Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ _____Информатика и системы управления_____

КАФЕДРА _____Программное обеспечение ЭВМ и информационные технологии_____

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ:

__Метод генерации запросов к реляционной базе _____данных на ограниченном естественном _____ __языке с использованием семантической сети ___

Студент ___ИУ7-42м_____ (Группа)	____ _____ (Подпись, дата)	____**Бородин Д.С.**____ (И.О.Фамилия)
Руководитель ВКР	____ _____ (Подпись, дата)	__**Клышинский Э.С.** _ (И.О.Фамилия)
Консультант	_____ (Подпись, дата)	__ **Строганов Ю.В.**___ (И.О.Фамилия)
Консультант	_____ (Подпись, дата)	_____ (И.О.Фамилия)
Нормоконтролер	____ _____ (Подпись, дата)	____**Строганов Ю.В.**__ (И.О.Фамилия)

2019 г

Реферат

Расчетно-пояснительная записка 111 с., 4 ч., 34 рис., 8 табл., 51 источник

ОБРАБОТКА ЕСТЕСТВЕННОГО ЯЗЫКА, СЕМАНТИЧЕСКИЕ СЕТИ, СИНГУЛЯРНО-ВЕКТОРНОЕ РАЗЛОЖЕНИЕ, СЕМАНТИЧЕСКАЯ БЛИЗОСТЬ, ВЕКТОРНОЕ ПРЕДСТАВЛЕНИЕ СЛОВ, ТРАНСЛЯЦИЯ

Объектом исследования является использование естественного языка для коммуникации человека с ЭВМ. Объект разработки – система, позволяющая осуществить трансляцию запроса на естественном языке в запрос к реляционной базе данных. Задачи, решаемые в работе:

- анализ задачи преобразования естественно-языковых запросов в информацию, пригодную для машинной обработки;
- определение формата входных данных для системы;
- определение формы представления данных в системе;
- разработка метода автоматического построения семантических сетей;
- транслирование запросов к базе данных на основании извлеченной из текста информации;

Область применения – информационно-поисковые тезаурусы, интеллектуальные интерфейсы.

В первой части работы описывается актуальность разработки посредством прогнозирования развития рынка обработки естественного языка на ближайший период. Описываются существующие системы, реализующие подобный функционал. Рассматриваются основные методы генерации семантических сетей в зависимости от выделяемых элементов. Во второй части работы описываются основные методы, реализуемые в данной работе. В третьей части представлен выбор технических средств, структура разрабатываемого программного обеспечения, описание основных моментов реализации. В четвертой части приведены исследования количественных характеристик выделения основных элементов семантической сети, а также временных характеристик работы основных компонентов системы.

Предполагаемые направления дальнейшего развития:

- расширение числа классов обрабатываемых запросов;
- модификация основных компонент системы с целью ускорения процесса настройки и обработки естественного языка;
- модификация системы с целью обработки сложных и вложенных запросов

Поставленная цель достигнута: система обработки естественно-языковых запросов к реляционной базе данных разработана, спроектирована, реализована и проверена на практике. Были рассмотрены существующие достоинства и недостатки и предложены пути дальнейшего развития.

Содержание

Введение

Современное состояние развития общества говорит нам о том, что все чаще и чаще в различных сферах деятельности уделяется внимание технической составляющей различных процессов, происходит их автоматизация и внедрение передовых технологий. В частности, для САПР стоит задача автоматизации обработки пользовательских запросов к базам данных в целях сокращения цикла обучения пользователей. Эта задача может быть решена при помощи методов, преобразующих текст на естественном языке в запросы к базе данных на формальном языке.

Целью данной работы является разработка алгоритма, позволяющего строить семантические сети в автоматическом режиме, что позволяет упростить разработку естественно-языковых интерфейсов к различным базам данных, а также системы, производящей трансляцию естественно-языковых запросов в запросы к реляционной базе данных. Для достижения поставленной цели необходимо решить задачи, связанные с выделением из естественно-языковых источников информации основных компонент любой семантической сети – сущностей, связей и атрибутов сущностей. Дополнительно необходимо решить задачи, связанные с преобразованием запросов и транслированием запросов в информацию, пригодную для машинной обработки.

Результат данной работы может быть применим в конструировании различного рода систем с интерфейсами на естественном языке. Такие системы широко используются в различных сферах человеческой деятельности, таких как электронная коммерция, медицинское обслуживание, образование, развлекательная индустрия и многие другие области, где широко внедрение и использование различных баз данных, а также происходит контакт людей, малознакомых с технологическими составляющими работы с данными.

1. Аналитический раздел

1.1. Обзор предметной области

1.1.1. Естественно-языковые системы

Во второй половине XX века в сфере информационных технологий появилось новое направление, получившее название «обработка естественного языка» (Natural Language Processing - NLP). Это направление предполагало разработку различных систем, призванных решить задачи, связанные с анализом и преобразованием всевозможных текстов на языке общения людей между собой. Такие системы стали называться естественно-языковыми (ЕЯ-системы).

На сегодняшний день существует множество различных классов таких систем. Все они условно могут быть описаны следующей схемой (рисунок 1) [1]:

Рисунок 1. Основные классы естественно-языковых систем

При этом естественно-языковые системы призваны выполнять различные функции, такие как поддержание диалога с пользователем, предполагающую знание системой структуры диалога, а также ролей участников диалога в каждый момент времени, а также функции связанные с организацией общения между системой и пользователем:

- преобразование предложений на естественном языке в информацию, пригодную для машинной обработки (для этого данные трансформируют в высказывания на языке внутреннего представления системы);

- обработка преобразованных предложений, заключающаяся в постановке задач для системы на основании полученных данных на текущем шаге диалога;
- формирование ответа, который может быть выражен, как предложением на естественном языке, так и выборкой запрошенных данных в случае систем общения с базами данных.

Естественно-языковые системы обладают целым рядом преимуществ по сравнению с системами, в которых отсутствует поддержка интерфейсов на языке, понятном человеку – это и минимальная предварительная подготовка пользователя такой системы, и простота задания запросов на выборку данных на естественном языке, и большая скорость создания произвольного запроса засчет отсутствия стадии формального задания запроса [1].

Однако, естественно-языковые системы не лишены недостатков. К ним относят появление множественности смыслов ввиду неоднозначности естественного языка, что очень хорошо отражается на примере русского языка, где «семафор» в мореплавании совершенно не то же самое, что «семафор» в проектировании операционных систем. Более того, анализаторы естественно-языковых запросов, используемые в таких системах, могут быть недостаточно надежными, что приведет к неправильному пониманию текста исходного запроса, а соответственно, к неправильной обработке.

При проектировании естественно-языковых систем необходимо учитывать вопрос используемых в системе знаний, так как именно они позволяют обеспечить решение задач, связанных с реализацией функционала таких систем. Исходя из классификации знаний, представленной в [1], можно сделать вывод о том, что при рассмотрении вопроса использования знаний в системе, использующей естественно-языковой интерфейс, важно учитывать область знаний, а также формат представления. Условно, существуют логические и эвристические модели представления знаний. Логические делят на реляционные, псевдофизические логики и знания, использующие исчисления предикатов. Отдельного внимания заслуживают эвристические модели. Их

разделяют на сетевые, продукционные и фреймовые. Продукционные представляют собой набор правил, содержащих утверждения о содержимом базы данных и действия, позволяющих модифицировать эти данные. Сетевые представляют собой использование семантических сетей – направленных графов, где в качестве узлов содержатся объекты, а в качестве дуг – отношения между этими объектами. И, наконец, фреймовая модель, представляющая собой набор действий в конкретной языковой ситуации.

1.1.2. Компьютерная лингвистика

Компьютерная лингвистика - направление в прикладной лингвистике, ориентированное на использование компьютерных инструментов – программ, компьютерных технологий организации и обработки данных – для моделирования функционирования языка в тех или иных условиях, ситуациях, проблемных сферах и т.д., а также вся сфера применения компьютерных моделей языка в лингвистике и смежных дисциплинах [2]. Компьютерная лингвистика возникла на стыке таких наук, как математика, информатика, лингвистика и искусственный интеллект.

