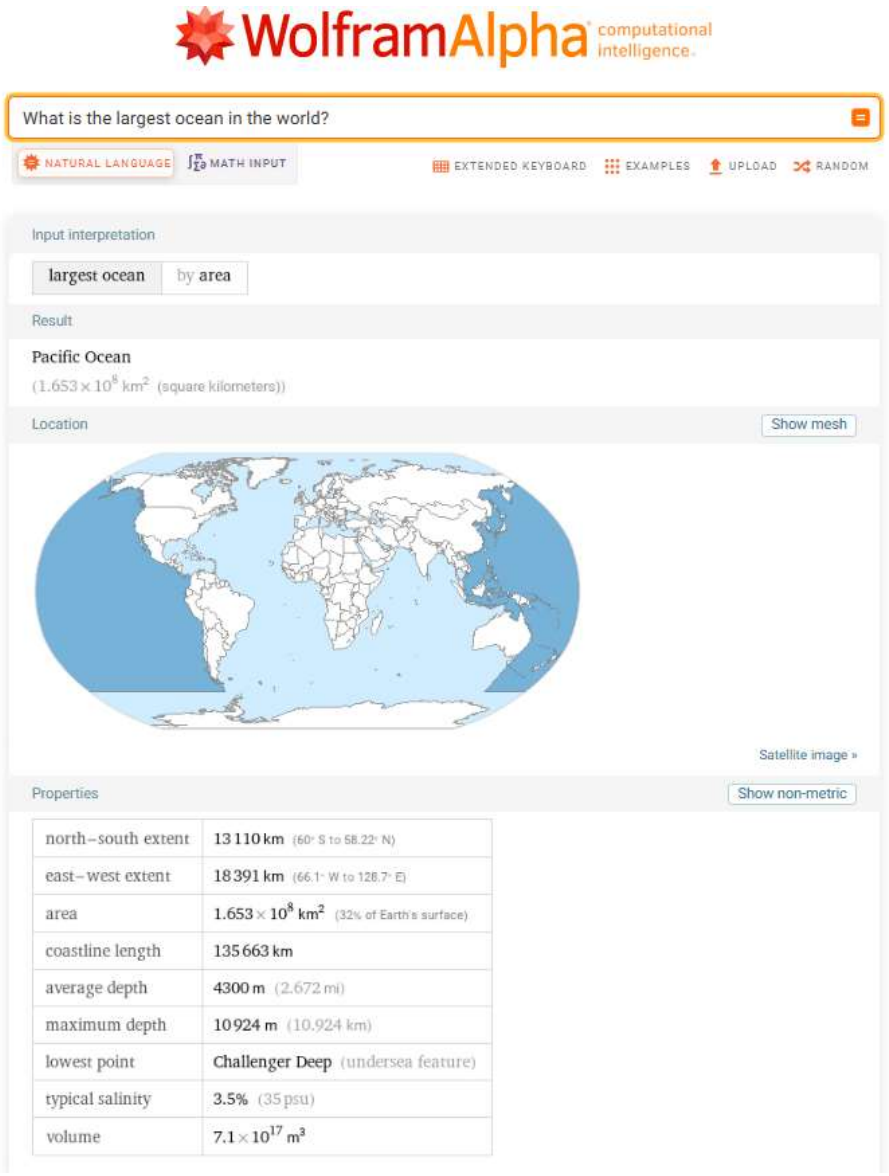

Рисунок 4 – Пример ответа системы WolframAlpha

1.3 Виды вопросно-ответных систем

ВОС можно разделить на две основные группы [29]:

– общие – вид ВОС, ориентированный на ответ на любые вопросы или большинство их. Такие ВОС для извлечения ответа на заданный пользователем вопрос часто используют Интернет. Общие ВОС чаще всего обрабатывают информацию, которая не структурирована, в связи с чем должны иметь сложную внутреннюю структуру и использовать различные технологии NLP;

– узкоспециализированные (узкого назначения) – вид ВОС, который рассчитан на вопросы определенной тематики или предметной области (медицина, искусство, и т.д.). Такие ВОС используют в основном локальные хранилища данных на тему предметной области.

Также ВОС можно классифицировать по методам реализации ВОС с учетом таких отличительных черт, как способ ИП, вид и наличие информационной БД, потребность привлечения экспертов определенной тематики, архитектурная гибкость ВОС, вычислительный ресурс [33].

По подходам к реализации и принципам построения ВОС можно подразделить на следующие виды:

– ВОС, которые используют веб-поиск (с английского «web-based question answering system») [17];

– ВОС, содержащие коллекцию вопросов и ответов;

– ВОС, содержащие коллекцию индексированных документов;

– экспертные ВОС.

1.3.1 Вопросно-ответные системы, которые используют веб-поиск

ВОС, которые используют веб-поиск [2, 17], в качестве системы поиска информации они используют результат работы ПС, который является веб-страницами или их фрагментами. В данном случае в архитектуру ВОС (рисунок 6) входит какая-либо из существующих ПС. Алгоритм работы такой ВОС может быть следующим: ВОС получает вопросительное предложение на ЕЯ от пользователя, далее обрабатывает вопрос и создает запрос из ключевых слов к ПС. Ключевые слова подбираются исходя из самого вопроса. В результате поискового запроса ВОС получает результат работы ИП в виде набора ссылок и небольших отрывков текста, которые называются сниппетами.

Сниппет обычно содержит контекст, который содержит в себе ключевое слово, найденное в тексте веб-страницы. Далее ВОС обрабатывает данные фрагменты, используя методы NLP, и генерирует пользователю ответ. Для обработки текстов веб-страниц обычно используются морфологические, синтаксические и семантические методы анализа текста [17], подробно описанные в п. 2.1.

Преимущества данных ВОС заключаются в следующем:

– нет потребности в хранении и индексировании большого размера текстовой информации;

– наличие в архитектуре ВОС ПС освобождает разработчиков от решения задачи разработки методов нахождения релевантных запросу документов.

Недостатки данных ВОС заключаются в следующем:

– точность и качество полученного ответа зависит от выбранной ПС. Несмотря на то, что в настоящее время популярные ПС обычно выдают релевантные результаты, результатом работы может быть недостоверная или неактуальная информация;

– результатом работы ПС является сниппет, его размер нельзя изменить и информация, содержащаяся в нем, не структурирована.

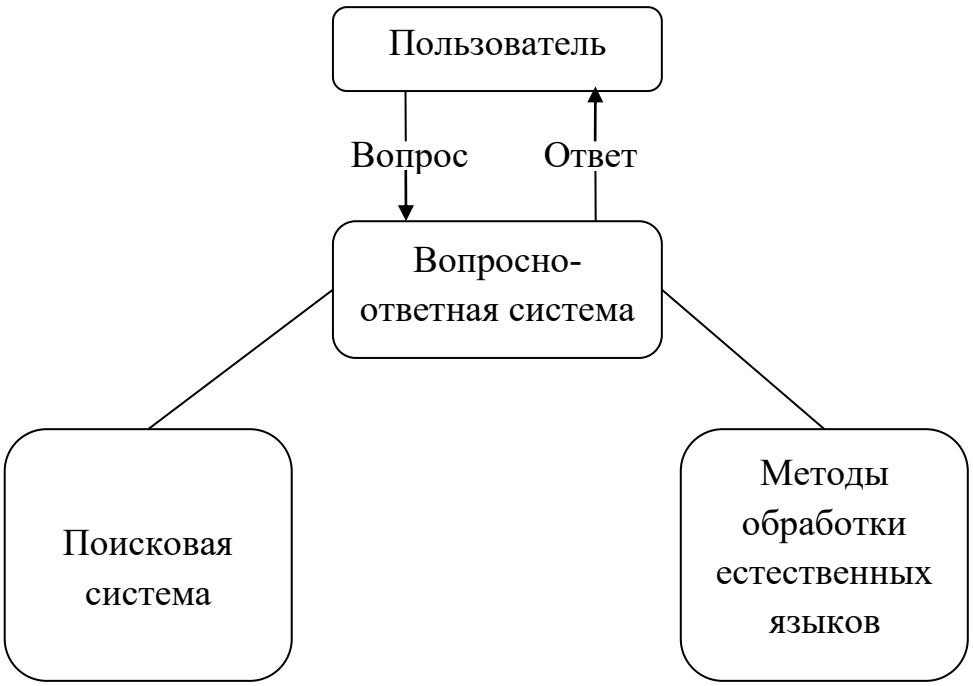

Рисунок 5 – Обобщенная архитектура ВОС, которые используют веб-поиск

1.3.2 Вопросно-ответные системы, содержащие локальную коллекцию вопросов и ответов

Другим видом ВОС, отличающимся подходом к разработке, являются ВОС, содержащие коллекцию вопросов и ответов. Архитектура данной системы предусматривает необходимость хранилища, содержащего локальную коллекцию вопросов и соответствующих им ответов (рисунок 7). Такую систему можно организовать через социальную систему, являющеюся веб-порталом, посредством создания открытой базы вопросов и ответов. На созданном веб-портале пользователь имеет возможность задать вопрос или ответить на вопрос другого пользователя. Пользователь системы задает вопрос на ЕЯ, затем ВОС ищет похожий вопрос в созданной коллекции и выдаёт ранее сгенерированный ответ на вопрос пользователя.

Система выдает наиболее похожий вопрос, где ранее пользователи добавили свои ответы, в начале списка ответов располагается тот, за корректность которого проголосовало большинство пользователей. В случае, если похожего на заданный вопрос не нашлось – пользователю предлагается добавить вопрос в коллекцию и дождаться ответов от других пользователей. Исходя из этого, можем сделать вывод, что данные в такой системе представлены в виде коллекции вопросов с ответами, пополняемой другими пользователями. Также БД с вопросами и ответами может автоматически пополняться с помощью сети Интернет и анализа текста, выявляя посредством NLP пары «вопрос-ответ».

Преимущества данных BOC заключаются в следующем:

– пользователи могут дать развернутый и полный ответ;

– нет необходимости разработки алгоритмов обработки объемных текстов, поиска релевантных ответов;

– точность и корректность ответов проверяют другие пользователи.

Недостатки данных BOC заключаются в следующем:

– потребность в разработке открытой социальной системы с коллекцией вопросов и ответов;

– потребность во времени для сбора и дальнейшего пополнения коллекции;

– необходимость в привлечении пользователей;

– потребность в создании и поддержании объемного хранилища данных.

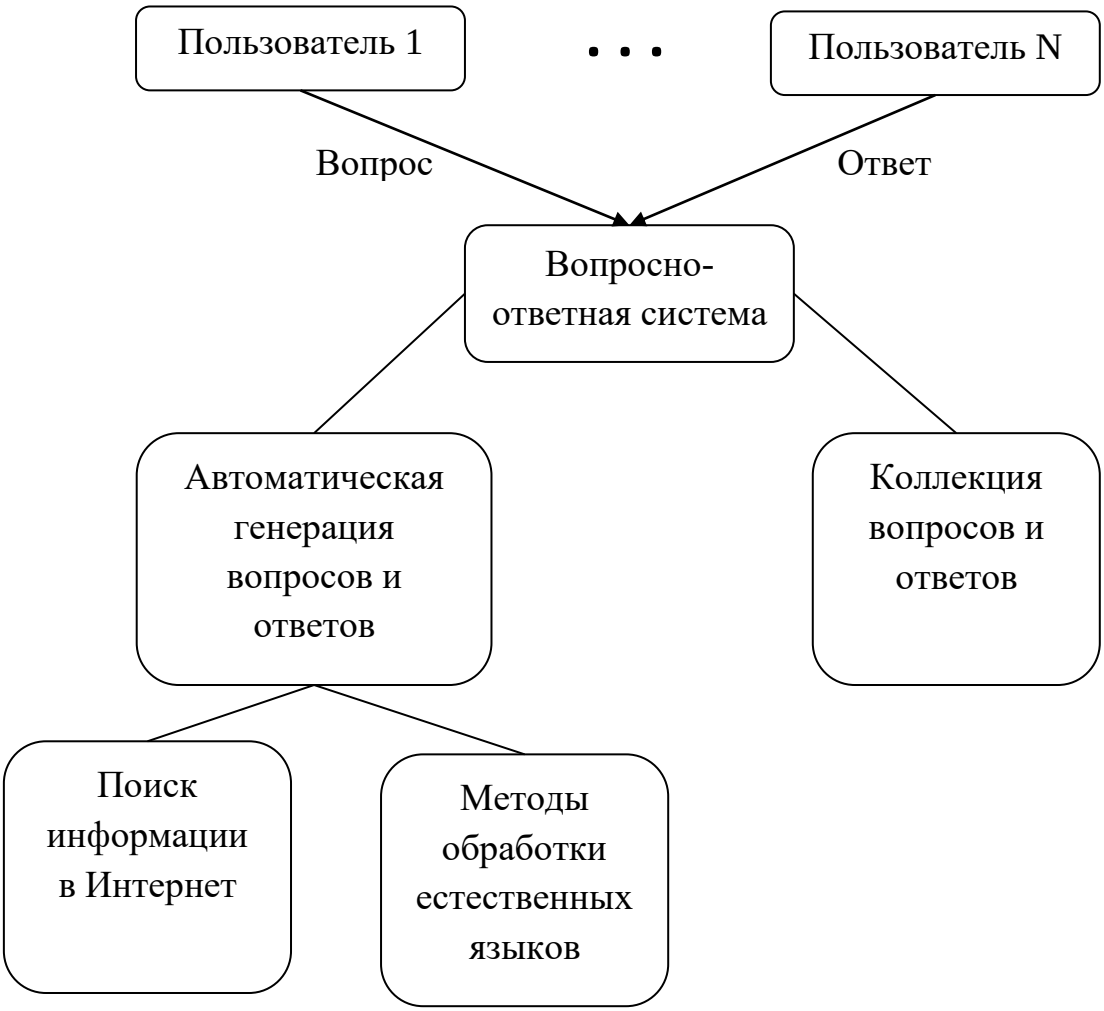

Рисунок 6 – Обобщенная архитектура ВОС, содержащих коллекцию вопросов и ответов

1.3.2 Вопросно-ответные системы, содержащие коллекцию индексированных документов

Применение локальной коллекции индексированных документов является еще одним подходом к реализации ВОС. Архитектура таких ВС предусматривает необходимость модуля поиска по индексу документов и модуль поиска нахождения документов, соответствующих запросу пользователя (рисунок 8). Для каждого нового документа коллекции строится поисковый индекс, который отличается от индексов классических

ПС. Обычно элементами поискового индекса являются не ключевые слова, а объекты лингвистического анализа [9]:

– именованные сущности;

– синтаксические связки.