Объектом компьютерной лингвистики являются различные тексты. Под текстом понимается любой образец письменной или устной речи, имеющий одномерную и линейную структуру и несущий в себе определенный смысл. Язык же, в свою очередь, выступает в роли средства приема и передачи информации [3]. При этом текст состоит из собственных структурных единиц, которые могут быть причислены к различным уровням. На сегодняшний день общепризнанно были выделены следующие уровни [3]:

- уровень предложений (высказываний) – синтаксический уровень;
- уровень слов (словоформ – слов в определенной грамматической форме, например, столом, дружбы) – морфологический уровень;
- уровень фонем (отдельных звуков, с помощью которых формируются и различаются слова) – фонологический уровень;

Так как в постановке нашей задачи не стоит вопроса о распознавании речевой информации, этот уровень не несет для нас особой смысловой нагрузки в данной работе. Наибольший интерес представляет обработка письменной текстовой информации на естественном языке, а следовательно, необходимо учитывать остальные уровни. Также следует отметить, что письменные запросы на естественном языке зачастую составляются с соблюдением грамматических и пунктуационных норм языка. Таким образом, необходимо учитывать это при обработке запросов на естественном языке, вводя при этом графематический уровень (уровень всех структурных единиц – абзацев, слов, знаков препинания и пр.)

Учитывая вышесказанное, можно предположить, как будет выглядеть система обработки текстовых запросов на естественном языке. Она будет представлять собой программу, производящую поэтапную обработку с использованием нескольких модулей:

- графематический анализ;
- морфологический анализ;
- синтаксический анализ;

Графематический анализ представляет собой первоначальный этап обработки текста, в ходе которого определяются элементы грамматической структуры (слова, знаки пунктуации, числа, сокращения и т.д.) [4]. При этом можно выделить следующие функции графематического анализа [5]:

- разбиение текста на графемы;
- определение границ предложений;
- различение слов и служебных графем (например, знаков пунктуации);
- определение регистра слов;
- распознавание собственных имен;
- распознавание сокращений;

Морфологический анализ — это процесс определения грамматического значения словоформы и выделения ее основы. Может быть словарным (со словарем основ и окончаний или словарем словоформ) или бессловарным

(только со словарем окончаний; может быть встроен в алгоритм морфологического анализа). Бессловарный метод используется только для определения переменной морфологической информации (не всегда однозначно), а словарный - во всех остальных случаях.

Задачей синтаксического анализа является определение частей предложения и их связи между собой. Результатом данного этапа является синтаксическое дерево, отражающее такие связи. Следует отметить, что получаемые в результате синтаксического анализа деревья должны быть адаптированы к представлению компьютером информации, содержащихся в них.

Немаловажной проблемой при обработке текстовой информации на естественном языке является распознавание смысла содержащихся в тексте слов и выражений, а также различных коллокаций, представляющих синтаксически и семантически неделимые единицы, выбор компонентов которых зависит от контекста. Эта проблема решается средствами семантического анализа, в котором производится оценка различного рода многозначностей, связанных с большой системностью естественного языка.

Добавим, что омонимия существенно проявляется на всех уровнях естественного языка, укажем некоторые ее виды:

- Лексическая омонимия означает одинаково звучащие и пишущиеся слова, не имеющие общих элементов смысла, например, рожа – лицо и вид болезни.
- Морфологическая омонимия – совпадение форм одного и того же слова (лексемы), например, словоформа круг соответствует именительному и винительному падежам.
- Лексико-морфологическая омонимия (наиболее частый вид) возникает при совпадении словоформ двух разных лексем, например, стих – глагол в единственном числе мужского рода и существительное в единственном числе, именительном падеже).

- Синтаксическая омонимия означает неоднозначность синтаксической структуры, что приводит к нескольким интерпретациям: Студенты из Львова поехали в Киев и другие.

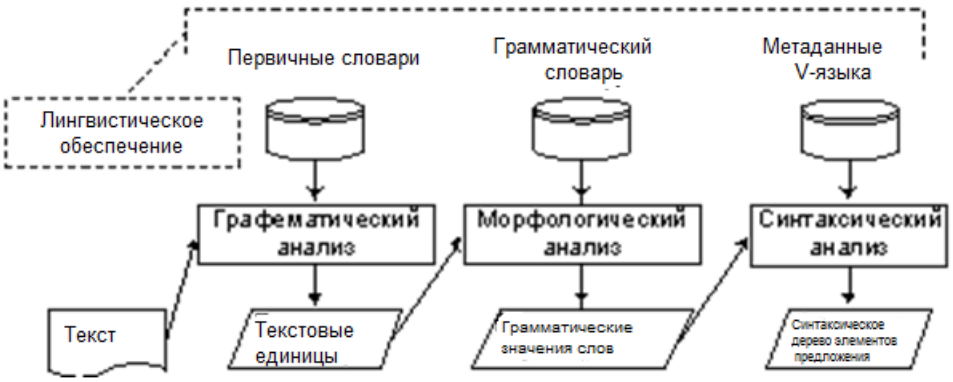


Рисунок 2. Схема лингвистического модуля.

При этом мы предполагаем, что составляемые на естественном языке текстовые запросы к системе будут составляться однозначно, что отбросит необходимость проведения семантического анализа текста. Таким образом, обработка предложений на естественном языке будет завершаться этапом синтаксического анализа с получением в качестве выходного результата синтаксического дерева, имеющего определенный набор объектов и свойств, который впоследствии может быть преобразован в запросы на формальном языке.

Описанные выше этапы автоматической обработки текста носят довольно сложный характер и тесную связь между собой, т.к. данные, получаемые на выходе одних этапов, служат входными данными для других этапов. Далее в работе будет приведен анализ имеющихся программных средств автоматической обработки текстовой информации на естественном языке, т.к. на сегодняшний день существует довольно много средств автоматической обработки текстов на естественном языке, дающих определенные результаты.

1.1.3. Семантические сети

Семантическая сеть является одним из распространенных способов представления знаний в естественно-языковой системе. Это структура данных, которая несет в себе определенный смысл. Несмотря на отсутствие стандартного определения, под семантической сетью понимают следующее [6]: «Семантическая сеть – это система знаний, имеющая определенный смысл в виде целостного образа сети, узлы которой соответствуют понятиям и объектам, а дуги – отношениям между понятиями и объектами».

Исходя из данного определения, можно выделить основные элементы любой семантической сети, а именно понятия (или объекты) и свойства, которые выражают узлами ориентированного графа, а также отношения между объектами, выраженные дугами. При этом под понятиями представляют сведения об абстрактных или физических объектах предметной области и реального мира, под свойствами – уточнение этих понятий, а под отношениями – связь между объектами семантической сети. В контексте разрабатываемой системы, преобразующей естественно-языковые запросы в запросы к базам данных, можно провести аналогию между сущностями в базах данных и понятиями в семантической сети, так как они оба отражают сведения об объектах реального мира, а атрибуты таблиц баз данных отражают свойства объекта. Поэтому в дальнейшем под сущностью будем понимать понятие семантической сети, а под атрибутами свойства объектов моделируемой семантической сети. При этом связь между объектами будем понимать в первозданном виде.

Дополнительно следует отметить виды связей в семантических сетях. Условно, их разделяют на лингвистические, логические, теоретико-множественные и квантифицированные [6]. В нашей работе мы сконцентрируемся на лингвистических связях, так как они отображают смысловую взаимосвязь между событиями, сущностями и атрибутами. При этом лингвистические отношения бывают глагольными (выражены глаголом с соответствующими характеристиками – время, вид, род, залог, наклонение),

атрибутивными (цвет, размер, форма) и падежными. Так как в разрабатываемом методе предполагается выделение атрибутов сущностей семантической сети, то было принято решение сконцентрироваться на глагольных лингвистических отношениях между сущностями.

1.2. Анализ экономической обстановки в предметной области

Несмотря на выделение обработки естественного языка в отдельное направление, ее существование неотделимо от более общей темы — искусственного интеллекта. Так, некоторые аудиторские агенства, например Deloitte, рассматривает обработку естественного языка, как одно из технологических направлений искусственного интеллекта [7]. Поэтому, для того, чтобы определить актуальность и целесообразность работы в выбранном направлении, необходимо проанализировать текущую и последующую экономическую обстановку в отрасли информационных технологий, связанных с искусственным интеллектом.