Такие объекты выделяют из документа с помощью NLP.

Преимущества данных BOC заключаются в следующем:

– самостоятельное индексирование и поиск по индексу;

– потребность в меньшей вычислительной мощности при обработке вопроса посредством самостоятельной индексации документов.

Недостатки данных BOC заключаются в следующем:

– изменения в алгоритме индексации потребуют изменения BOC в алгоритме, отвечающем за индексирование документов и поиск по индексу;

– необходимость индексирования документов нуждается в больших вычислительных затратах.

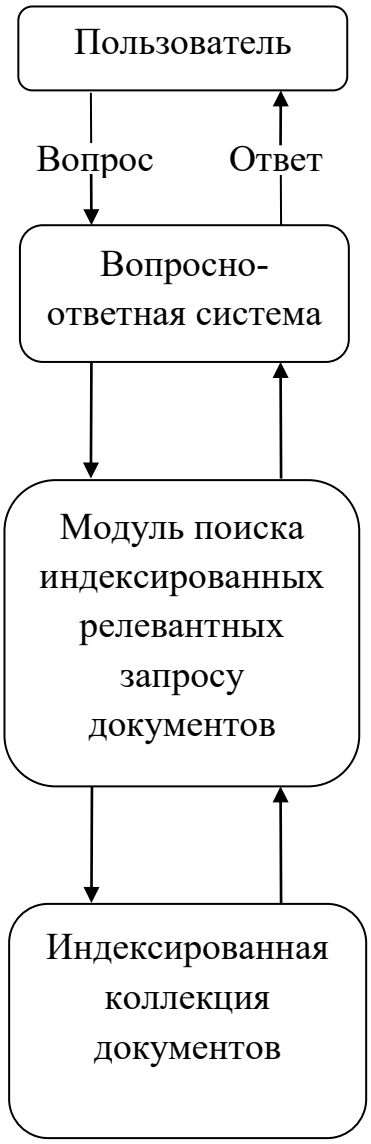

Рисунок 7 – Обобщенная архитектура ВОС, содержащей локальную коллекцию индексированных документов

1.3.4 Экспертные вопросно-ответные системы

Другой тип ВОС по методу реализации – экспертные ВОС. Как и во всех ВОС пользователь задает вопрос системе, при этом ВОС может потребовать пояснения и задать наводящие вопросы пользователю [30]. Вопрос, заданный на ЕЯ, анализируется алгоритмами, далее внутренним алгоритмом, который для поиска ответа использует большую БД, составляется решение вопроса и возвращается ответ. Чаще всего экспертные

ВОС содержат в базе знаний документы на определенную тематику или предметную область, такие ВОС являются узкоспециализированными.

Архитектура ВОС такого типа (рисунок 9) подразумевает наличие специализированной базы знаний. Такая база знаний должна содержать исключительно достоверную информацию о тематике или предметной области. Факты являются единицами знаний в такой базе, кроме них в базе содержатся правила, которые называются продукциями. Каждая продукция – это программа из выражений, имеющая следующий вид: «ЕСЛИ <выражение1>, ТО <выражение2>», где «выражение» – факт, содержащий логические операции «И/ИЛИ». С помощью таких продукций ВОС может получать актуальные данные о тематике, новые данные также будут достоверными. Все действия с продукциями выполняет эксперт предметной области, например, добавление, удаление, изменение и др. Базу знаний таких ВОС пополняет исключительно эксперт.

Преимущества данных ВОС заключаются в следующем:

– достаточно высокая скорость работы;

– высокая точность ответа.

Недостатки данных ВОС заключаются в следующем:

– потребность в привлечении экспертов определенной тематики и необходимость поддержания системы;

– потребность в создании объемной БД, которая будет содержать структурированную информацию и продукции.

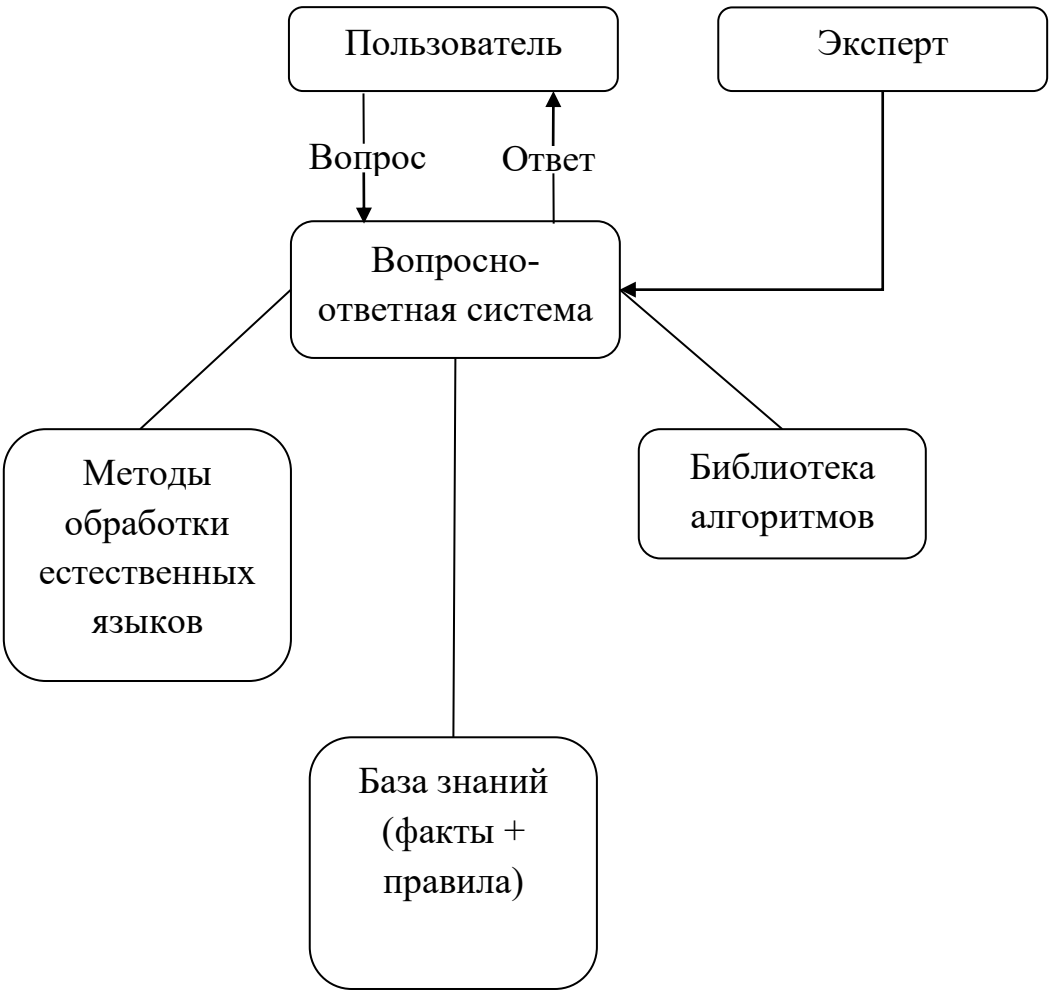

Рисунок 8 – Обобщенная архитектура ВОС экспертного типа

Обычно такая ВОС предусматривает постоянную работу экспертов определенной тематики.

– необходимость в чётком понимании потребности пользователей, а также регулярное расширение и усовершенствование модели данных для других предметных областей;

– для извлечения достоверной информации необходимо наличие исходных текстов;

– для создания базы фактов требуются большая организационная и вычислительная трудоемкость.

Приведенное деление ВОС по принципам разработки и архитектуре является традиционным, но есть примеры ВОС, которые объединяют в себе различные подходы и способы реализации. К примеру, описанная в п. 1.2.3 суперкомпьютерная система проекта DeepQA – IBM Watson [10]. IBM Watson производит поиск информации как с помощью веб-поиска, так и при помощи поиска по локальному хранилищу документов.

1.4 Этапы работы вопросно-ответных систем

Основная часть существующих ВОС создана для работы с английским языком. Исследуя некоторые работы по ВОС, можно сделать вывод, что процесс работы ВОС можно разделить на несколько этапов (рисунок 10):

– обработка и анализ вопроса;

– ИП ответа;

– извлечение ответа.

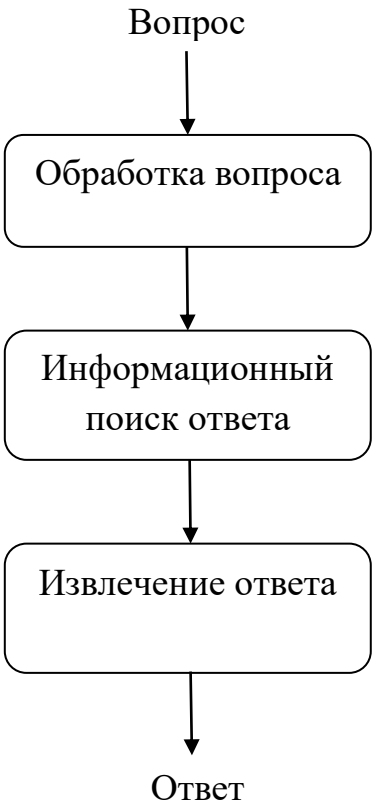

Рисунок 9 – Последовательность этапов работы ВОС

Дополнительно можно выявить этапы:

– оценка и интерпретация ответа;

– предоставление ответа;

– обратная связь и дополнительные вопросы.

1.4.1 Обработка и анализ вопроса

На данном этапе пользователь вводит вопрос на ЕЯ и происходит обработка и анализ вопроса. Обычно современные ВОС обрабатывают некоторые заранее определенные виды вопросов. По виду ответа их можно классифицировать следующим образом:

– вопросы о фактах;

– вопросы мнения;

– вопросы причины.

Вопросы о фактах – это вопросы, которые предполагают краткий ответ и не требуют обобщения или анализа. Примерами вопросов о фактах могут являться вопросы о личностях, о времени или вопросы, которые требуют ответа «да» или «нет». Например, вопросы могут быть следующими: «Какой океан самый большой в мире?», «Почему море голубого цвета?».

Большинство ВОС обрабатывают именно вопросы о фактах, так как они не требуют развернутого ответа.

Вопросы о фактах можно классифицировать следующим образом:

– вопросы, которые требуют ответа «да» или «нет»;

– вопросы о личностях;

– вопросы о географических местах;

– вопросы о составе чего-либо;

– вопросы об определениях и др.

Более сложная процедура ответа требуется для вопросов мнения и вопросов причины. Для получения ответа на такие вопросы необходим логический анализ данных и выявление причинно-следственной связи между словами и предложениями. Обычно ответы в данном случае являются развернутыми и не всегда достоверными, так как существующие разработки в области анализа текстовых документов и вывода их структурной логики достаточно плохо справляются с типами таких вопросов. Вопросы мнения предполагают анализ каких-либо высказываний автора, они требуют глубокой проработки блогов или сайтов средств массовой информации.

На начальном этапе анализа вопроса определяется тип вопроса, а затем соответствующий ему класс ответа. Данные действия нужны для того, чтобы определить тип ожидаемого ответа (с английского «expected answer type»). После определения типа ответа ВОС будет игнорировать предложения, которые не содержат информацию необходимого типа. Многие ВОС содержат в себе систему, которая поддерживает типы ожидаемого ответа, иногда используется таксономия, имеющая вид иерархической структуры, в которой описана стратегия поиска и извлечения ответа (рисунок 11). Разные ВОС могут иметь различную классификацию ответов. В IBM Watson, упоминаемой в п. 1.2.3, встроена таксономия, состоящая примерно из 2500 типов ожидаемого ответа.

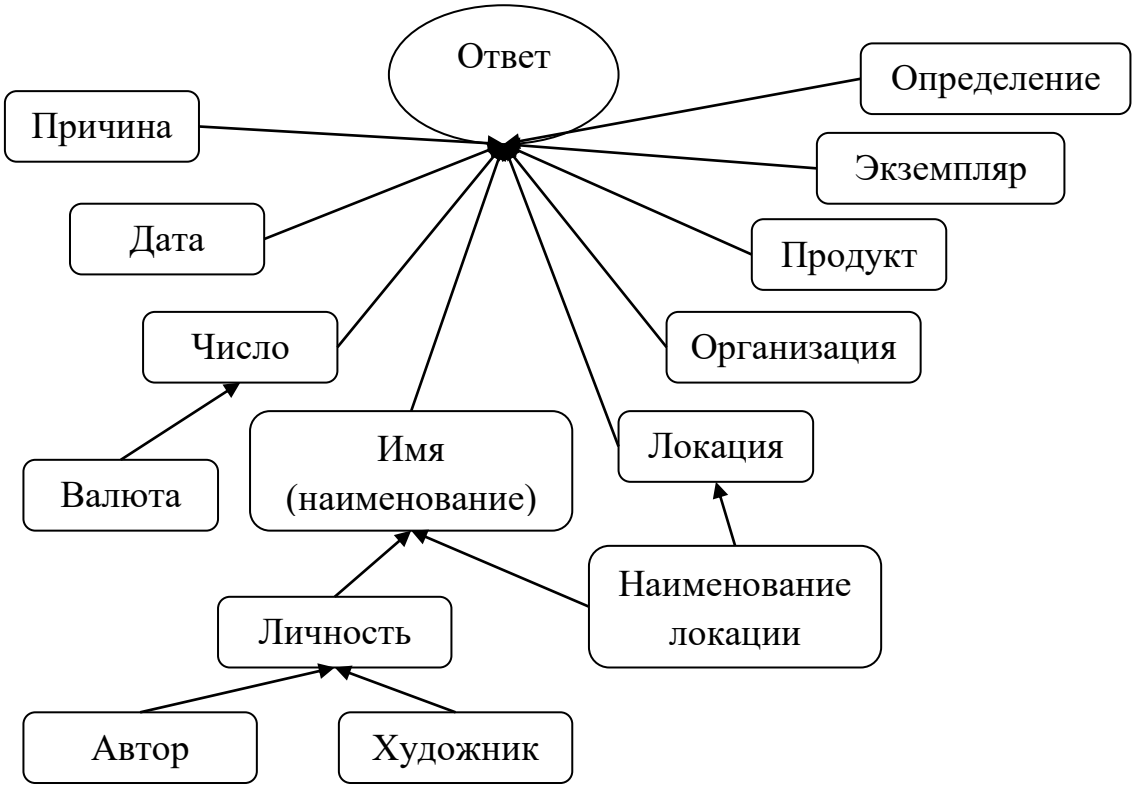

Рисунок 10 – Таксономическое дерево типов ответов

Чаще всего тип ожидаемого ответа распознается посредством использования списка регулярных выражений или подготовленных шаблонов. Для каждого подготовленного шаблона вопроса существует класс ответа. В таблице 1 содержатся возможные примеры шаблонов и соответствующих им классов ответов, которые могли бы использоваться при разработке алгоритмов анализа текста. Обычно такие шаблоны создаются вручную, но с развитием данной области шаблоны могут создаваться автоматически по принципам МО. К примеру, система Webclopedia [25] содержит в себе более двести восьми десяти шаблонов, которые написаны вручную и соответствуют ста восьмидесяти типам ответов. Для определения необходимого шаблона вопроса производится его синтаксический анализ, чтобы получить информацию о частях речи в нем и наличии именованных сущностей.