По состоянию на сегодняшний день, рынок искусственного интеллекта привлекает все больше инвесторов засчет своих преимуществ по автоматизации процесса производства, причем большее внимание уделяется различного рода стартапам. Так, во второй половине 2016 года инвестиции в развитие искусственного интеллекта достигли максимума за всю историю. При этом основная доля сделок (около 60%) приходится на начальные этапы развития стартапов [7]. Это было достигнуто засчет нескольких крупных сделок, к которым относят китайский iCarbonX, занимающийся разработками в области медицины ($154 млн), более $100 млн было вложено в американский FractalAnalytics, еще около $100 млн было инвестировано в Cylance – систему обеспечения кибербезопаности. Основной процент сделок (примерно 70%) приходится на США.

По данным CBInsights к концу 2016 года было приобретено 140 частных компаний, занимающимися разработками в области искусственного интеллекта, причем 40 приобретений приходится на 2016 год. Объем

глобальных инвестиций в данную отрасль превышает 500 млн долларов. По прогнозам исследовательской компании Market&Markets, рынок искусственного интеллекта вырастет до $5 млрд засчет использования технологий распознавания естественного языка и машинного обучения в рекламе, розничной торговле, финансах и здравоохранении. При этом, по данным Gartner, к 2020 году почти половина всех коммуникаций с виртуальными помощниками будет приходиться на данные, полученные с помощью нейронных сетей.

Общий тренд доходности рынка искусственного интеллекта по прогнозам на 2016-2025 годы представлен на рисунке 3.

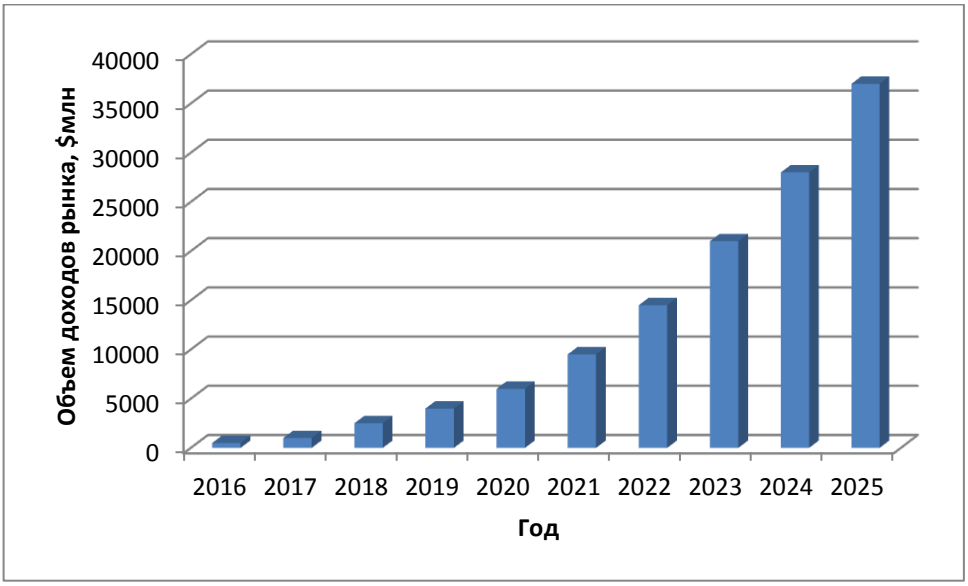


Рисунок 3. Доходы рынка искусственного интеллекта в 2016-2025 годах (по версии Tractica)

Рынок обработки естественного языка также отмечается аналитиками, как устойчиво растущий. Так, агенство Market&Markets оценило рынок естественно-языковой обработки в $7,64 млрд и его рост до $16,07 млрд к 2021 году со среднегодовым темпом роста с учетом сложного процента (Compound Annual Growth Rate – CAGR) в 19,1%. При этом основная доля компаний, занимающаяся данной отраслью, приходится на США (Amazon, Apple, Google, IBM, Microsoft, NetBase Solutions и другие) и Великобританию (Linguamatics) [7], толчком в развитии которых служит внедрение новых технических средств и более совершенных и производительных технологий.

Российские компании также не остаются в стороне в развитии данной отрасли. Так, компания «ЦРТ-инновации» разработала метод распознавания речевой информации, который был использован Microsoft в их системе речевой обработки, которая заявлена как ничуть не уступающая по уровню распознавания речи человеком. Компания Яндекс разрабатывает систему для распределения заказов в сервисе Яндекс.Такси, КамАЗ занимается разработкой полуавтономного управления автомобилем, а робот «Вера», разработанный петербургской компанией Stafory, способен выполнять работу по собеседованию с потенциальными кандидатами на вакансии, что значительно ускоряет работу персонала. Также стоит отметить Сбербанк, анонсировавший в конце 2016 года запуск робота-юриста и систему обзвона должников годом ранее.

По мнению аналитиков компании J'son&Partners основное внимание в России будет приковано к таким сферам применения методов распознавания речи и обработки языка, такие как транспорт, финансовая сфера и коммуникации. Однако, в дальнейшем планируется охватить все сферы деятельности общества [7].

1.3. Обзор существующих программных продуктов

Согласно [1], в направлении разработки естественно-языковых систем выделяют следующие категории программных продуктов:

- программы, реализующие интерфейсы к базам данных на естественном языке;
- средства естественно-языкового поиска в текстах;
- программы, выполняющие сканирование текстов;
- масштабируемые средства распознавания речевой информации;
- программы, связанные с голосовым управлением, сбором и вводом данных;
- прочие компоненты речевой обработки;

В рамках реализуемой работы для нас наибольший интерес представляют аналоги естественно-языковых систем, реализующих работу с базами данных. Это могут быть как обычные системы работы с базами данных на естественном языке, так и вопросно-ответные системы, поскольку они имеют схожий принцип работы, за исключением этапа формирования ответа. В вопросно-ответных системах ответ генерируется на языке запроса, в то время как в системах общения с базами данных ответ может формироваться из обычного набора данных, хранящихся в базе. Рассмотрим некоторые аналоги естественно-языковых систем.

1.3.1. Вопросно-ответные системы (Question & Answering Systems – QA-системы)

1. Система START

Данная система, разработанная в 1993 году, является наиболее известной и общей вопросно-ответной системой, обрабатывающей запросы на английском языке, главным отличием которой является универсальность. В качестве источников знаний использует собственную базу знаний, а также сеть Интернет.

Особенность этой системы заключается в том, что она реализует идею создания аннотаций на естественном языке к информации с целью более быстрой и качественной обработки исходного запроса и получения конечного ответа. Для этого применяются специально разработанные методы естественно-языковой обработки информации [8].

Вопросно-ответная система START способна обрабатывать следующие основные виды вопросов:

- вопрос о фактах (Who invented the telescope?);
- вопрос об определениях (What is a natural language processing?);
- вопросы об отношениях (Who is taller, Lebron James or Michael Jordan?);
- списковые запросы (show me list of films by Quentin Tarantino);

База знаний этой системы состоит из каталога слов, тернарных выражений и синтаксико-семантических правил вывода (S-правила), описывающих лексические, морфологические, синтаксические операции, а также использующиеся для описания логических импликаций. Основное преимущество S-правил заключается в двунаправленном использовании – для обработки запроса пользователя и пополнения базы знаний новыми тернарными выражениями. В качестве каталога слов используется WordNet, состоящий из синсетов (наборов синонимичных слов с описанием их значений). Каталог слов необходим для поиска соответствий тернарных выражений в базе знаний.

Также система START содержит при себе базу данных Omnibase, к которой осуществляются запросы на выборку фактов. Эта база данных реализует модель «ОБЪЕКТ – СВОЙСТВО - ЗНАЧЕНИЕ» для удобства поиска фактов по запросу засчет естественности использования такой модели. Однако, такая модель требует при себе обертки для каждого вида данных, что не очень удобно.

Еще одним важным свойством такой системы является использование семантической сети для описания ресурсов с помощью модели «ОБЪЕКТ – ОТНОШЕНИЕ - СУБЪЕКТ» [8].

На сегодняшний день одной из задач, решаемых командой разработчиков системы является автоматизация анализа семантических связей в документах и автоматическая декомпозиция.

Более подробное описание системы и примеры ее использования приведены в [8].

2. Женя Густман (Eugene Goostman)

Является наиболее ярким представителем таких вопросно-ответных систем, как чатботы – компьютерные программы для общения с человеком и имитирующие его поведение. Изначально разработан в Санкт-Петербурге в 2001 году группой российско-украинских разработчиков. Основную

известность получил в 2014 году, когда смог убедить 33% судей в том, что они общаются с живым человеком, тем самым став первой машиной, прошедшей тест Тьюринга.

Основными особенностями этой вопросно-ответной системы являются использование интерпретатора логических и математических выражений, поддержка контекста темы, наличие валидатора паттернов со сложными конструкциями и интеграцию паттернов с базой данных для использования онтологий. Также в системе используется специальный контроллер диалога, реализующий анализ, планирование и достижение поставленной в диалоге цели.

Однако, данная система не способна выделять факты из диалога в отличие от некоторых других систем этого класса. Более подробное сравнение основных вопросно-ответных систем такого вида приведено в таблице 1. [9]

Функции управления диалогом	ALICE	Facade	ChatScript	Eugene Goostman
Извлечение фактов из диалога	нет	да	да	нет
Переменные состояния	да	да	да	да
Предопределенные диалоги	да	нет	да	да
Определение контекста	нет	нет	да	да
Контроллер цели	нет, может быть реализован	нет, может быть реализован	нет, может быть реализован	да

Таблица 1. Сравнение функций управления диалогом в наиболее известных чатботах

3. Простая вопросно-ответная система на основе семантического анализатора русского языка

Эта система, как утверждает разработчик, «является простым примером использования» семантического анализатора русского языка, разработанного В.А. Тузовым [10]. В качестве хранилища данных используется обычный текстовый файл, в котором проверяются поочередно все предложения на предмет нахождения ответа на поставленный естественно-языковой вопрос. Система не пытается формировать какую бы то ни было базу знаний и обрабатывает специальные вопросы (вопросы к членам предложения).

Принцип работы данной вопросно-ответной системы реализуется с помощью синтаксического анализа тестируемой фразы и последующим анализом полученного дерева на предмет соответствия модифицированному дереву вопросительного предложения. В качестве ответа получают поддерево предложения, которое преобразуется в необходимую форму. При этом также используется словарь словоформ для решения проблемы лексикографического сравнения слов.

Более подробный принцип работы такой вопросно-ответной системы описан в [10].

4. RAZOOM

Программа, представляющая собой собеседника-эксперта, способного поддерживать диалог с пользователем в режиме социальной сети. Данная система ориентирована на анализ эмоционального поведения собеседника путем обработки информации, получаемой в процессе диалога.

Согласно заявлениям разработчиков приложения, эта система призвана выполнять следующие функции:

- анализ речевой информации на основе смысловой нагрузки без использования лингвистических правил;

- запоминание вкусов, интересов, желаний пользователя на основе сказанного;
- определение рекурсии диалога, выявление желаний пользователя на основании информации, накопленной о нем;
- оценка правдивости и эмоциональной окраски сообщений пользователя;
- другие функции [11];

Разработанная система предоставляет доступ к своему API по запросу разработчика. Так как разработка системы носит закрытый характер, нет возможности оценить используемые методы обработки информации и способы представления знаний в ней. На момент написания работы программа находится в разработке с полугодовой перспективой запуска открытого бета-тестирования.

5. ITFRU

Вопросно-ответная система ITFRU является еще одним проектом, находящимся на стадии разработки на текущий момент. Последняя стабильная версия программы позволяет задавать общие вопросы к онтологии, построенной на базе тезауруса РуТез-lite 2.0 и способна обрабатывать лишь ограниченное подмножество запросов в рамках небольшого количества типов отношений. Например: «Кто имеет хвост?» или «Информация о СССР». [12]

Текущий этап развития программы – реализация обработки естественно-языковых вопросов к русскому тексту, ориентировочная дата появления – 20 марта 2017 года. Разработка также носит закрытый характер и невозможно проанализировать компоненты системы и имеющуюся базу знаний.

1.3.2. Системы общения с базами данных

1. IBM Watson

На сегодняшний день IBM Watson является когнитивной системой, способной к обучению на основе диалога с живыми людьми. Сейчас это

облачная технология, которая насчитывает около 50 сервисов и предоставляет 28 API для разработчиков. Является частью проекта DeepQA по распространению естественно-языкового доступа к контенту по всему миру. Новый скачок популярности данная система приобрела после того, как смогла обыграть людей в аналог «Своей игры» - игру “Jeopard!”

Среди наиболее популярных сервисов в IBM Watson выделяют «Инвестиционный советник», классификатор вопросов на естественном языке, сервис «Что в кино?», в котором реализовано естественно-языковое общение с базой данных themoviedb.org.

Основным преимуществом IBM Watson является наличие открытого исходного кода для различных сервисов, что позволяет проанализировать архитектуру подобных приложений и системы обработки языковой информации. Однако, эта система не имеет аналога в российском сегменте рынка, так как развитие подобных технологий в России связано с многими проблемами обработки русскоязычных текстов.

Более подробное описание системы IBM Watson приведено в [15].

2. InBase

Эта технология, разрабатываемая РосНИИ Искусственного интеллекта, предназначена для естественно-языкового доступа к базам данных. На сегодняшний день представлена в виде коммерческого продукта, являющего собой, по заверениям разработчиков, прямое продолжение технологии InterBASE начала 90-х годов XX века и более ранней ЗАПСИБ.

На момент создания технологии InBase отмечал в качестве основной особенности использование построенной модели предметной области, являющейся связующим звеном между текстом на естественном языке и представлением информации в базах данных.

В основе технологии – семантически ориентированный подход, заключающийся в интерпретации грамматических и лексических единиц языка в понятиях предметной области. Основным принципом семантически

ориентированного анализа является моделирование взаимодействия типов с одинаковыми значениями ориентаций [13]. Использование такого подхода позволяет отделить язык от предметной области и сделать систему более общей для различных предметных областей.

Отмечается, что важной деталью в использовании технологии является выделение предметной области и ее увязка с моделью базы данных. Для визуального описания моделей используется диаграмма классов, схожая с UML-диаграммой. Машинное представление моделей описывается в виде XML-документов.

Еще одним отличием технологии InBase является наличие промежуточного представления запроса в виде “Q-языка”, который по структуре схож с языком реляционных баз данных, но основан на объектно-ориентированной парадигме. Использование дополнительного языка позволяет абстрагироваться от источников хранения данных [13]. Также следует отметить наличие словаря в данной системе, наполнение которой зависит от набора классов статей, выделенных в зависимости от предметной области. Для обработки объектно-ориентированной семантической сети используется лингвистический процессор, работающий в продукционных правилах.

3. Транслятор естественно-языковых запросов к базам данных (ВСГТУ)

Данная разработка, зарегистрированная в ноябре 2014 года, позволяет пользователю писать запрос к базе данных на естественном языке. При этом транслятор автоматически анализирует запрос пользователя, переводит его в запрос к СУБД, обращается к базе данных и выводит пользователю требуемые результаты поиска.

В качестве исходной информации система принимает запрос к SQL-ориентированной базе данных. При этом на запросы накладываются следующие ограничения [14]:

1. Запрос может состоять из одного или нескольких предложений на русском языке;

2. Запрос должен требовать информацию, соответствующую рассматриваемой базе данных;

3. Запрашиваемая в запросе информация должна иметь четкий характер, то есть не содержать указательных местоимений;

4. Рекомендуется начать вопрос со специальных слов, таких как «показать», «где», «какой» и другие;

5. Если запрос состоит из нескольких предложений, то каждое следующее предложение должно содержать существительное, приведенное в ранее рассмотренных предложениях.

Собственно модель транслятора представлена в работе в виде трехуровневой системы, состоящей из лингвистической модели, базовых механизмов обработки предложений и ассоциированных процедур[14]. В лингвистической модели производится синтаксический анализ с преобразованием полученных результатов в семантические отношения. Дополнительно проводится морфологический анализ для более удачного выполнения синтаксического анализа. Как синтаксический, так и морфологический анализ представлены целым набором правил, описанными в [14] в качестве баз правил соответственных видов анализа. Следует отметить, что описание лингвистические модели производится по методике Т.М. Яхно, в которой вводятся понятия термов, констант и функциональных символов. Также определяются факты, подстановки и продукция, как основные элементы описания лингвистической модели.

Еще одним немаловажным фактором работы такой системы является использование различных словарей готовых словоформ, основ, окончаний и других для более точного проведения морфологического анализа, описывается метод проведения такого анализа.