Таблица 1 – Примеры классов ответов и шаблонов вопросов

Класс ответа	Примеры шаблонов
Имя	/(Кк)ак зовут+/?
Личность (персона)	/[Кк]то+/?
Число	/[Кк]ак много+/?, /[Сс]колько +/?
Вещь	/[Чч]то+/?, /[Чч]ем является+/?
Дата	/[Кк]огда +/?, /[Вв](каком) году+/?
Локация	/[Гг]де +/?, /[Кк]уда +/?, /[Оо]ткуда +/?
Столица	/[Кк]акая столица +/?, /[Кк]акой город является столицей +/?
Дата смерти	/[Кк]огда .* умер(ла) +/?, /[Вв]каком году .* умер(ла) +/?
Дата рождения	/[Кк]огда .* родил(ся/лась) +/?, /[Вв]каком году .* родил(ся/лась) +/?

С помощью рассмотренного метода определения класса ожидаемого ответа ограничивается большинство возможных ответов и определяется дальнейшая стратегия извлечения ответа.

Недостатки рассматриваемого метода:

– невозможно создать шаблоны для всех классов ответов. Шаблоны создаются под какое-либо множество подобных вопросов, но задать вопрос, который не будет подходить под шаблон, но подразумевать определенный класс ответа достаточно легко;

– связь между ожидаемыми классами ответов и шаблонами не всегда является прямолинейной. К примеру, вопросительное предложение, начинающееся со слова: «кто», может подразумевать не только личность, но и группу персон (например, ответом на вопрос: «Кто придумывает народные приметы?», будет являться слово: «народ»).

Еще одной задачей, которая решается на данном этапе работы ВОС, является поиск ключевых слов для того, чтобы сформировать запрос к поисковому модулю ВОС. Такие слова определяют смысл вопроса. Состав набора ключевых слов напрямую зависит от типа ВОС.

Если ВОС базируется на веб-поиске и используется современные ПС (Яндекс, Google и т.д.) в запросе к ним можно оставить сам вопрос, заданный пользователем, так как ПС устойчивы к стоп-словам.

Во время формирования запроса к поисковому модулю из вопроса можно удалить вопросительные слова (такие, как: «что», «как», «кто» и т.д.), посредством чего улучшается поиск, так как такие слова практически не встречаются в ответах. Исключением вопросительных слов можно повысить такой критерий оценки качества ВОС, как полнота (с английского: «recall»), который будет рассмотрен в п. 3.2.

Другим способом формулирования запроса к поисковому модулю является перефразирование заданного пользователем вопроса в неполное утвердительное предложение.

Пример перефразирования в утвердительное неполное предложение:

– «Кто первым вышел в космос?» → «Первым вышел в космос».

Ещё одним способом перефразирования, используемом в ВОС, является использование синонимизация. Система заменяет неизвестные ей слова на похожие.

Пример перефразирования с использованием синонимизации:

– «Кто первым вышел в космос?» → «Кто первым вышел в космическое пространство».

Этап анализа и обработки вопроса является очень важным в работе ВОС. Если модуль анализа текста будет работать некорректно, то это будет причиной некорректной отработки следующих этапов

После формирования запроса к поисковому модулю алгоритм ВОС переходит на следующий этап – этап ИП.

1.4.2 Информационный поиск ответа

На этапе ИП происходит получение релевантных запросу документов или фрагментов текста, которые могут содержать ответ на вопрос пользователя. В современных ВОС этот модуль является стандартной поисковой машиной, которая получает на вход запрос из ключевых слов. После получения набора релевантных запросу текстовых документов, извлекаются фрагменты, которые с большой вероятностью могут содержать ответ на вопрос. Для получения таких фрагментов, текст документа необходимо разделить на части – способом деления на абзацы, который наиболее распространен в ВОС. Ответ будет извлечен на следующем этапе из

того абзаца, в котором находятся все ключевые слова или большинство их в сравнении с другими абзацами. В системе ВОП, которая разработана в Южном Методистском Университете США [24] используется отличный от предыдущего способ получения фрагментов текстовых документов, в котором применен более сложный алгоритм поиска фрагмента. Текст релевантного документа делится на какие-либо параграфы. В данном алгоритме используется понятие окна параграфа – окно содержит в себе весь текст параграфа от первого включения ключевого слова до последнего. Счетчик считает число включений каждого ключевого слова в окно параграфа. Например, поисковый запрос содержит следующий набор ключевых слов: {к1, к2, к3}. Текстовый документ разделен на параграфы, в одном из которых содержатся все ключевые слова и наибольшее число их включений в окно параграфа, к примеру, к1 и к2 встречаются 1-2 раза, а к3 – один раз. В нашем случае можно получить четыре вариации окна параграфа:

[к1-1, к2-1, к3], [к1-2, к2-1, к3], [к1-1, к2-2, к3], [к1-2, к2-2, к3].

Окно параграфа оценивается по следующим параметрам:

1. Same_word_sequence_score – число ключевых слов поискового запроса, которые встречаются в окне параграфа в том же порядке, как в поисковом запросе;

2. Distance_score – число слов текста, которые разделяют первое ключевое слово и последнее в окне параграфа;

3. Missing_score – число ключевых слов поискового запроса, которые не встречаются в окне параграфа.

В итоге фрагмент документа выбирается посредством сравнения всех параметров каждого окна параграфа и сортировки.

Для поиска информации в некоторых ВОС используется булевский поиск [6]. В таком типе поиска поисковый запрос не только из слов, но и из

булевых операций «AND/OR/NO», с помощью которых происходит поиск слов в тексте. Такой тип поиска был использован в системе Falcon [6][23], которая продемонстрировала хороший результат на конференции TREC 2001 [16]. В Falcon использовалась поисковой модуль булевского типа SMART. Главным преимуществом такого типа поиска является структура формирования запроса. В случае если результат поиска не соответствует поисковому запросу или вовсе нет ответа, то запрос модифицируется, формируется и подается на вход заново, что называется циклическим поиском.

Другие системы используют поиск по индексированной коллекции вместо стандартного поиска по ключевым словам (п. 1.3.2). Коллекция текста заранее отмечается метками такими, как именованные сущности и синтаксические связи и индексируется по ним, что значительно ускоряет поиск.

Многие современные ВОС в качестве данного модуля используют различные ПС (п. 1.3.1). ИП для ВОС состоит в получении списка релевантных ссылок и сниппетов. Обычно сниппеты содержат в себе контекст, в котором встречаются ключевые слова в тексте веб-страницы. Проанализировав сниппет, можно понять, соответствует ли страница запросу пользователя без открытия самой веб-страницы. Использование данного модуля удобно тем, что разработка нового решения ИП не требуется.

Многие ВОС английского языка для поиска информации используют словари-тезаурусы, наиболее известным из которых является WordNet. WordNet – это мощная лексическая БД, результатом работы которой является справка по слову в удобном структурированном виде для компьютерного анализа. При этом словарные статьи системы на английском языке предназначены для чтения человеком. Сейчас данный словарь переводится на русский язык, в дальнейшем будет адаптирован для использования в решении прикладных задач.

1.4.3 Извлечение ответа

На этом этапе извлекается ответ на вопрос из полученных релевантных текстовых фрагментов. Извлечение ответов происходит следующим образом: выявляются кандидаты для ответа (с английского «answer candidate») – отдельные слова или словосочетания, рассматривающийся как ответ на вопрос. Все кандидаты для ответа анализируются, оцениваются по определенным параметрам и выбирается кандидат, имеющий наивысшую оценку. В извлечении ответа важную роль имеет тип ответа. Подробнее о методах выбора кандидатов ответа описано в п. 2.2.

1.4.4 Оценка и интерпретация ответа

Данный этап является дополнительным, на этом этапе проводится оценка полученного ответа и его интерпретация. Важно убедиться, что ответ корректен и соответствует заданным требованиям. Если необходимо, можно провести дополнительный анализ ответа для более глубокого понимания темы.

1.4.5 Предоставление ответа

Данный этап является дополнительным, после того, как ответ был получен и оценен, необходимо предоставить его тому, кто задал вопрос. Ответ может быть представлен в различных форматах, таких как текст, электронное письмо или устное сообщение.

1.4.6 Обратная связь и дополнительные вопросы

Данный этап является дополнительным, на этом этапе можно получить обратную связь от того, кто задал вопрос, и ответить на дополнительные вопросы, если они возникли. Важно убедиться, что получатель понял ответ и удовлетворен им. Если есть какие-то недопонимания или вопросы, необходимо провести дополнительный анализ и поиск информации для того, чтобы предоставить более точный ответ.

1.5 Выводы

В данной главе было проведено исследование и были проанализированы существующие типы ВОС, детально рассмотрен каждый тип ВОС и построены схемы работы каждого типа. Также был произведен анализ существующих ВОС. В приведенной главе была построена общая схема работы ВОС и детально исследован каждый этап работы таких систем.

2 МЕТОДЫ ОБРАБОТКИ ЕСТЕСТВЕННОГО ЯЗЫКА И ВЫДЕЛЕНИЯ КАНДИДАТОВ ОТВЕТА

2.1 Методы анализа текста

2.1.1 Семантический анализ

Данный метод, который позволяет понимать смысл слов и фраз в контексте запроса пользователя. Например, если пользователь задает вопрос "Какое самое высокое здание в мире?", система может использовать семантический анализ, чтобы понять, что он ищет информацию о зданиях, которые имеют самую большую высоту [17].

2.1.2 Морфологический анализ текста

Метод, который используется при разработке ВОС для анализа текста. Он позволяет определить грамматическую структуру слов и выделить основные части речи, такие как существительные, глаголы, прилагательные и т.д.

В процессе морфологического анализа текста система разбивает текст на отдельные слова и проводит их лемматизацию – приведение к нормальной форме. Это позволяет учитывать различные формы слов при анализе запроса пользователя.

Морфологический анализ также может использоваться для определения синтаксической структуры предложения, что позволяет системе понимать, какие слова являются субъектом, объектом или дополнением в предложении.

В целом, морфологический анализ текста является важным методом при разработке ВОС, поскольку он позволяет системе понимать грамматическую структуру запроса пользователя и выделять ключевые слова для поиска наиболее точного ответа.

2.1.3 Синтаксический анализ текста

Метод, который используется при разработке ВОС для анализа текста. Он позволяет определить синтаксическую структуру предложения и выделить его основные элементы, такие как подлежащее, сказуемое, дополнение и т.д.

В процессе синтаксического анализа текста, система проводит глубокий анализ предложений и определяет связи между словами. Это позволяет учитывать контекст запроса пользователя и понимать его смысл.

Синтаксический анализ также может использоваться для определения семантической структуры предложения, что позволяет системе понимать, какие слова являются ключевыми для ответа на запрос пользователя.

В целом, синтаксический анализ текста является важным методом при разработке ВОС, поскольку он позволяет системе понимать смысл запроса пользователя и выделять ключевые слова для поиска наиболее точного ответа.

2.2 Выбор кандидатов ответа

2.2.1 Выделение именованных сущностей

Тип ожидаемого ответа определяет стратегию выявления ответа из фрагмента текста. Обычно для вопросов о фактах во время первичного анализа фрагментов текста применяется метод выделения именованных сущностей. К примеру, для ожидаемых ответов таких, как имена личностей, локация, название географического места (примеры классов ответов рассмотрены в таблице 1 на этапе извлечения ответа будут применены методы выявления имён собственных. В случае, если необходимо извлечь сущности может быть использована готовая система, которая извлекает информацию и обучается подобранной и обработанной по каким-либо

правилам совокупностью текстов, а также в которой могут использоваться словари, которые содержат списки или наборы различных сущностей.

2.2.2 Использование шаблонов

Соответствие шаблонам – ещё один популярный способ извлечения ответа из текста (с английского «pattern matching»). Механизм работы такого способа следующий: создаются шаблоны для каждого типа ожидаемого ответа, посредством которых в фрагментах текстов происходит поиск и извлечение кандидатов ответа. Для выбора определенного ответа применяется информация о классе ожидаемого ответа, которая была получена на этапе обработки и анализа вопроса, рассмотренном в п. 1.4.1, и символьные шаблоны.

Для создания шаблонов для ответа модно использовать как ручной метод, так и автоматическое создание посредством работы обучаемых алгоритмов. Для определения необходимого шаблона можно использовать автоматические методы определения типа ожидаемого ответа. Алгоритмы обучаются с целью выявления и построения связи между ожидаемым типом ответа, например, Дата рождения, и определенным объектом концентрации вопроса, например, какая-либо личность. Следовательно, посредством обучения создается шаблон, который связывает два типа этих фраз, например, <Дата рождения/Личность>. Рассмотрим ориентировочный алгоритм обучения для определения шаблона:

– создание списка из правильных пар для формирования связи между двумя сущностями;

– произведение запроса поисковому модулю из составляющих этих пар;

– выборка предложений из наиболее релевантных документов, которые содержат обе составляющие пар;

– выявление шаблона, содержащего слова и пунктуацию между этими составляющими пар;

– оценка и выбор шаблона.

Рассмотрим этап оценки и выбора шаблона. К примеру, данный этап можно выполнить следующим образом:

– составление запроса поисковому модулю из ключевых слов, которые входят в вопрос и подходят по паре, например, запрос – Дата каждения какой-либо Личности;

– выявление фрагментов текста и к ним происходит применение походящего шаблона. В связи с тем, что правильный ответ уже найден, подбираются шаблоны, имеющие высокий процент точности извлеченных ответов.