Модель синтаксического анализа в рассматриваемой системе основана на знаниях о пунктуации, порядке слов в предложении, морфологических характеристиках и отношениях словоформ, выделяются три группы правил синтаксического анализа, более подробно описанные в [14].

Сама же модель трансляции в системе описывает преобразование графа, вершинами которого являются словоформы, полученные в ходе морфологического и синтаксического анализов в конечный SQL-запрос. Вводится основная терминология о понятиях, которые в нашей работе именуются сущностями, а также об отношениях, возможных между этими видами сущностей. Метаописание понятий производится с помощью составных предикатов [14]. Для успешного проведения процесса трансляции в системе рассматриваются знания, прямо или косвенно имеющие отношения к базе данных и рассматривается метод преобразования зависимостей терминов логической модели в граф зависимостей терминов физической модели, на основании которого формируется конечный SQL-запрос.

Описанные в представленной работе методы анализа текста и трансляции позволяют реализовывать естественно-языковой интерфейс к базам данных для разных предметных областей. Однако, в данной работе нет четкого представления о наполнении базы данных в зависимости от разных предметных областей, что вынуждает разработчика учитывать более сложные проблемы, которые могут возникнуть при конструировании системы, например, неоднозначность терминов в зависимости от предметной области.

1.4. Анализ методов и алгоритмов автоматического формирования семантической сети

1.4.1. Анализ методов выделения сущностей

I. Алгоритмы поиска ключевых слов

Ключевые слова представляют собой такие функциональные единицы текста, которые позволяют составить описание текста и определить его тематику. Получаемый набор ключевых слов может использоваться для метаописания текстовых документов, необходимых для поиска, кластеризации, аннотирования и реферирования [16].

На сегодняшний день алгоритмы поиска ключевых слов разделяют необучаемые, самообучаемые и обучаемые. По аппарату распознавания выделяют статистические, нейросетевые, гибридные и структурные. По лингвистическим ресурсам методы делят на бессловарные, словарные, а также на методы, которые реализованы на основе онтологий и корпусов, как неразмеченных, так и размеченных [17].

Рассмотрим наиболее известные и эффективные методы извлечения ключевых слов в тексте.

1. TF-IDF

Данный алгоритм основан на статистическом подходе, главной идеей которого является предположение о том, что слова, встречающиеся в тексте наиболее часто, отражают основную мысль текста. Является наиболее простым алгоритмом из всех алгоритмов данного подхода, так как рассчитывает вес слова как величину, прямо пропорциональную количеству появлений терма t в документе и обратно пропорциональную количеству появлений в других документах текста.

Для вычисления меры, используемой в данном алгоритме, получают значение двух величин: term frequency (tf) – число появлений слова в тексте документа и inverse document frequency (idf) – обратная документальная частота, определяемая как логарифм отношения числа появлений слова в других текстах документа к общему числу документов. Затем происходит вычисление «весов» слов по формуле $tf - idf_{t,d} = tf_{t,d} \cdot idf_t$, где t – текущий терм (слово), d – текущий документ. Слова с наибольшими весами считаются ключевыми.

Основным недостатком методов, основанных на статистическом подходе является то, что они не учитывают связность слов в тексте, что может оказывать существенное влияние на конечный результат [18]. При этом данная проблема может быть решена с помощью методов лингвистического анализа.

2. Функциональный подход к выделению ключевых слов

Методика, разработанная соотечественниками из Воронежского Государственного университета в 2009 году, опирается на использование алгоритма тематически маркированной лексики [19]. Взвешивание слов производится по двум параметрам – частотному и длине слова. Взвешивание по длине слова производится из соображения о том, что чем чаще слово употребляется в тексте, тем оно короче. При этом производится соотнесение параметров длины слова и частоты встречаемости. Разработчики считают, что стабильно часто встречающееся длинное слово в тексте будет определять его специфику.

Параметр частоты встречаемости определяется по формуле:

$$Q_n = \frac{\sum r - R_{i-1}}{\sum r},$$

где $\sum r$ - сумма единиц всех рангов в частотном словаре (словарь содержит упорядоченные по убыванию частоты встречаемости слова, где наиболее часто встречающиеся имеют ранг 1, далее по убыванию происходит увеличение ранга на единицу), а R_{i-1} - сумма единиц от первого до данного ранга включительно. Аналогичная формула применяется для параметра F_n, где используется словарь, в котором слова упорядочены по убыванию длины слов в звуках. Далее вычисляется разность Q и F, при этом положительные значения разности характеризуют тематически маркированную лексику, то есть необходимые ключевые слова.

Результатом применения данного метода является выделение однословных терминов в количестве 15% от общего числа всех ключевых слов и 40% терминологических сочетаний[19].

Основным недостатком алгоритма считается неспособность выделения ключевых слов с низкой частотой встречаемости (до 1-3 раз в тексте). При этом, остальные выделенные слова на 100% попали в выделенные вручную экспертом ключевые слова.

3. Извлечение ключевых терминов из микроблогов с помощью Википедии

Является одной из последних модификаций алгоритмов, использующих статистический подход. При этом в качестве источника данных для извлечения необходимых ключевых слов используется Википедия, как наиболее полная энциклопедия, содержащая большие объемы информации.

Работа алгоритма осуществляется в три этапа: препроцессинг, в котором исходный текст преобразуется к формату входных данных, далее происходит формирование списка всех возможных ключевых терминов путем перебора всех возможных N-грамм – последовательностей идущих друг за другом слов. Третьим этапом происходит удаление из списка терминов последовательностей, содержащих стоп-слова (слова, не несущие смысловой нагрузки) и упорядочение списка терминов в порядке убывания весов [20].

Недостатки этого алгоритма заключаются в неэффективности работы алгоритма по времени засчет перебора всех возможных N-грамм и небольшой процент выделения необходимых ключевых терминов засчет наличия большого числа именованных сущностей, распознающихся, как ключевые термины, однако не отражающие общей смысловой нагрузки в тексте.

4. Структурные методы извлечения ключевых слов

Алгоритмы, реализующие структурный подход, опираются, в основном на теорию графов. Так, алгоритм TextRank, разработанный в 2004 году, использует граф для оценки важности слов в тексте. При этом, значимость вершины графа рассчитывают через значимости смежных вершин. Смежность определяется расстоянием между словами, происходит фильтрация лексики перед постановкой вершин в граф, затем происходят расчеты значимость и упорядочение слов по убыванию важности. Причем ключевыми словами в данном методе считают не более 20 первых слов, комбинации которых образуют ключевые словосочетания. Недостатком алгоритма является его чрезмерная сложность [17].

Еще одним граф-ориентированным методом является Rake, в котором список возможных ключевых слов формируется на основании внутреннего словаря, а значимость вершин графа определяется в зависимости от частоты появления вершины в тексте и степени вершины. Для словосочетаний используется принцип суперпозиции – значимость словосочетания является суммой значимостей каждого отдельного слова. [17].

Более простым граф-ориентированным методом считают DegExt, который значительно легче реализуется в отличие от TextRank. Особенностью метода является то, что в начале удаляются стоп-слова, а затем происходит постановка вершин графа и соединение дугами, причем дугами соединяют только те вершины, слова которых находятся рядом в предложении и не разделены знаками препинания. Рекомендуется для выделения около 15 ключевых слов.

Другие методы, такие как метод Шультера, отличаются от вышеописанных лишь набором дополнительных параметров в оценке значимости текущей вершины графа, таких как информация о позиции и длине слова, показателей центральности по степени, близости и посредничеству [17].

Общим недостатком граф-ориентированных методов является усложненная логика вычисления значимости вершин при сравнительно невысоких качественных результатах выделения и затратность таких алгоритмов с точки зрения ресурсов системы.

5. GenEx

Данный метод извлечения ключевых слов из текстовых документов представляет собой обучаемый алгоритм с использованием «выделителя» ключевых слов и генетического алгоритма. Основополагающими факторами определения важности слова является частота появления и позиция первого вхождения слова в текст [17].

Основным преимуществом использования генетических алгоритмов решении задачи выделения ключевых слов является концептуальная простота,

широкая применимость, менее жесткие требования к решению задач, способность к распараллеливанию, устойчивость к динамическим изменениям [21]. Однако, алгоритмы не лишены недостатков, к которым относят неочевидность конфигурации алгоритма при решении задачи, проблему выборов параметров генетического алгоритма, таких как мощность популяции и другие, а также требовательность алгоритмов к вычислительным ресурсам.

6. Алгоритм с использованием метода обнаружения сообществ в сетях Гирвана-Ньюмана

Разработка российских ученых совмещает в себе расчет меры семантической близости слов на основе Википедии и описанный в названии алгоритма метод поиска сообществ в сетях. Неоднозначность терминов в алгоритме решается путем выявления наибольшей семантической близости. Далее строится семантический граф, вершины которого упорядочиваются.

Основным недостатком этого метода считают высокое время реализации. Также неизвестно, как данный алгоритм справляется с русскоязычными текстами, так как, согласно авторам [22] метод тестировался на англоязычных текстах.

Общее сравнение описанных выше методов данной категории производилось в [17] по трем параметрам – точности, полноте и F-мере (отношение выделенных терминов к общему числу терминов в тексте). Наиболее эффективными считаются алгоритм Wikify(94% точности), алгоритм с использованием сетей Гирвана-Ньюмана (полнота около 73%) и алгоритм извлечения терминов из микроблогов на основе Википедии (F-мера 50,6%, третья после двух описанных выше). Однако, данные различных исследователей разнятся засчет использования различных корпусов текстов для тестирования. Вдобавок стоит отметить, что не существует объективной оценки описанных методов для выборки русскоязычных текстов. Более того, как показало исследование, алгоритмы данной группы выделяют в качестве ключевых слов много информации, непригодной для использования в рамках

поставленной перед нами задачи. В частности, большой процент занимают именованные сущности (понятие именованной сущности будет описано далее), которые не несут при себе общности смысла.

II. Алгоритмы выделения именованных сущностей

Именованная сущность – слово, выражающее конкретный, определенный предмет или явление. Оно позволяет выделить его среди прочих похожих объектов данной категории. Обязательным условием именованной сущности является наличие референта – общего понятия, на которое направлен этот объект [23], например: «За футбольную сборную выступили братья Березуцкие». Березуцкие в данном предложении – именованная сущность, а братья – обязательный референт.

Помимо описанных в разделе 4.1.1. алгоритмов, позволяющих выделить определенный процент подобных сущностей, существует целый набор методов, ориентированных конкретно на такой вид сущностей. При этом выделяют подходы, основанные на машинном обучении и на правилах. Рассмотрим отдельно некоторые из имеющихся алгоритмов.

1. Метод использующий деревья принятия решений.

Деревья принятия решений являются одним из методов машинного обучения и используются обычно для классификации данных. Основная идея заключается в построении дерева и графа принятия решений засчет минимизации функций, описанных в [24]. Тестирование данного метода на корпусе текстов ConLL2003 показало эффективные характеристики точности, полноты и F-меры при выделении именованных сущностей.

2. Метод опорных векторов.

Метод опорных векторов является алгоритмом обучения с учителем, который используется для задач классификации. Обучение модели, являющейся бинарным линейным классификатором, происходит на основе

тренировочного множества. Затем происходит построение гиперплоскости позволяющей разделить объекты на классы [25].

3. Скрытые марковские модели и случайные условные поля

Скрытые марковские модели также относят к методам машинного обучения. Модель основана на марковских цепях, в которых мы можем наблюдать только некоторые нескрытые состояния, зависящие от текущего [23]. Применительно к задаче выделения именованных сущностей, скрытым состоянием является тип именованной сущности, а наблюдаемым – слова из текста.

В отличие от скрытых марковских моделей в случайных условных полях текущее состояние зависит не только от предыдущего, но и от последующего состояния, а также от наблюдаемого состояния.

4. Метод с использованием нейросетей

Данный метод выделения и классификации сущностей использует в качестве знаний для выделения именованных сущностей информацию из Википедии и DBPedia. Сам алгоритм комбинирует правила, задаваемые регулярными выражениями и модель взвешенного персептрона, обученный на текстах Википедии. [26]

Несмотря на закрытость внутренней структуры алгоритма, разработчики утверждают, что алгоритм показывает высокую точность для семи естественных языков и 15 классов именованных сущностей.

В общем случае все алгоритмы данной группы требуют при себе решения проблемы определения референтов, необходимых нам для построения семантической сети, что существенно усложняет разработку метода. Более того, алгоритмы, основанные на машинном обучении требуют точных и качественных данных для обучения, что не всегда возможно предоставить. Наконец, обучение требует достаточно большого количества времени, что также сказывается на работе методов.

III. Алгоритмы выделения терминов на основании различных текстов

Алгоритмы данной группы позволяют выделять термины различного уровня (однословные, многословные) на основании текстов, не имеющих специальной разметки. Рассмотрим некоторые из них.

1. Метод извлечения информации из неструктурированных текстов

В основе данного метода итеративное выполнение функций по структуризации исходных текстов, формированию набора предикатов и объектов, а также поиска и описания фактов. Использует набор правил, на основании которых происходит выделение сущностей и отношений между сущностями. При этом выделяют 5 видов сущностей и два вида отношений между сущностями. Алгоритм позволяет строить эффективные аннотации на основе предложенных ему текстов. [27]

Важным преимуществом данного алгоритма является его способность обрабатывать русскоязычные тексты без потери эффективности. Однако, это достигается засчет огромной предварительной работы по составлению правил выделения сущностей, что не очень удобно.

2. Метод извлечения низкочастотных терминов из специализированных текстов

Исходя из названия, алгоритм опирается на заранее подготовленный корпус текстов, состоящий из словарных статей, обработка которого осуществляется в три этапа [28]:

- кластеризация текста;
- выделение наиболее частотных классов для каждого кластера;
- формирование списка терминов из слов, принадлежащих данным классам;

В качестве терминов выделялись однословные термины, а в качестве метода кластеризации используется метод Уорда, считающийся наиболее точным среди остальных методов [28].

Неоспоримым преимуществом данного метода является способность выделения даже тех терминов, которые редко встречаются в тексте (менее 3-х раз). Более того, метод не требует при себе обучающей выборки, так как реализует механизм кластеризации исходных текстов. Также достоинством алгоритма является способность выделять общие однословные термины с высокой степенью точности [28]. Однако, для работы этого алгоритма необходимо решить проблему подготовки специализированного текста для выделения терминов.

1.4.2. Анализ методов выделения связей между сущностями

Алгоритмы выделения связей между сущностями представлены меньшим числом, так как из-за особенностей русского языка, такой связью зачастую служит предикат, выраженный глаголом, который требует при себе существительные в определенной форме. Рассмотрим некоторые методы выделения связей помимо представленного в разделе 4.1.3. метода.

1. Простой метод выделения соотношений.

Алгоритм строится из соображения о том, что объекты-сущности относятся к ближайшим к ним глаголам. Такое утверждение не всегда верно, но, как утверждает разработчик, зачастую является правдой. При этом задача выделения отношений сводится к задаче аннотирования последовательностей, описанная в [29].

Основным преимуществом данного метода является простота его реализации засчет имеющегося API. При этом время составления обучающей выборки составляет примерно 4 часа. Однако, ввиду отсутствия качественной оценки эффективности выделения связей, говорить о надежности алгоритма не представляется возможным.

2. Методы ориентированные на топологию и ядро.

В основе данных методов лежит алгоритм извлечения фиксированного множества дифференциальных признаков с последующим запуском классификатора, в качестве которого может выступать, например, дерево принятия решений. Один из таких методов более подробно описан в [30].

Методы, ориентированные на ядро представляют собой системы из специальных ядер для перехвата сходств между структурами, такие как дерево или граф. В качестве ядра обычно выступают ключевые слова.

1.5. Формализация постановки задачи

На основании проведенного анализа предметной области, а также существующих методов и алгоритмов, постановку задачи можно сформулировать следующим образом:

В рамках работы над проектом необходимо реализовать систему обработки естественно-языковых запросов к базам данных, которая использует семантическую сеть в качестве способа представления знаний в системе. Входными данными для этой системы являются строки запросов на выбранном естественном языке. Ограничения на запросы определяются разработчиком системы. Выходными данными являются запросы к реляционной базе данных, синтаксически корректные с точки зрения языка работы с базой данных.