Примеры шаблонов для ответов, подходящих запросу к поисковому модулю, заданному выше:

<ИМЯ> (<ДАТА_РОЖДЕНИЯ>)
<ИМЯ> родился/лась <ДАТА_РОЖДЕНИЯ>

2.2.3 Использование N-грамм

Еще одним выбора кандидатов ответов из текстовых фрагментов является извлечение их использованием n-грамм. N-грамма – это последовательность, состоящая из n элементов, которые следуют друг за другом. Такой алгоритм результативно использовать в применении к сниппетам, найденным после поискового запроса, который формируется после этапа обработки и анализа вопроса. Алгоритм данного способа можно разделить на два этапа:

– сначала из сниппета извлекаются униграммы, биграммы и триграммы. Затем таким n-граммам присваиваются веса, которые равны числу сниппетов, в которых использовалась такая n-грамма;

– оценка и выбор кандидатов из n-грамм. Целью оценивания является определение того, как точно определенная n-грамма соответствует классу ожидаемого ответа. Затем n-граммы ранжируются и извлекается определенное число их с высокими оценками, после чего формируется кандидат ответа, посредством объединения n-грамм. Кандидат ответа, имеющий наиболее высокую оценку выбирается в качестве итогового ответа.

2.3 Оценка кандидатов ответа

За этапом выбора кандидатов ответа следует этап оценки и выбор предполагаемого ответа. Оценка происходит посредством проверки (с английского «answer candidate proofing») разными способами. Как было определено ранее, для каждого класса ответа создаются шаблоны, посредством которых в фрагментах текстов происходит поиск и определение кандидатов ответа. Существует множество различных методов, которые могут применяться для решения задачи извлечения ответа из кандидатов, в основном все подходы различаются тем, как обрабатываются и сравниваются в них предложения на ЕЯ для оценки и извлечения предполагаемых ответов среди выявленных кандидатов ответа.

2.3.1 Метод мешка слов

Метод мешка слов является одним из простых и самых распространённых методов вычисления оценки для кандидата ответа:

$$S_k = \frac{|Q_k \cap A_k|}{|Q_k|},$$

где Q_k – множество слов запроса, A_k – множество слов сниппета, который содержит кандидата ответа, S_k – оценка кандидата.

S_k для каждого k-го кандидата рассчитывается, как отношение пересечения множеств Q_k и A_k и множества Q_k. В исследуемой работе [20] применили похожую, но жестче отсеивающую оценку вариации метода мешка слов (q_i – i-ое слово в запросе, a_j – j-ое слово в сниппете):

$$s = \sum_{i<|Q|,j<|A|}^{|Q|} \frac{match(q_i, a_j)}{|Q|},$$

match(q_i, a_j) = *1*, если $q_i = a_j$, *0* – иначе.

2.3.2 Метод с использованием грамматик зависимостей предложений

Еще одном способом оценки потенциального кандидата ответа является метод, который использует грамматики зависимостей предложений (с английского «dependency grammars»). В исследуемой работе [19] используется оценка с помощью сравнения множества грамматик зависимостей в виде деревьев. Пример такого дерева представлен на рисунке 12. Для вопроса и найденного предложения в сниппете, который содержит кандидата ответа, формируются деревья грамматик зависимостей, которые отражают связи между словами и отражают их как части речи. Затем они сравниваются, и рассчитывается для всех кандидатов оценка неточного сравнения так называемых деревьев грамматик и извлекаются кандидаты с меньшей оценкой:

$$s = \arg\min_{a_i \in A} DR(q, a_i),$$

где A – множество предложений с кандидатами ответа, q – дерево вопроса, a_i – предложение, содержащее кандидат ответа, DR – оценка неточного сравнения деревьев.

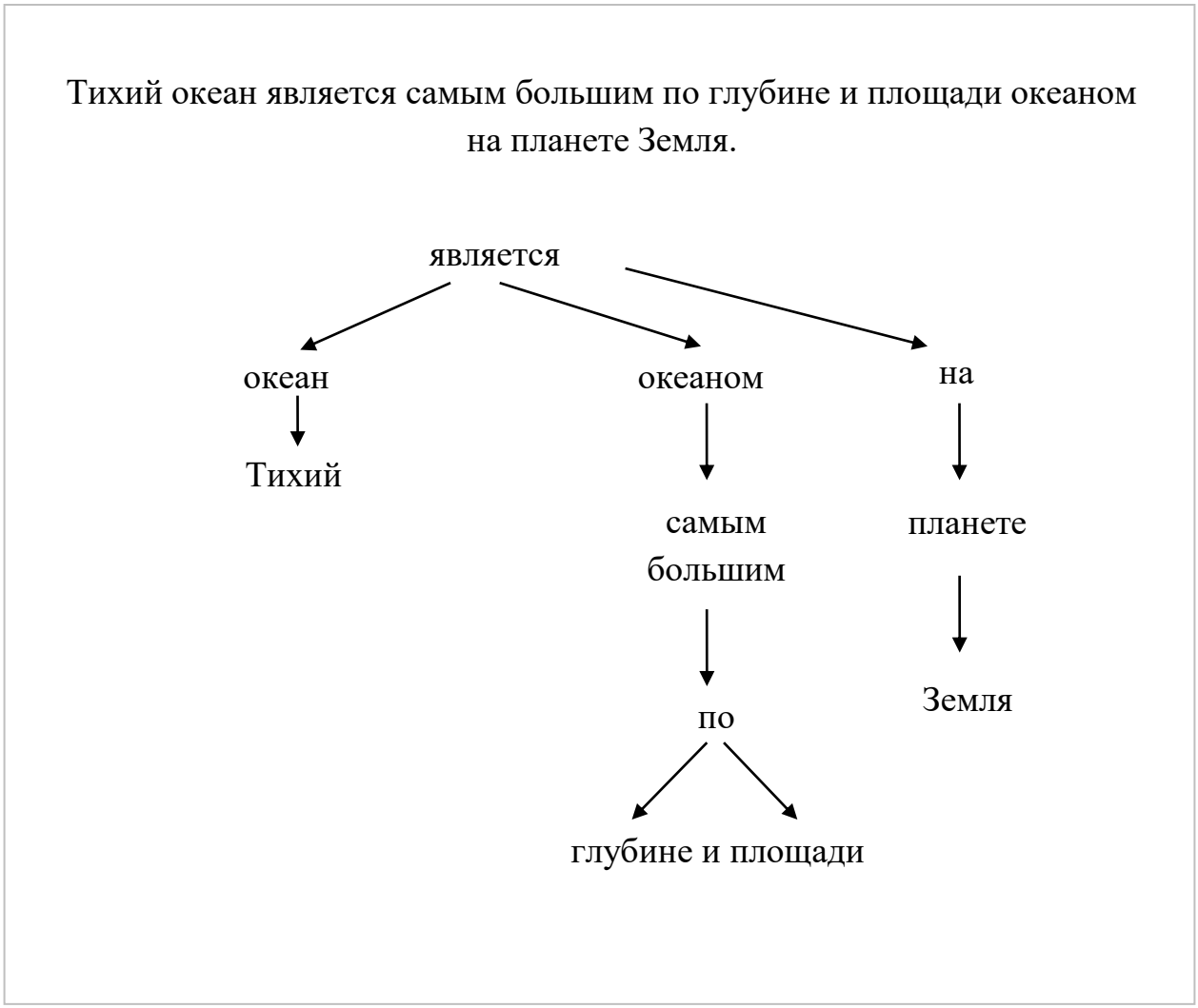

Рисунок 11 – Дерево грамматик зависимостей для предложения «Тихий океан является самым большим по глубине и площади океаном на планете Земля»

Метод оценки сложности преобразования одного в другое применятся для неточного сравнения деревьев зависимостей. Для этого вводятся три функции для модификации дерева: изменение вершины, вставка вершины, удаление вершины. Каждая такая функция обозначатся – s_i и имеет некую стоимость, обозначаемую – $\gamma(s_i)$. Основная задача состоит в том, чтобы найти последовательность операций: $S = <s_1, s_2, \ldots, s_n>$, модифицируемое дерево предложения, обозначенного – T_1 из сниппета в дерево вопроса, обозначенного – T_2 и имеющее меньшую суммарную стоимость:

$$\gamma(s) = \sum_{i=1}^{n} \gamma(s_i)$$

Исходя из этого, для двух деревьев вычисляется оценка минимального изменения:

$$\delta(T_1, T_2) = \min(\gamma(s) | S(T_1) = T_2),$$

где минимум ищется по всем S.

Авторы работы [19] провели оценку этого метода на около 450 парах вопросительных предложений и ответов и сравнили свой разработанный метод с методом мешка слов, результаты работы которых приведены в таблице 2. Результаты представлены в процентном отношении точно проанализированных и обработанных пар к всему числу тестируемых данных.

Таблица 2 – Результаты сравнения рассмотренных методов извлечения ответов

Метод	Корректно отработанных вопросов, %
Метод сравнения грамматик зависимостей	40
Метод мешка слов	33

2.3.3 Метод с использованием семантических структур

Другим методом оценки кандидата ответа является обработка текстовой информации, которая связана с определением семантических ролей (с английского «semantic role labeling») [13]. В целом, задача данного метода является следующей: для определенного предложения на ЕЯ необходимо обозначить всех участников какой-либо описанной в предложении ситуации и их семантические роли, которые соответствуют отношениям между участниками. В итоге создается семантическая структура, являющаяся структурированным представлением текстовой информации, имеющее вид ориентированного графа, используемого для оценки кандидата ответа. Для оценки происходит сравнение графа вопроса и графа предложения, содержащее кандидат ответа. Оценка кандидата ответа вычисляется по схожести этих двух графов.

Решение с использованием данного метода для русского языка является технология семантического анализа проекта АОТ [18]. Выявление семантической структуры предложения подразумевается под семантическим анализом. Рассмотрим терминологию, описанную в системе АОТ:

Семантическая структура содержит в себе семантические узлы (участники ситуации) и семантические отношения (семантические роли). Для начала происходит анализ предложения, после чего выявляется набор узлов и отношений между ними. Узлы – это слова или словосочетания, выделенные из предложения. Отношения – это связи с отметками, которые обозначают различные семантические роли. В итоге выявленные узлы и отношения можно представить как ориентированное дерево-граф, называемый авторами АОТ семантическим графом предложения. На рисунке 13 визуализирован пример, демонстрирующий разбор предложения в виде семантического графа.

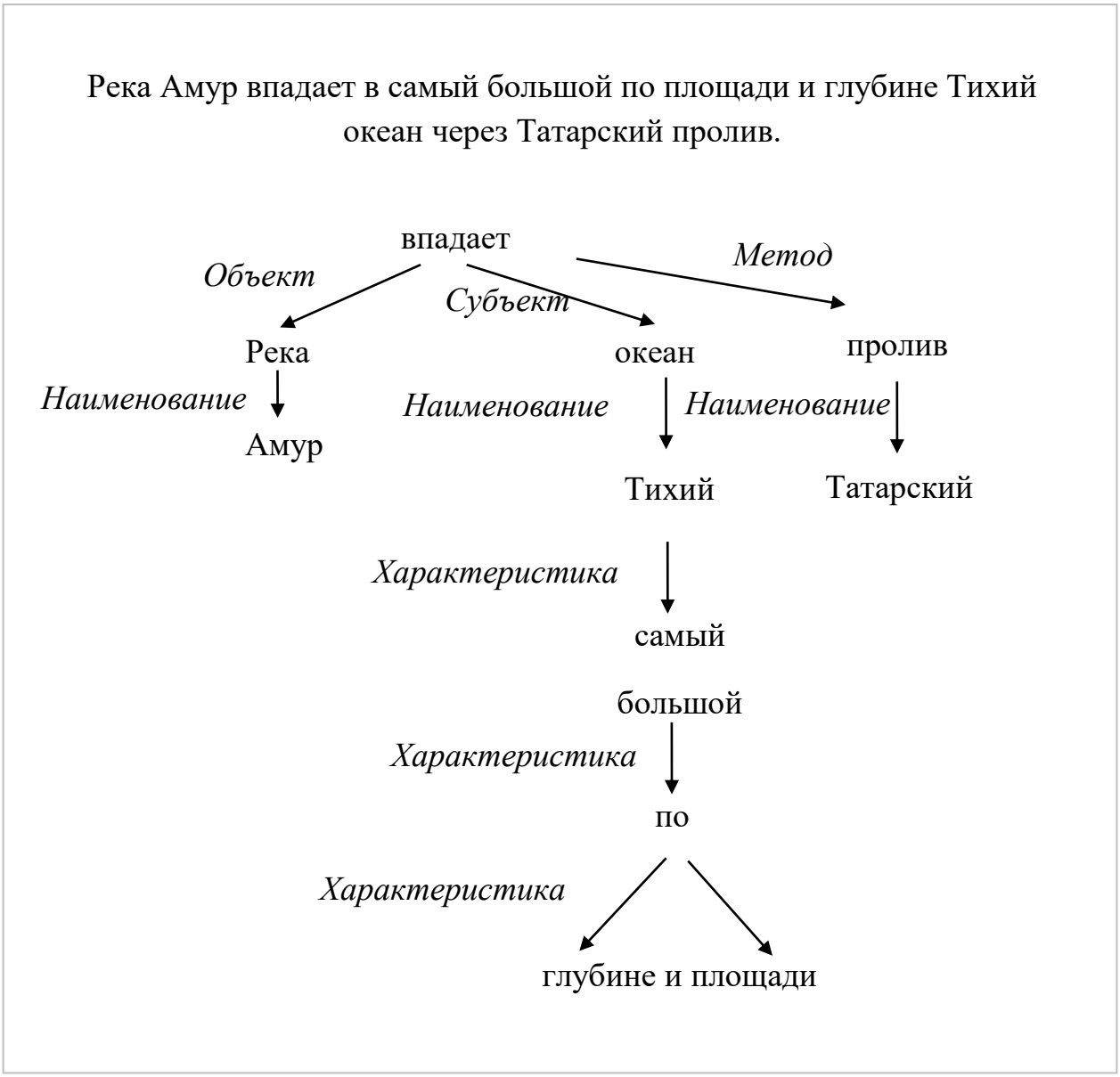

Рисунок 12 – Семантический граф для предложения «Река Амур впадает в самый большой по площади и глубине Тихий океан через Татарский пролив»

2.4 Выводы

В данной главе были исследованы методы обработки и анализа текста, методы выбора кандидатов ответов, методы оценки кандидатов ответов. Для оценки кандидатов ответа приведены формулы и детально рассмотрены примеры для каждого метода оценки кандидатов.

3 КРИТЕРИИ ОЦЕНКИ КАЧЕСТВА РАБОТЫ ВОПРОСНО-ОТВЕТНЫХ СИСТЕМ

Оценка качества работы ВОС проводится на основе экспериментальных исследований. Для этого подбирается специальный набор тестов с готовыми вопросами и ответами. На входе подают вопросы тестовой подборки, а затем проверяют корректность ответа, полученного на выходе.