Используемая семантическая сеть, в соответствии с техническим заданием, должна генерироваться автоматически. Таким образом, необходимо разработать метод, осуществляющий данную генерацию. Разрабатываемый метод предполагает наличие выборки текстов для формирования сети.

Так как техническое задание предполагает разработку системы, необходимо реализовать программный продукт, демонстрирующий работу системы в самом простом варианте. Провести исследования работы реализуемых методов и системы в целом.

2. Конструкторский раздел

2.1. Построение семантической сети

Основываясь на результатах, описанных в аналитической части, мы можем сделать выводы о том, каким образом формализуется задача построения семантической сети в общем смысле. Для автоматического построения семантической сети необходимо выделить основные компоненты – понятия (термины, сущности), свойства(атрибуты), а также связи между сущностями. Выделение основных компонент осуществляется на основе входной выборки текстов с использованием различных признаков, позволяющих утверждать о том, чем в тексте является конкретная лексема – термином, связью или атрибутом. Также при построении онтологий и семантических сетей зачастую упоминают о выделении связей нескольких уровней вложенности (связи между связями), однако мы в своей работе будем считать, что строящаяся семантическая сеть состоит из простых связей между сущностей.

Еще одним фактом, который стоит учесть при автоматической генерации семантической сети, является то, что запросы к данной сети впоследствии будут транслироваться в запросы к реляционной базе данных. Однако, большинство современных реляционных баз данных поддерживают интерфейс запросов на английском языке. Таким образом, может возникнуть проблема трансляции выделенных компонент на английский язык с целью удобства и корректности трансляции естественно-языковых запросов в запросы к базе данных.

После того, как все компоненты семантической сети выделены и транслированы на английский язык, необходимо сформировать ее путем описания на языке, поддерживающем описание различных баз знаний, каким могут выступать, например, языки логического программирования.

На основании всего вышеизложенного, можно сформировать такую последовательность действий при автоматической генерации семантической сети (рисунок 4):

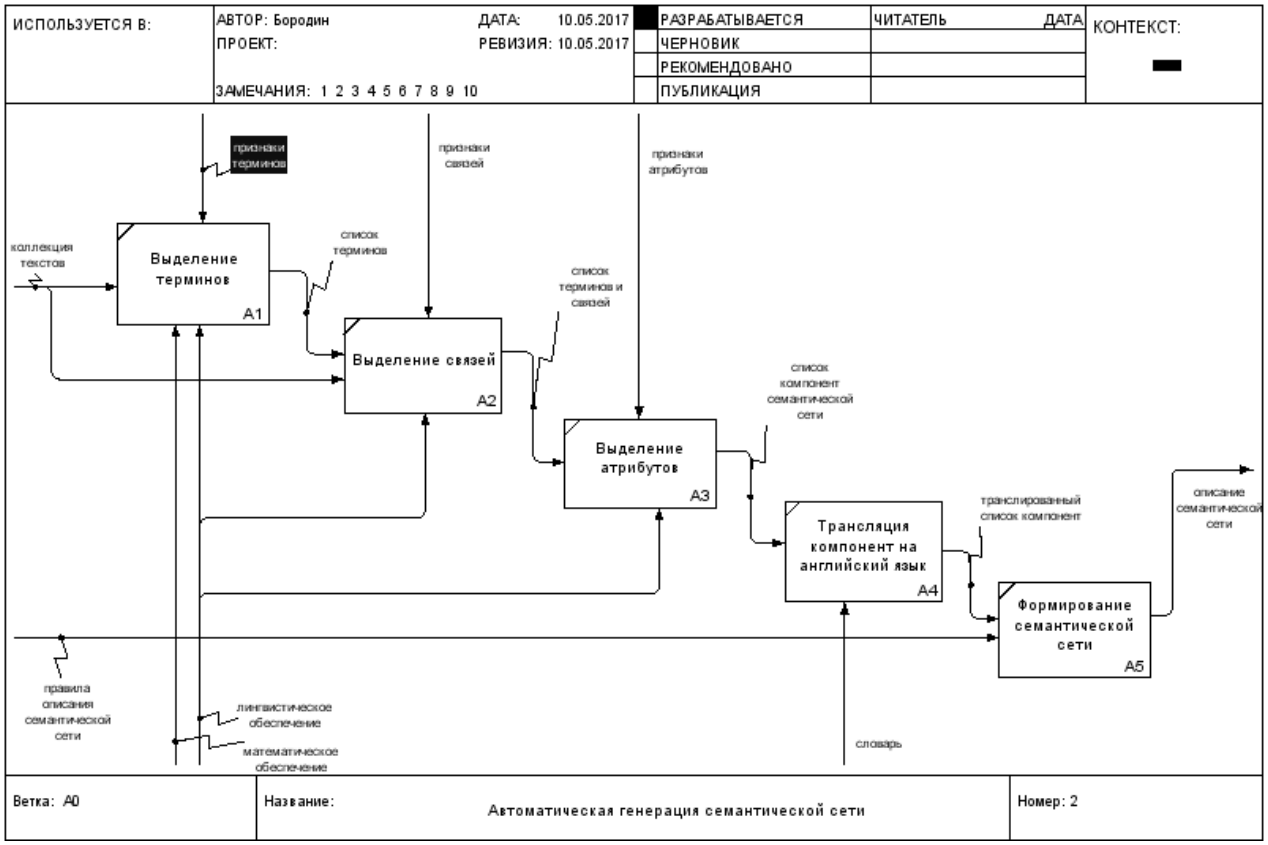


Рисунок 4. Алгоритм автоматической генерации семантической сети

2.1.1. Извлечение набора сущностей для семантической сети

Задача выделения терминов предусматривает следующий набор вопросов, требующих решения:

1. Что понимать под терминами, которые выделяют из текстов? (формализация задачи)

2. Откуда выделять термины – обучающая выборка, произвольные тексты? (проблема входных данных)

3. Как определить на основе входных данных требуемые для выбранной предметной области термины? (проблема нахождения семантической близости слов).

В своей работе мы основываемся на том, что реализуемая семантическая сеть должна отражать реальное описание базы данных, к которой будут составляться запросы на естественном языке. Как известно, заголовок таблицы реляционных баз данных обычно представляет собой однословный термин, записанный на английском языке, и при этом сама таблица отражает сущности

реального мира. Поэтому в качестве выделяемых терминов мы будем понимать однословные термины, выраженные существительными и связанные с выбранной предметной областью.

Для решения второй подзадачи, связанной с задачей выделения терминов, обратимся к результатам проделанной работы в аналитической части. Как видно из результатов, наиболее хорошо с задачей выделения однословных терминов справляется метод, основанный на работе со структурированными специализированными текстами, описанный в [28]. Исходя из этих данных, решено использовать коллекцию текстов, связанных с киноиндустрией. При этом используемые тексты должны быть структурированными, поэтому было принято решение использовать словарные статьи на заданную тематику. При этом, для того, чтобы процесс выделения терминов был достаточно быстрым с точки зрения времени, принято решение использовать небольшую по размерам коллекцию статей (около 500 статей).

Так как в обычных текстах существует довольно большое количество информации, никак не связанной с необходимыми в нашей задаче терминами, для входной коллекции словарных статей требуется подготовка образов документов (под документом будем понимать одну словарную статью из коллекции документов). Для этого необходимо убрать все слова, несвязанные с целевыми: служебные части речи (предлоги, союзы и т.п.), а также все самостоятельные части речи, не являющиеся существительными.

Далее требуется определить, каким образом разрабатываемый алгоритм выделения однословных терминов будет распознавать близкие понятия на заданную тематику. Так как разрабатываемый метод получает в качестве входных данных неизвестную выборку текстовых документов, о которой нет дополнительной информации, удобно использовать положения дистрибутивной семантики [31].

Основным положением, позволяющим использовать методы данной области компьютерной лингвистики, заключается в следующем: «смысл слова является распределением над его контекстами». Иными словами, имея выборку

из текстовых документов, можно посчитать значения каких-либо счетчиков, которые позволят определить, являются ли слова, встречающиеся в тексте, семантически близкими по смыслу. При этом обычно сталкиваются с двумя видами совстречаемостей [31]:

1. ***Совстречаемость первого порядка (связанность)*** – например, кран и вода – это такие слова, у которых высокая совместная встречаемость в текстах, но при этом они не обязательно могут быть синонимами или относиться к какой-либо предметной области

2. ***Совстречаемость второго порядка (схожесть)*** – например, кран и смеситель – это такие слова, у которых высокая совстречаемость, но при этом они с высокой долей вероятности относятся к какой-либо выделенной категории.