Критерии оценки качества работы вопросно-ответной системы могут быть различными, но обычно они включают следующие аспекты:

3.1 Точность ответов

Критерий является основным. Он оценивает, насколько точными и полными являются ответы, которые дает ВОС на поставленные вопросы.

Точность ответов вычисляется по следующей формуле:

$$Precision = \frac{\text{Корректно отвеченные вопросы}}{\text{Отвеченные вопросы}}.$$

3.2 Полнота системы

Критерий является основным. Он оценивает, какой процент корректный ответов на вопросы дает система в отношении к общему числу заданных вопросов ВОС.

Полнота системы вычисляется по формуле:

$$Recall = \frac{\text{Корректно отвеченные вопросы}}{\text{Всего вопросов}}.$$

3.3F-мера

F-мера системы является основным критерием. Она является гармоническим средним между полнотой и точностью и вычисляется по формуле:

$$F = \frac{2 * Precision * Recall}{Precision + Recall}.$$

3.4Скорость ответов

Этот критерий оценивает, насколько быстро BOC может давать ответы на поставленные вопросы.

3.5 Понятность ответов

Этот критерий оценивает, насколько понятными и доступными являются ответы, которые дает BOC на поставленные вопросы.

3.6 Интерактивность

Этот критерий оценивает, насколько хорошо BOC может взаимодействовать с пользователем, задавая уточняющие вопросы и предлагая дополнительную информацию.

3.7 Надежность и стабильность

Этот критерий оценивает, насколько надежной и стабильной является работа BOC, как часто возникают ошибки или сбои.

3.8 Адаптивность

Этот критерий оценивает, насколько хорошо BOC может адаптироваться к конкретному пользователю и его потребностям, учитывая его предпочтения и историю запросов.

3.9 Удобство использования

Этот критерий оценивает, насколько удобно и просто пользователю работать с ВОС, какие функции и возможности доступны, как легко можно найти нужную информацию.

3.10 Обучаемость

Этот критерий оценивает, насколько легко и быстро можно обучить ВОС новой информации или задачам, какие инструменты и методы используются для этого.

3.11 Выводы

В данной главе были описаны и исследованы возможные критерии оценки ВОС, а также приведены формулы расчета для оценки ВОС.

4 ПРОЕКТИРОВАНИЕ ПРОТОТИПА ВОПРОСНО-ОТВЕТНОЙ СИСТЕМЫ

В ходе исследования выявлена наиболее распространенная архитектура работы ВОС, состоящая из трех этапов:

– обработка и анализ вопроса;

– информационной поиск ответа;

– извлечение ответа.

Однако ВОС различаются реализацией этих трех этапов, а именно, видом системы и используемыми методами обработки ЕЯ.

Для проектирования прототипа ВОС было решено исследовать и использовать языковую модель для русского языка BERT.

Использование языковой модели приведено в Приложении Б.

4.1 Исследование языковой модели BERT

BERT (Bidirectional Encoder Representations from Transformers) – это двунаправленная предобученная модель, основанная на трансформерах (рис. 14). Обычно архитектура таких моделей состоит из двух частей: кодировщика (coder) и декодировщика (decoder), но в BERT используется только кодировщик, который обрабатывает входные данные и выдает их представление в виде вектора.

Кодировщик состоит из 12 слоев, каждый из которых содержит следующие подслои:

– первый подслой – многословная аттенция (механизм внимания, с английского «multi-head attention») – позволяет модели учитывать контекст и зависимости между словами. Она позволяет модели обрабатывать входные

данные более эффективно, чем другие методы. Многословная аттенция позволяет модели сосредоточиться на различных аспектах текста одновременно. Механизм внимания применяется ко всей последовательности токенов и формирует несколько векторов внимания;

– второй подслой – полносвязный слой (с английского «feed-forward-layer») – применяется к каждому элементу последовательности независимо и применяет линейное преобразование к выходу многословной аттенции.

Двунаправленность отличает модель BERT от моделей GPT (рис. 15) и ELMo (рис. 16). Все перечисленные модели состоят из Трансформер слоев и отличаются способом обработки входных данных. Эта особенность позволяет модели изучать контекст слова на основе всего его окружения (слева и справа от слова). В GPT последовательность обрабатывается слева направо для построения представлений токенов. В ELMo используются представления с двух независимых рекуррентных сетей: одна обрабатывает последовательность слева направо, вторая — справа налево. В ELMo тоже есть двунаправленность, но представления с каждого из направлений получаются независимыми.

Таким образом, архитектура BERT является очень эффективной для обработки ЕЯ и позволяет модели достигать высоких результатов на различных задачах. С помощью BERT можно разрабатывать эффективные ИИ-программы для обработки ЕЯ: отвечать на вопросы (ВОС), заданные в произвольной форме, создавать чат-ботов, выполнять автоматические переводы, анализировать текст и др.

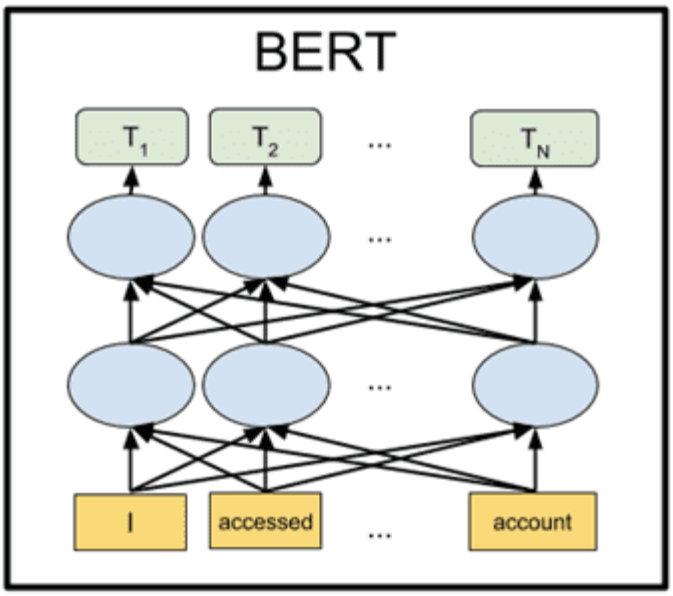

Рисунок 13 – Архитектура BERT

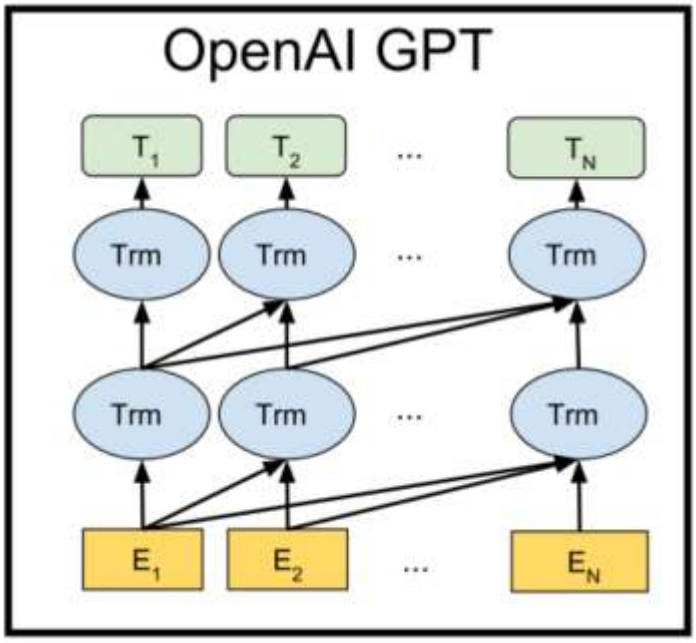

Рисунок 14 – Архитектура GPT

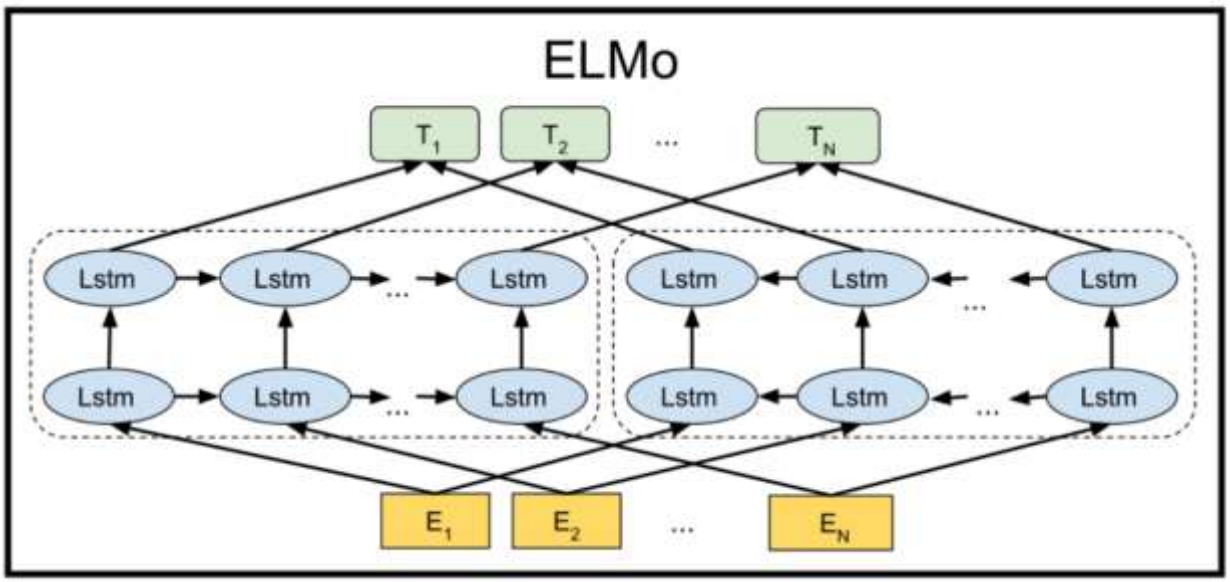

Рисунок 15 – Архитектура ELMo

Проектируемый прототип будет работать следующим образом: пользователь вносит текст неопределенного размера, в котором может находиться ожидаемый ответ, а также задает вопрос системе на ЕЯ. Текст и вопрос передаются BERT. Модель анализирует вопрос и разбивает его на ключевые слова и фразы для поиска в заданном тексте, извлекает из него ответ и передает пользователю.

В проектируемом прототипе используется модель RuBERT, которая была предобучена в рамках проекта DeepPavlov [21] на русскоязычной Википедии и новостных данных, и разработчики могут скачать и встроить в свои проекты по обработке ЕЯ уже готовый инструмент, не тратя время на обучение НС с нуля [32]. Модель RuBERT состоит из 12-слойного Трансформера, 768 подслоев, 12 голов внимания и 180 миллионов параметров.

4.2 Алгоритм обработки текста и обучение BERT

Любая языковая модель работает исключительно с текстом, но компьютер распознает только числа. Для распознавания текста компьютером используется деление текста на токены и их кодирование.

Токеном может являться буква, часть слова (слог, окончание) или целое слово. Обычно, берутся самые часто встречающиеся слова или указывающие на часть речи слоги. Способ кодировки токена называется «эмбеддинг» (с английского «to embed — встраивать»), так как его помещают в числовое пространство. Для кодировки токенов используются специальные алгоритмы.

Итак, «эмбеддинг токена» — это числовое обозначение буквы, части слова или целого слова.

Языковая модель принимает как входные данные числовые обозначения токенов и сообщает результат, зависящий от задачи. Существуют стандартные наборы задач, которые необходимо выполнить на стандартных наборах данных для того, чтобы НС справлялась с поставленной задачей. BERT тестировали на различных наборах стандартных задач, например, на SQuAD [21] (набор для ВОП) [12].

С помощью обучения НС удобно заполнять и корректировать нужные «матрицы весов». Веса представляются в виде чисел, влияющих на важность данных для системы. Данные обрабатываются и умножаются на их вес, в соответствии с их важностью.

Обычно, код пишется единожды и никем не может быть отредактирован, но НС при выполнении задачи адаптирует часть своего кода, а точнее – изменяет веса из матриц. НС не создает другой алгоритм для себя, но для корректного выполнения задачи может подобрать числа в своем алгоритме.

BERT [11] предобучался на «маскированной языковой модели». Суть такой модели состоит в том, что нужно определить или предсказать слово, которое находится не в конце предложения. Модель называется маскированной, так как необходимый токен заменяется на токен [MASK].

Также BERT [11] обучалась на задаче предсказания следующего предложения (с английского «next sentence prediction»), состоящей в том, что BERT обучался определению того, являлись ли два предложения или фрагмента текста следующими подряд в тексте. С помощью данной задачи BERT строит зависимости между предложениями или фрагментами текста.

BERT использует такие токены: [MASK] —маскирование токена, [SEP] –обозначение конца предложений/фрагментов текста, [CLS] – специальный токен, использующийся для предсказания следующего предложения и классификации. Текст разбивается следующим образом: входной текст делится на токены, далее в начало последовательности добавляется [CLS], потом вставляются токены первого текста, далее [SEP], потом токены второго текста и в конце текста [SEP].

Входные представления содержат три компоненты:

– Векторное представление токена (с английского «token embeddings») — соответствие вектора каждому токену.

– Векторное представление сегмента (с английского «segment embeddings») — два вектора представляют два фрагмента текста. Это позволяет указать BERT принадлежность токена к предложению.

– Векторное представление позиции в тексте (с анлийского «position embeddings») — это вектор, который кодирует информацию о позиции токена в последовательности.

Итак, итоговый вектор – это сумма из трех векторов, которые перечислены выше. Каждый тип векторов проходит обучение совместно с моделью BERT.

Для вопросно-ответных систем существует предобученная модель на наборе данных SQuAD. Алгоритм работы такой модели следующий: вопрос подается в качестве первого фрагмента текста, а текст подается как второй фрагмент. Фрагменты разделяются специальным токеном [SEP], а в начало последовательности добавляется токен [CLS]. Если текст не имеет ответ на вопрос, то модель выбирает токен [CLS], как правильный ответ.

4.3 Выводы

В данной главе исследован механизм разрабатываемого прототипа ВОС. Рассмотрев модель BERT, можно сделать вывод, что она является одной из лучших и перспективных языковых моделей. Существующие предобученные системы на основе BERT являются готовым инструментом для создания ВОС. В разрабатываемом прототипе использована модель RuBERT, которая была предобучена и тестирована на стандартных задачах SQuAD.

5 РЕАЛИЗАЦИЯ РАБОЧЕГО ПРОТОТИПА

Для разработки рабочего прототипа было решено использовать язык программирования Python.

5.1 Язык программирования

Python является одним из наиболее популярных языков программирования и удобен для разработки ВОС. Это связано с рядом преимуществ, которые предоставляет Python:

1. Простота и удобство использования. Python имеет простой и понятный синтаксис, что делает его очень удобным для программистов и позволяет быстро создавать прототипы.

2. Большое количество библиотек и фреймворков. Python имеет широкий выбор библиотек и фреймворков, которые помогают ускорить разработку и облегчить решение задач.

3. Высокая производительность. Python является достаточно быстрым языком, что позволяет создавать эффективные ВОС.

4. Широкое распространение и поддержка сообщества. Python имеет огромное сообщество разработчиков, которые активно работают над улучшением языка, созданием новых библиотек и фреймворков, а также предоставляют поддержку и помощь другим разработчикам.

5. Поддержка многих операционных систем. Python поддерживает большое количество операционных систем, что делает его универсальным инструментом для разработки ВОС.

Таким образом, Python является отличным выбором для разработки ВОС благодаря своей простоте, широкому выбору инструментов и высокой производительности.

5.2 Фреймворк

Для реализации рабочего прототипа были рассмотрены два популярных фреймворка Django и Flask. Они являются самыми распространенными фреймворками для разработки веб-приложений на языке Python. Вот некоторые из их основных отличий:

1. Django – это полноценный фреймворк для создания крупных проектов, в то время как Flask – это более легкий и гибкий фреймворк, который подходит для создания небольших приложений.

2. Django имеет встроенные функции аутентификации, авторизации и администрирования, что делает его очень удобным для создания сложных приложений с большим количеством пользователей. Flask предоставляет несколько расширений для реализации аутентификации, таких как Flask-Login и Flask-Security. Эти расширения облегчают создание системы аутентификации и авторизации в Flask-приложениях.

3. Django имеет строгую структуру проекта, что может быть полезным для командной разработки. Flask не навязывает такую структуру и позволяет разработчикам организовывать свой проект по своему усмотрению.

4. Flask имеет более простой и понятный синтаксис, что делает его более доступным для начинающих программистов. Он также имеет меньший размер и меньше зависимостей, что делает его более легковесным и быстрым.

Таким образом, был выбран фреймворк Flask, в связи с тем, что он является отличным выбором для создания небольших и гибких веб-приложений с простым синтаксисом.

5.3 База данных

Для объектно-реляционного отображения в разрабатываемом прототипе была применена библиотека SQLAlchemy.

SQLAlchemy – это библиотека для работы с БД в языке программирования Python. Она предоставляет высокоуровневый API для работы с различными типами БД, включая PostgreSQL, MySQL, SQLite и др. SQLAlchemy позволяет создавать объектно-ориентированные модели данных и выполнять запросы к БД с помощью языка SQL. Она также поддерживает ORM (Object-Relational Mapper – объектно-реляционное отображение), что позволяет работать с данными в виде объектов Python, а не запросов SQL. SQLAlchemy является одной из наиболее популярных библиотек для работы с БД в Python.

Одним из преимуществ SQLAlchemy является то, что код приложения будет оставаться тем же вне зависимости от используемой БД. Это позволяет с легкостью мигрировать с одной БД на другую, не переписывая код.

В разрабатываемом приложении нет необходимости хранить текст, вопросы и ответы на них. Однако в приложении добавлена функция авторизации, что выявляет необходимость в хранении учетных данных пользователей. SQLite, встроенная в Python отлично справляется с данной задачей. SQLite – компактная встраиваемая СУБД. Исходный код библиотеки передан в общественное достояние.

SQLite – это реляционная СУБД, которая хранит данные в локальном файле базы данных. Она не требует отдельного сервера БД и может быть интегрирована непосредственно в приложения. SQLite поддерживает стандартный набор SQL-команд и имеет небольшой размер, что делает ее привлекательным выбором для приложений и других проектов с ограниченными ресурсами.

SQLite также обладает высокой производительностью и поддерживает транзакции ACID для обеспечения целостности данных.

5.3 Схема функционирования

На рисунке 16 изображена схема работы реализованного прототипа ВОС.

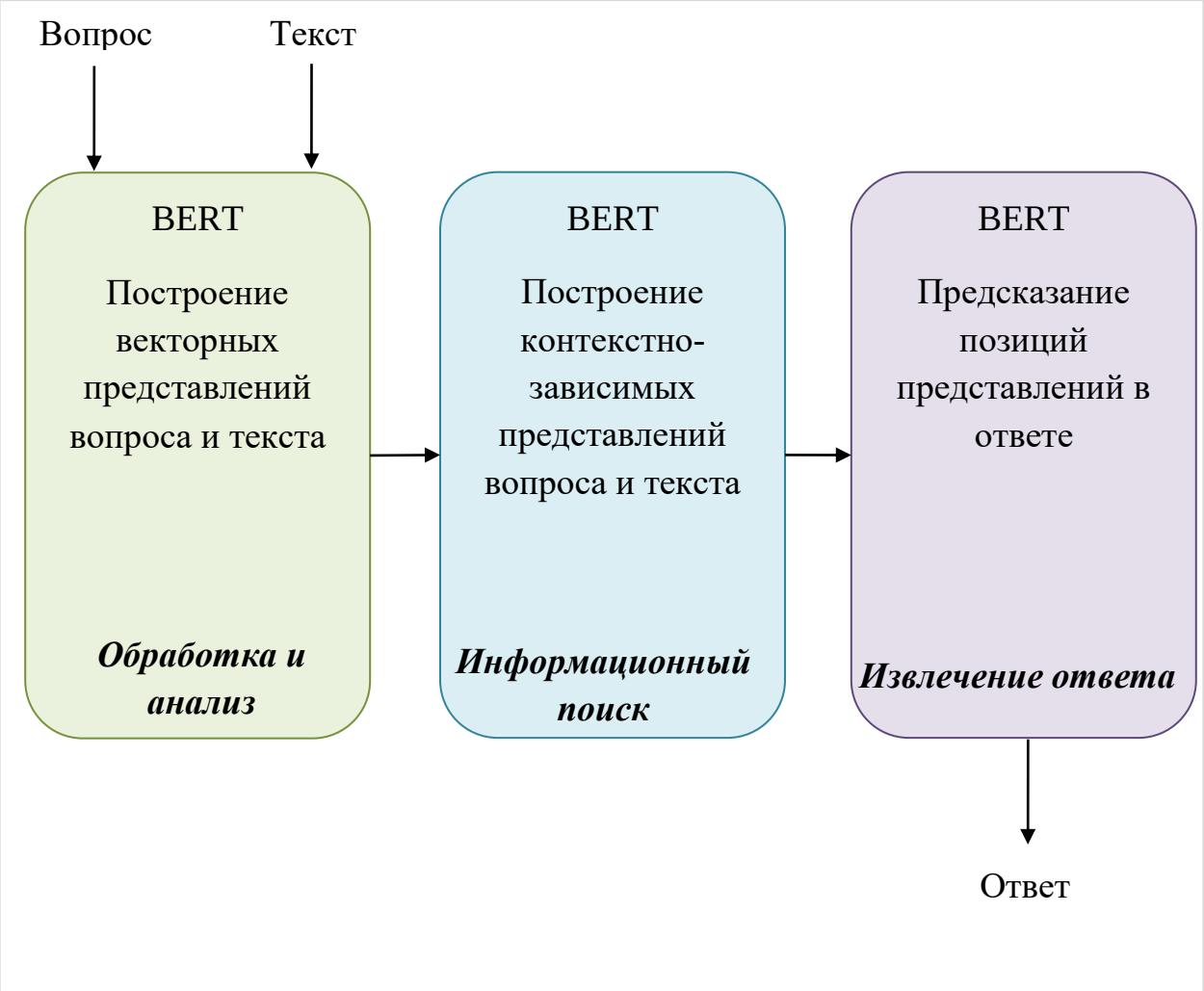

Рисунок 16 – Схема функционирования прототипа ВОС

Таким образом, настоящее приложение является относительно несложным с архитектурной точки зрения и выполняет лишь одну основную функцию: извлечение структурированной информации из заданного текста, реализуемую системой BERT.

5.4 Диаграмма классов

На рисунке 17 изображена диаграмма классов разработанного прототипа.

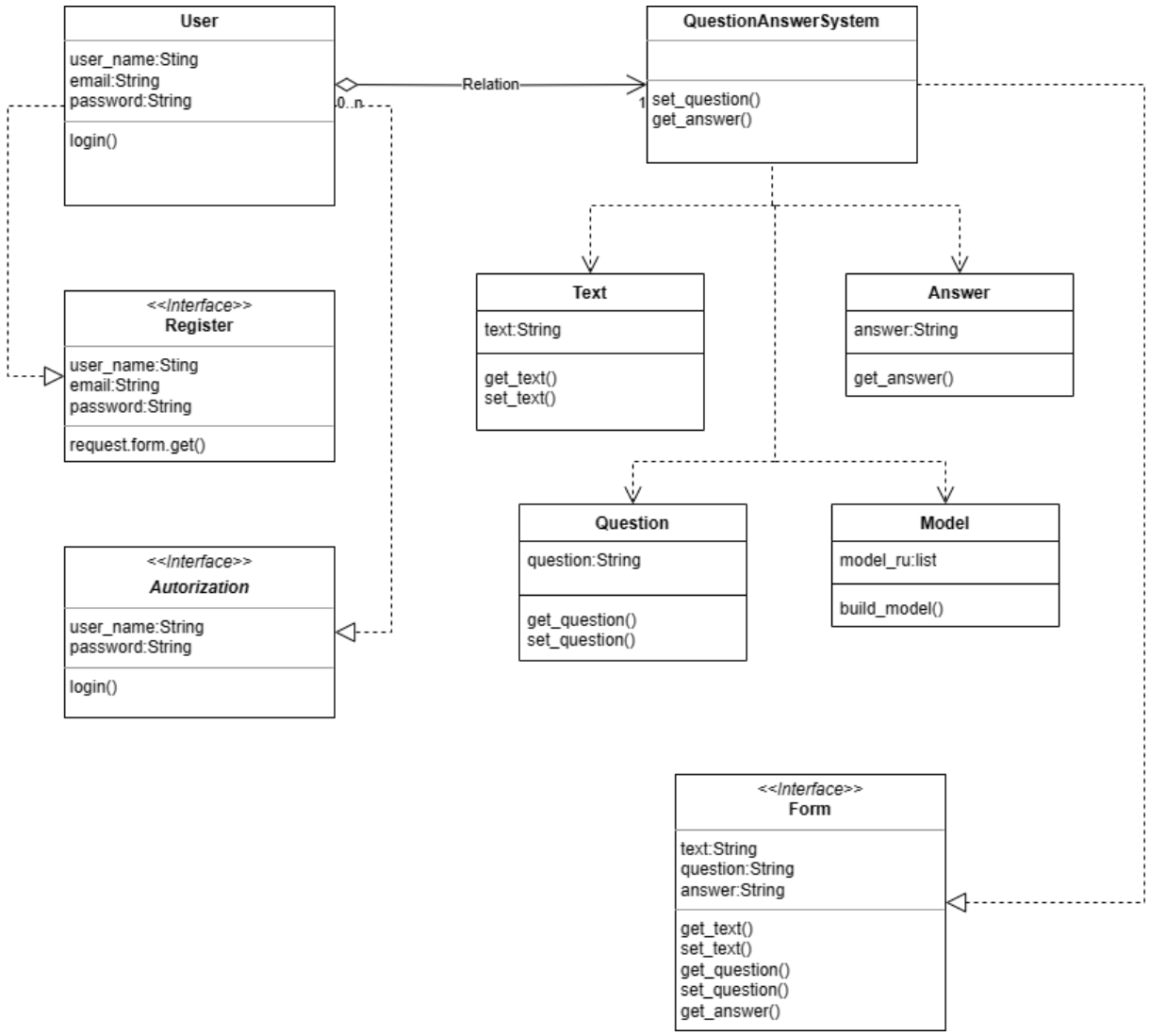

Рисунок 17 – Диаграмма классов

5.5 Графический интерфейс

На рисунке 18 изображен графический интерфейс разработанного прототипа ВОС.

Интерфейс содержит 2 поля для ввода данных, текста и вопроса соответственно. Также находится обработчик текста и кнопка для получения ответа. Интерфейс содержит окно вывода ответа.

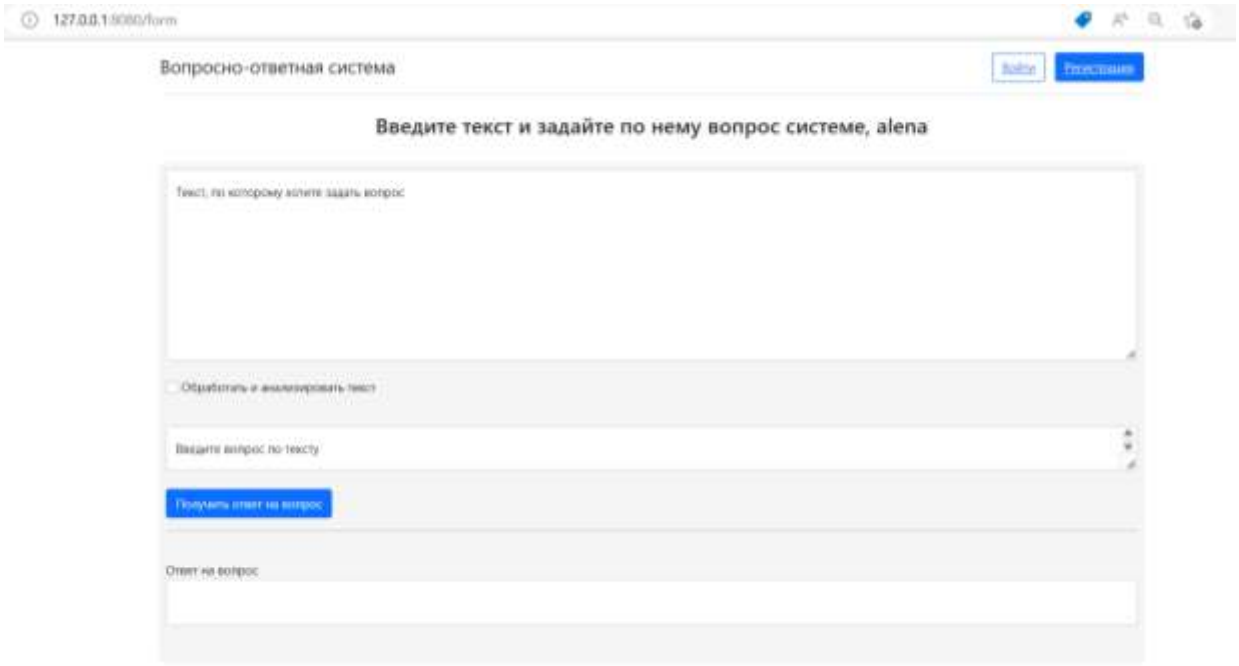

Рисунок 18 – Графический интерфейс ВОС

В приложении А приведены пять примеров работы разработанного прототипа системы.

На рисунке 19 изображен интерфейс регистрации пользователя, содержащий окна для ввода имени, электронной почты и пароль.

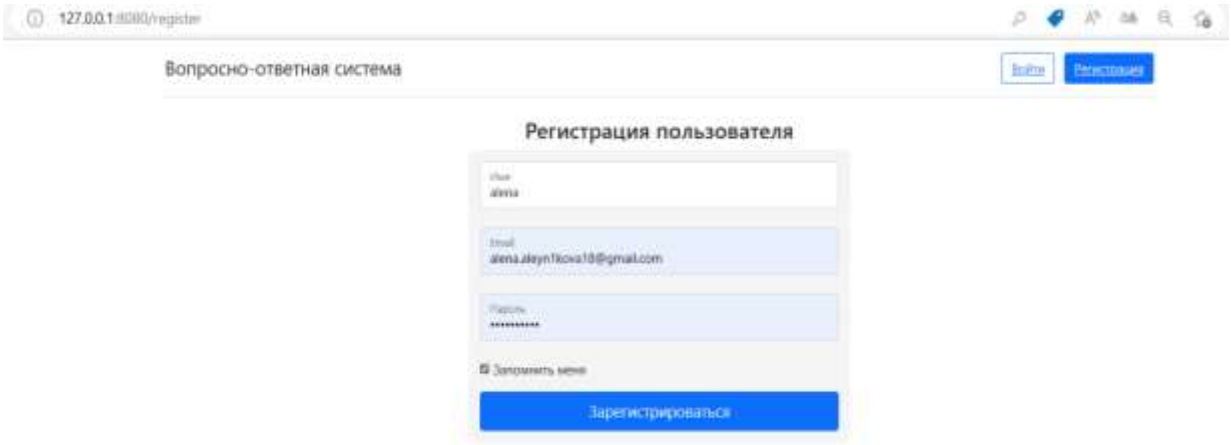

Рисунок 19 – Регистрация пользователя

5.6 Функциональные характеристики

Вычислительная машина, на которой тестировался разработанный прототип:

– процессор Intel® Core ™ i3-1115G4 3.00GHz;

– оперативная память 8 Gb.

Среднее время отклика прототипа ВОС с использованием графического интерфейса и системы BERT равно 5,21с. Расчет среднего времени работы производился с помощью 15 тестовых текстов и вопросов.

5.7 Оценка работы системы

Протестируем систему и оценим по критериям, приведенным в главе 3.

1. Точность ответов

Точность ответов вычисляется по формуле, обозначенной в п. 3.1:

$$Precision = \frac{14}{15} = 0{,}94.$$

2. Полнота системы

Полнота системы вычисляется по формуле, обозначенной в п. 3.2:

$$Recall = \frac{14}{15} = 0{,}94.$$

3. F-мера

F-мера вычисляется по формуле, обозначенной в п. 3.3:

$$F = \frac{2 * 0{,}94 * 0{,}94}{0{,}94 + 0{,}94} = 0{,}94.$$

4. Скорость ответов

Разработанный прототип ВОС достаточно быстро анализирует и обрабатывает текст и вопрос. Среднее время ответов на заданные тестовые тексты и вопросы составило 5,21с. В сравнении участвовало 15 тестовых примеров.

5. Понятность ответов

Разработанный прототип ВОС дает понятные ответы. В оценке понятности ответов участвовали 11 человек. Оценка является положительной, всем 11 людям ответы ВОС были понятны.

6. Интерактивность

Разработанный прототип ВОС не взаимодействует с пользователем и не задает дополнительные вопросы, что можно отнести к перспективам разработки.

7. Надежность и стабильность

Разработанный прототип является стабильным и надежным, так как использует в своей архитектуре модель предобученную BERT. В случае если ВОС будет выдавать неверные ответы, можно будет дообучить систему.

8. Адаптивность

Разработанный прототип не хранит историю запросов пользователя и не адаптируется под него, но имеет возможность авторизации, что позволяет в дальнейшем проработать адаптируемость ВОС.

9. Удобство использования

Разработанный прототип ВОС является удобным для пользователя. В оценке удобства участвовали 11 человек. 9 из 11 человек положительно оценили удобство работы, 2 из 11 хотели бы, чтобы ВОС отвечала на вопрос посредством веб-поиска или нахождением ответа в экспертной БД.

10. Обучаемость

Используемая в разработанном прототипе модель BERT является дообучаемой, что является удобным механизмом для дальнейшего исследования.

5.8 Выводы

В данной главе описана схема работы разработанного прототипа ВОС, описаны инструменты для разработки, приведена диаграмма классов, изображен графический интерфейс с примером работы прототипа. Также приведены функциональные характеристики и расчет среднего времени работы системы с использованием графического интерфейса. Стоит отметить, что в главе приведена оценка разработанного прототипа по критериям, описанным в главе 3. В целом, из 10 заданных критериев – 8 являются положительными, в связи с чем можно сделать вывод, что ВОС выполняет свою задачу и имеет хорошую оценку. Недоработанные критерии можно отнести в перспективы для разработки.

ЗАКЛЮЧЕНИЕ

В рамках данной работы было проведено исследование существующих типов ВОС и методов анализа текста. Был проведен анализ существующих ВОС. А также был разработан прототип ВОС для русского языка, протестирован и оценен, то есть все поставленные задачи были выполнены:

– исследованы существующие методы анализа текста;

– исследованы существующие ВОС;

– исследованы этапы работы ВОС;

– разработан прототип ВОС;

– протестирован разработанный прототип ВОС;

– оценены результаты тестирования.

В ходе исследования был разработан рабочий прототип ВОС, основанный на системе BERT для русского языка. Используемая модель RuBERT была предобучена и протестирована на стандартных задачах SQuAD. Такие модели являются готовым инструментом для разработки каких-либо приложений для обработки текстов на ЕЯ, что позволяет разработчикам не тратить время на анализ текста и разработку собственных алгоритмов. Данные модели являются дообучаемыми, что дает неограниченность действий и возможность доработки систем для определенных разработок и самостоятельного определения весов. В ходе работы модель была протестирована в разработанном рабочем прототипе и показала высокие результаты.

Перспективой разработки по теме исследования является разработка экспертной ВОС, использующей локальную БД и состоящей из четырех этапов: обработка и анализ вопроса, информационный поиск ответа, извлечение ответа и обратная связь и дополнительные вопросы. На этапе извлечения ответа планируется производить выбор кандидатов ответа с использованием N-грамм и их оценка с использованием метода мешка слов. Другой перспективой работы является детальное исследование существующих языковых моделей и методов машинного обучения этих языковых моделей, а также самостоятельное дообучение моделей. Также к перспективе разработки можно отнести проработку разработанного прототипа ВОС в части адаптивности под пользователя и интерактивности.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Vanitha Guda, Suresh Kumar Sanamrudi, I. Lakshmi Manyakamba Approaches for question answering // International Journal of Engineering Science and Technology. 2011. 3. №2. P. 990-995.

2. Gaizauskas R., Humphreys K. A Combined IR/NLP Approach to Question Answering Against Large Text Collections. Department of Computer Science. University of Sheffield, Sheffield 2010. №10.1.1.26.119.

3. Google Assistant – ваш личный помощник от Google. [Электронный ресурс] // URL: https://assistant.google.com/.

4. CIKM. The Conference on Information and Knowledge Managment. [Электронный ресурс] // URL: https://www.cikm.org/.

5. Siri – Apple (RU). [Электронный ресурс] // URL: https://www.apple.com/ru/siri/.

6. Richard J Cooper, Stefan M Rüger A Simple Question Answering System // Imperial College of Science, Technology and Medicine, 2007.

7. IBM Watson. IBM Watson. [Электронный ресурс] // URL: http://www.ibm.com/watson.

8. Блог компании Яндекс. Поиск в интернете: региональные особенности. [Электронный ресурс] // URL: https://yandex.ru/company/researches/2010/ya_regions_search_2010.

9. Lynette Hirschman, Marc Light, Eric Breck, John D. Burger Deep Read: A Reading Comprehension System // Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics. Stroudsburg, USA: Association for Computational Linguistics, 1999. P.325-333.

10. D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan Building Watson: An Overview of the DeepQA Project // AI Magazine. 2010. №3. P. 59-79.

11. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / J. Devlin, M.-W. Chang, K. Lee, K. Toutanova // Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Longand Short Papers). — 2019. — P. 4171—4186.

12. SQuAD: 100,000+ Questions for Machine Comprehension of Text / P. Rajpurkar, J. Zhang, K. Lopyrev, P. Liang // Proceedings of the 2016 Conferencen Empirical Methods in Natural Language Processing. — Austin, Texas: Association for Computational Linguistics, 11/2016. — P. 2383—2392. [Электронный ресурс] // URL: https://aclanthology.org/D16-1264/.

13. Shen Dan, Mariella Lapata Using Semantic Roles to Improve Question Answering // Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Jeju, Korea: Association for Computational Linguistics, 2007. P.12-22

14. ICTIR. International Conference on Theory on Information Retrieval. [Электронный ресурс] // URL: https://www.ictir.org.

15. Wolframalpha [Электронный ресурс] // URL: https://www.wolframalpha.com/.

16. Dan Roth, Gio Kao Kao, Xin Li, Ramya Nagarajan, Vasin Punyakanok, Nick Rizzolo, Wen-tau Yih, Cecilia Ovesdotter Aim, Liam Gerard Moran Learning Components for a Question-Answering System // TREC, 2001. P. 539-548.

17. Dongli Han, Yuhei Kato, Kazuaki Takehara, Tetsuya Yamamoto, Kazunori Sugimura, Minoru Harada QA System Metis Based on Web Searching and Semantic Graph Matching // IFIP International Federation for Information Processing. 228. 2007. P.123-133

18. AOT – Автоматическая обработка текстов. [Электронный ресурс] // URL: http://www.aot.ru.

19. Punyakanok, V., Roth, D. and Yih, W. Natural language interface via dependency tree mapping: An application to question answering // AI and Math. January 2004. P.22-34.

20. S. Quateroni, S. Manandhar Designing an Interactive Open Domain Question Answering // Natural Language Engineering. 2008. №1. P.1-23.

21. DeepPavlov: Open-Source Library. [Электронный ресурс] // URL: http://docs.deeppavlov.ai.

22. SIGIR. Special Interest Group on Information Retrieval. [Электронный ресурс] // URL: https://sigir.org.

23. Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Girju, Vasile Rus, Paul Morarescu FALCON: Boosting Knowledge for Answer Engines // Department of Science Southern Methodist University, Dallas, 2005.

24. Dan Moldovan, Sanda Harabagiu LASSO: The structure and perfomance of open domain question answering system // Southern Methodist University, Dallas, 2000.

25. Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, Chin-Yew Lin Question Answering Webclopedia // Information Sciences Institute University of Southern California.

26. ECIR. European Conference on Information Retrieval. [Электронный ресурс] // URL: https://esir.org.

27. WSDM. Web Search and Data Mining. [Электронный ресурс] // URL: https://www.wsdm-conference.org.

28. Аксенов К. А. Гибридное моделирование мультиагентных процессов преобразования ресурсов: монография / К. А. Аксенов, Н. В. Гончарова; Министерство образования и науки Российской Федерации, Уральский федеральный университет имени первого Президента России Б.Н. Ельцина. — Москва: Издательский дом «Академии Естествознания», 2019. — 222 с. — ISBN 978-5-91327-606-3. [Электронный ресурс] // URL: http://hdl.handle.net/10995/79339.

29. Antonova A. S., Aksyonov K. A., Ziomkovskaia P. E. (2022). Development of Method and Information Technology for Decision-Making, Modeling, and Processes Scheduling. В T. E. Simos, T. E. Simos, T. E. Simos, T. E. Simos, & C. Tsitouras (Ред.), International Conference on Numerical Analysis and Applied Mathematics, ICNAAM 2020 [130003] (AIP Conference Proceedings; Том 2425). American Institute of Physics Inc.. P.130003-1 – 130003 [Электронный ресурс] // URL: https://doi.org/10.1063/5.0082100 P/.

30. Antonova A., Aksyonov K., Ziomkovskaya P. (2021) Development of a Method and a Software for Decision-Making, System Modeling and Planning of Business Processes. In: Succi G., Ciancarini P., Kruglov A. (eds) Frontiers in Software Engineering. ICFSE 2021. Communications in Computer and Information Science, vol 1523. Springer, Cham. [Электронный ресурс] // URL: https://doi.org/10.1007/978-3-030-93135-3_10.

31. Aksyonov K.A., Ziomkovskaya P.E., Dougandaga Danwitch, Aksyonova O.P. and Aksyonova E.K. Development of a text analysis agent for a logistics company's Q&A system. Journal of Physics: Conference Series, Volume 2134, VIII International Young Scientists Conference "Information Technologies, Telecommunications and Control Systems" (ITTCS 2021) 16-17 December 2021, Innopolis, Russia. J. Phys.: Conf. Ser. 2134 012021. [Электронный ресурс] // URL: https://iopscience.iop.org/article/10.1088/1742-6596/2134/1/012021/pdf doi:10.1088/1742-6596/2134/1/012021.

32. Tarasiev A., Filippova M., Aksyonov K., Aksyonova O., Antonova, A. Using of Open-Source Technologies for the Design and Development of a Speech Processing System Based on Stemming Methods. IFIP Advances in Information and Communication Technology Volume 582 IFIP, 16th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2020; Innopolis; Russian Federation; 12-14 May 2020, 2020. pp.98-105. DOI: 10.1007/978-3-030-47240-5_10.

33. Aksyonov K., Aksyonova O., Antonova A., Aksyonova E., Ziomkovskaya P. Development of Cloud-Based Microservices to Decision Support System. IFIP Advances in Information and Communication Technology Volume 582 IFIP, 16th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2020; Innopolis; Russian Federation; 12-14 May 2020. 2020. pp.87-97. DOI: 10.1007/978-3-030-47240-5_9.

ПРИЛОЖЕНИЕ А. ПРИМЕРЫ РАБОТЫ СИСТЕМЫ

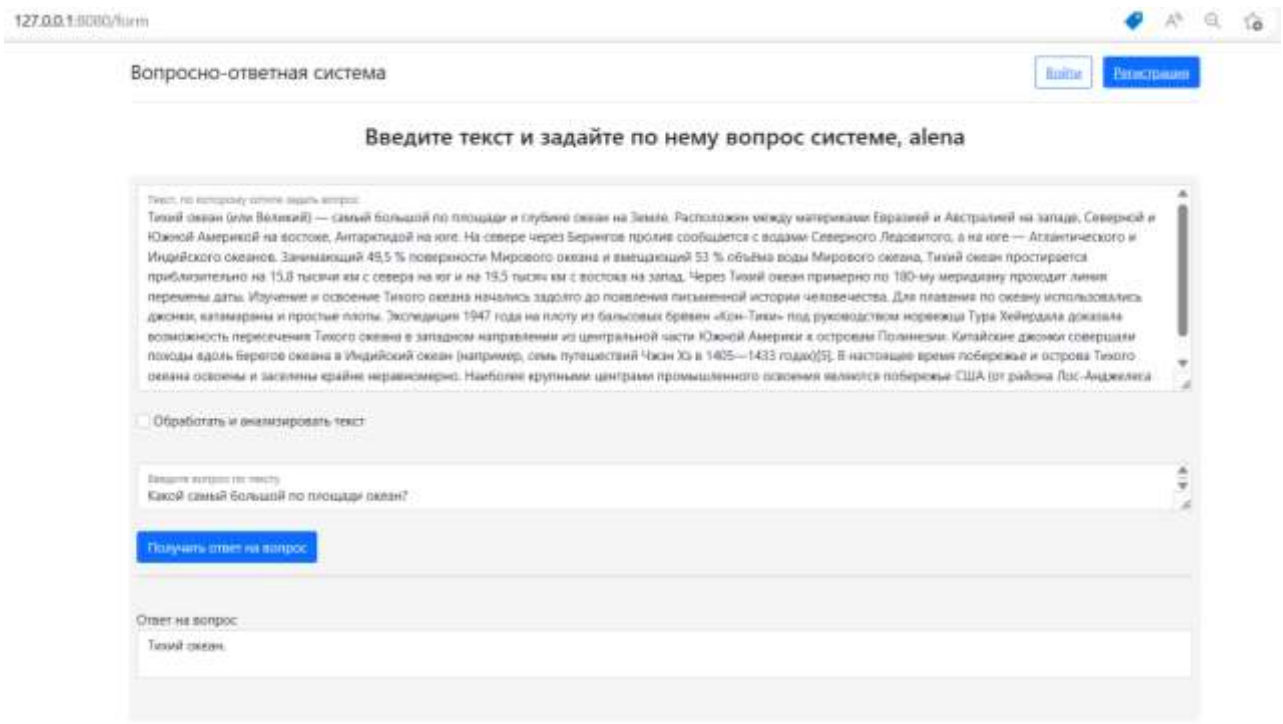

Рисунок А.1 – Пример работы системы

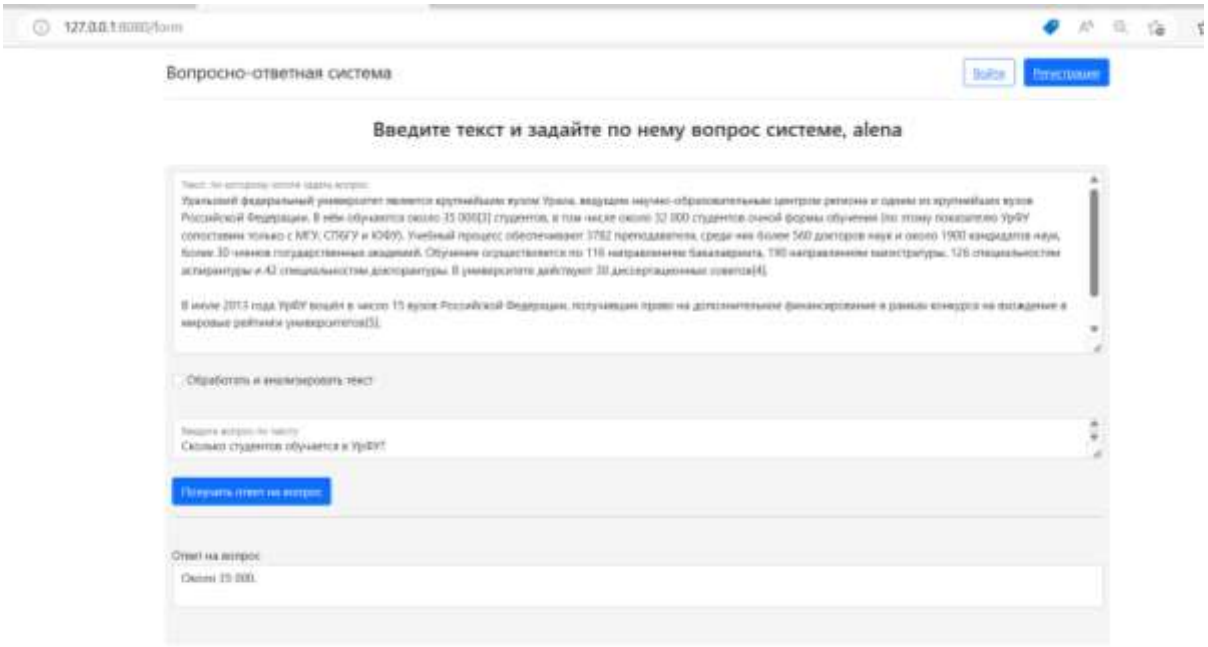

Рисунок А.2 – Пример работы системы

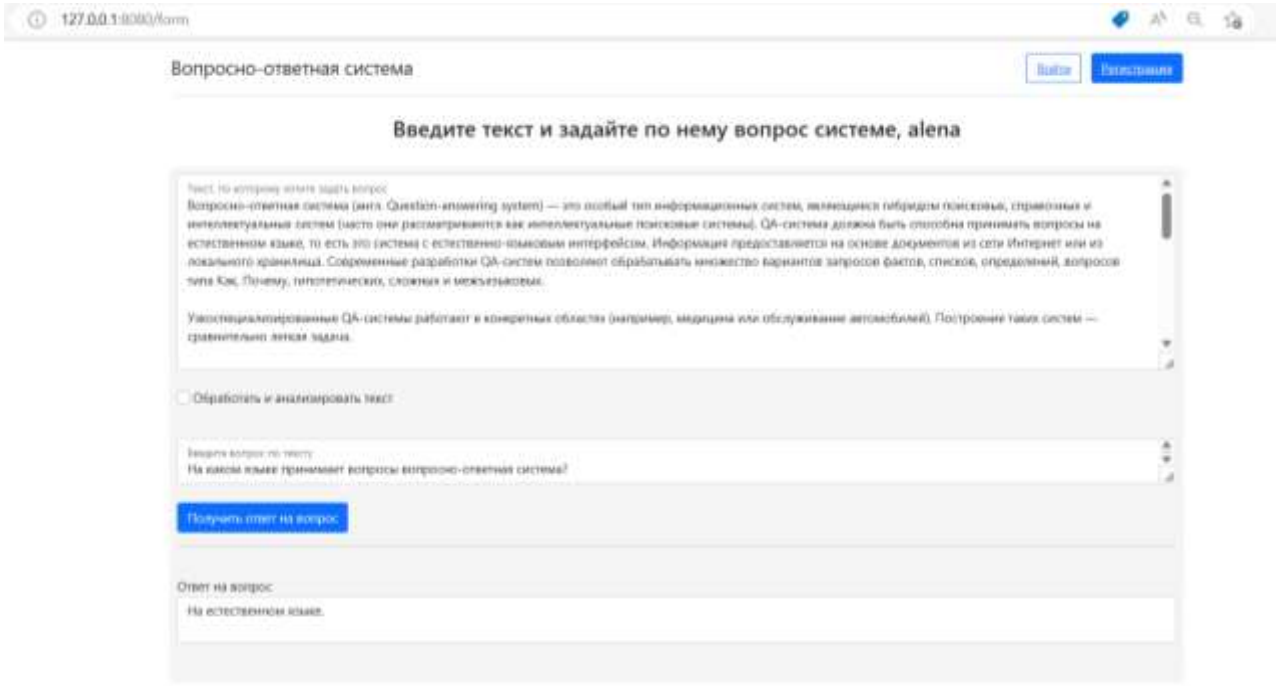

Рисунок А.3 – Пример работы системы

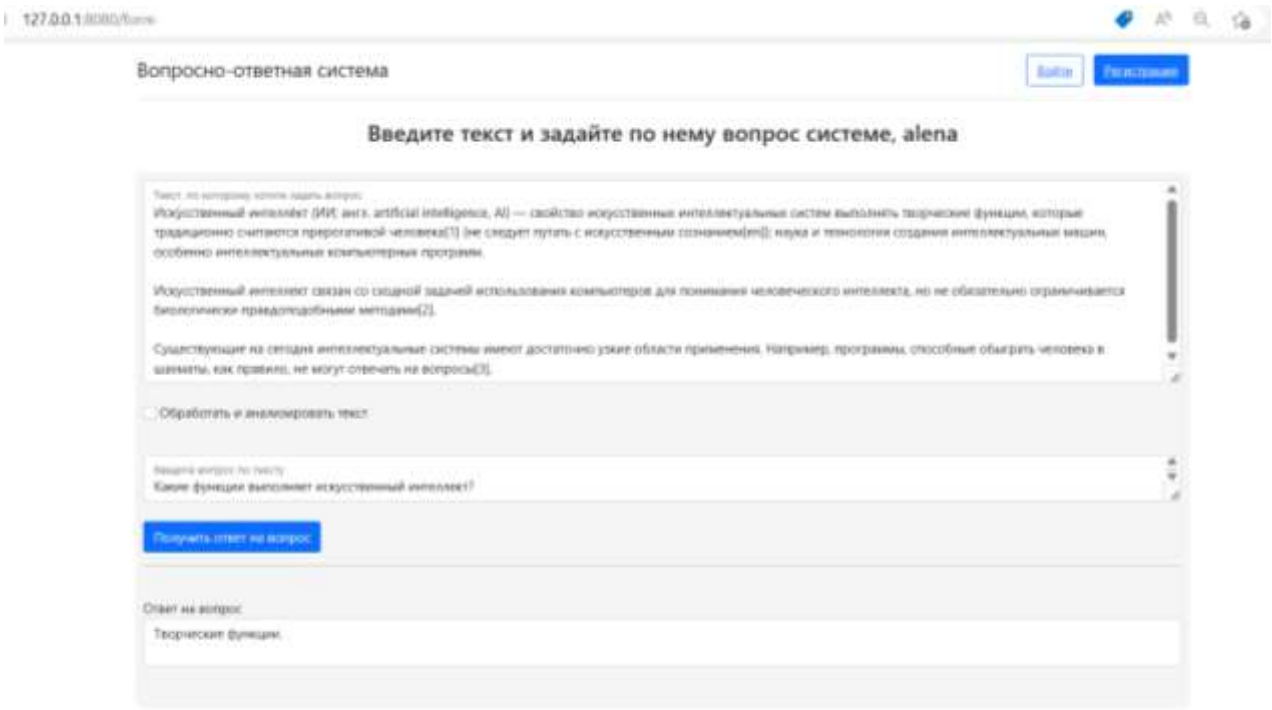

Рисунок А.4 – Пример работы системы

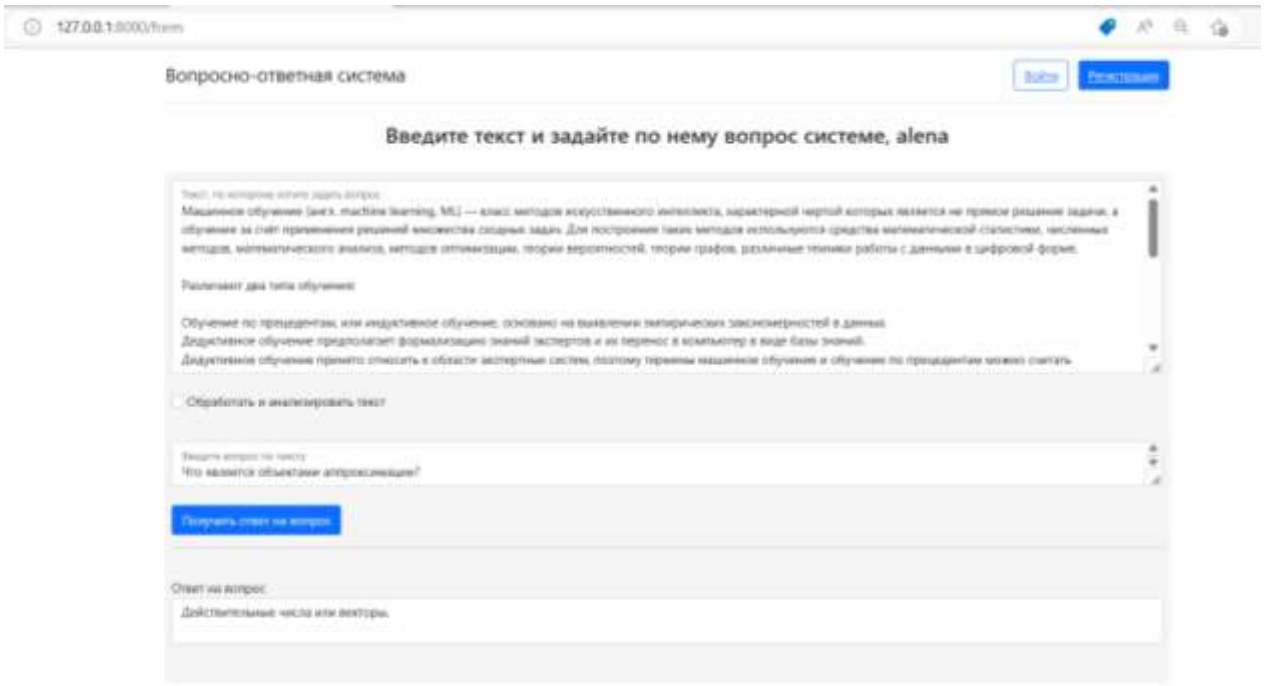

Рисунок А.5 – Пример работы системы

ПРИЛОЖЕНИЕ Б. ИСПОЛЬЗОВАНИЕ ЯЗЫКОВОЙ МОДЕЛИ

```
from deeppavlov import build_model, configs


class Model:

    def __init__(self, download=True):
        self.model_ru = build_model(configs.squad.squad_ru_bert, download=download)

```

Рисунок Б.1 – Использование языковой модели BERT