Так как в нашей работе мы концентрируемся на поиске однословных терминов из коллекции, связанных с заданной предметной областью, то, очевидно, необходимо искать совстречаемости второго порядка. При этом, согласно [32], для каждого слова, выделенного в тексте, строится вектор совстречаемостей, состоящий из значений счетчиков. Далее происходит сравнение пар векторов: если они совпадают с заданной точностью, то можно говорить с высокой степенью вероятности о семантической близости двух слов. Следует отметить, что векторы совстречаемостей строятся в зависимости от выбранной тематической модели [31] (термин-термин, термин –документ, термин-контекст). Поставленная перед нами задача, а также выбранные входные данные говорят нам о том, что разумно использовать тематическую модель термин-документ, в которой вектор строится для каждого слова из всех документов, а каждый элемент вектора – значение конкретного счетчика в конкретном документе.

Отдельного внимания при разработке метода выделения однословных терминов из коллекции текстовых документов требует выбор счетчика, который будет составлять элементы составляемых векторов. В дистрибутивной семантике существуют следующие основные виды счетчиков [31]:

1. ***Количество встречаемостей слова*** в тексте (обычный подсчет вхождений);

2. ***Счетчик совстречаемостей слов*** в текстовых документах – формируется путем обычного подсчета совместной встречаемости слов в тексте в пределах заданного окна в каждом предложении.

3. ***Счетчик совместной информации*** (mutual information) – отношение совместной встречаемости двух слов в конкретном текстовом документе к произведению отдельных встречаемостей каждого слова в тексте, выраженное формулой:

$$MI = \log(\frac{n_{uv} n}{n_u n_v}),$$

где n – число документов.

Так как в своей работе мы используем тематическую модель термин-документ, а также принимая во внимание тот факт, что мы используем небольшую коллекцию текстовых документов, удобно подсчитывать обычное число вхождений. Такой счетчик не требует значительных вычислений, а также может быть модифицирован для удобства подсчета засчет умножения на коэффициент (например, на инвертированное число документов в выборке).

После того, как для каждого слова сформирован вектор совстречаемости для модели термин-документ, получается матрица размерности число_слов*число_документов. При этом с высокой долей вероятности эта матрица будет разрежена, поскольку в большинстве случаев многие существительные, выделенные на этапе подготовки образов текста, не будут появляться в коллекции документов. Возникает проблема уменьшения размерности такой матрицы. В нашей работе было принято решение уменьшать размерность матрицы с помощью сингулярно-векторного разложения, постулаты которого говорят нам о том, что любая прямоугольная матрица P может быть декомпозирована на три матрицы, причем все матрицы связаны между собой уравнением:

$P = UDV^T$, где U и V – матрицы, состоящие из ортонормированных сингулярных векторов ($U^T U = V^T V = E$), а D – квадратная матрица,

содержащая на своей главной диагонали положительные сингулярные числа в порядке убывания [33]. С этими числами связана задача оптимизации, заключающаяся в выборе размерности матрицы D, которая в совокупности с преобразованными размерностями матрицы U и V давала бы наиболее приближенное описание исходной матрицы. Однако, решение задачи оптимизации выходит за рамки нашей работы, поэтому мы ограничимся двумерным разложением исходной матрицы – это позволит наглядно отобразить результаты выделения однословных терминов.

Наконец, одним из наиболее важных вопросов в выделении однословных терминов является оценка семантической близости слов на основании имеющихся векторов. Согласно [34], существуют следующие меры семантической близости различных слов:

1. ***Меры, основанные на расстояниях*** (Евклидово расстояние, расстояние Чебышева и другие);
2. ***Меры сходства***, основанные на принципах тождественности, неотрицательности и симметричности;
3. ***Угловые меры***, основанные на предположении о близости слов в зависимости от расстояния между многомерными векторами (косинусная мера);
4. ***Корреляционные меры***, основанные на предположениях из математической статистики (мера Жаккара, мера Пирсона и другие);

При этом косинусная мера является наиболее удобной для нас с точки зрения быстродействия алгоритма и получения результата, а также из-за имеющихся у нее преимуществ (независимость от длин векторов, способность выражать различие и сходство между терминами [34]). Важно также отметить, что так как мы знаем предметную область, в которой находятся искомые термины, мы будем находить семантическую близость к термину, заранее известному, как принадлежащему данному классу терминов – это позволит упростить задач классификации терминов.

Таким образом, разработанный алгоритм выделения однословных терминов-сущностей представляет собой следующую последовательность действий (рисунок 5):

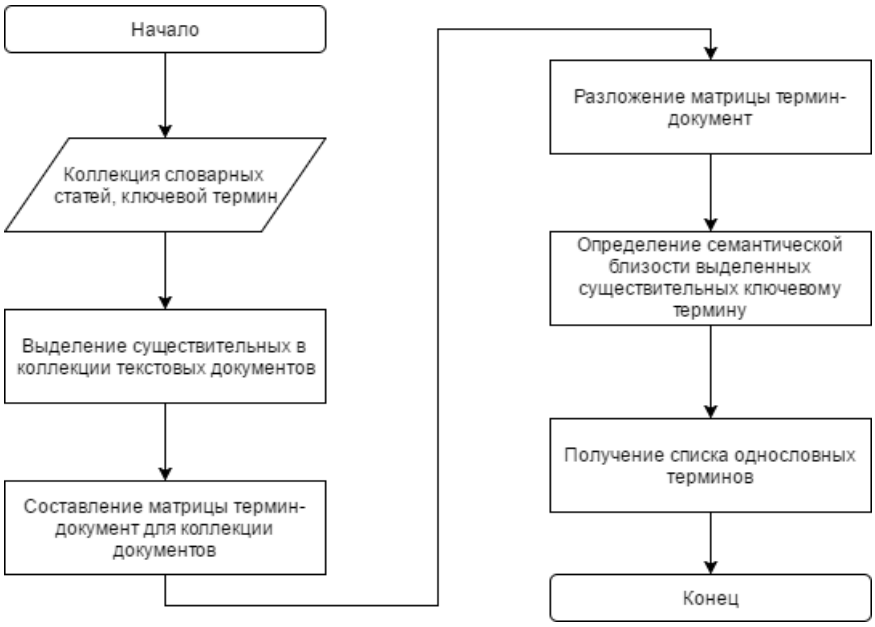


Рисунок 5. Алгоритм выделения сущностей семантической сети

Данный алгоритм может быть впоследствии модифицирован на предмет устранения лишних позиций в списке выделенных терминов засчет составления синсетов – наборов синонимичных терминов с последующим распознаванием их в качестве единого термина.

2.1.2. Извлечение связей между сущностями

Для того, чтобы определить, как связаны термины-сущности из списка выделенных сущностей, обратимся к грамматике русского языка, которая утверждает, что существует три вида синтаксических связей в предложении: согласование (например, зеленая аллея), управление (вижу друга) и примыкание (хорошо рассказывать) [35]. Эти связи характеризуют лингвистические отношения, описанные в аналитическом разделе (глагольные – управление, атрибутивные и падежные - согласование).

Данные синтаксические связи имеют четко выраженную структуру, например, синтаксическая связь «управление» требует при себе схемы словосочетания, состоящего из глагола и зависимого существительного в определенном роде, числе и падеже.

Основываясь на том, что в своей работе мы концентрируемся на глагольных отношениях (связи между сущностями выражены глаголами) и при этом перед нами не стоит задачи выделения отношений между отношениями, принято решение воспользоваться поиском таких глаголов, которые бы выражали связь между терминами из выделенного списка. При этом мы будем говорить о том, что данный глагол выражает связь между двумя терминами, если эти два термина расположены в окрестности данного глагола (схема **ОБЪЕКТ-ПРЕДИКАТ-ОБЪЕКТ** со всеми ее вариациями).

Таким образом, для того, чтобы автоматически выделить связи между терминами, воспользуемся следующим алгоритмом (принимаем во внимание предварительную очистку от лишних слов подобно алгоритму выделения сущностей, но учитывая при этом все глаголы в коллекции текстовых документов) (рисунок 6):

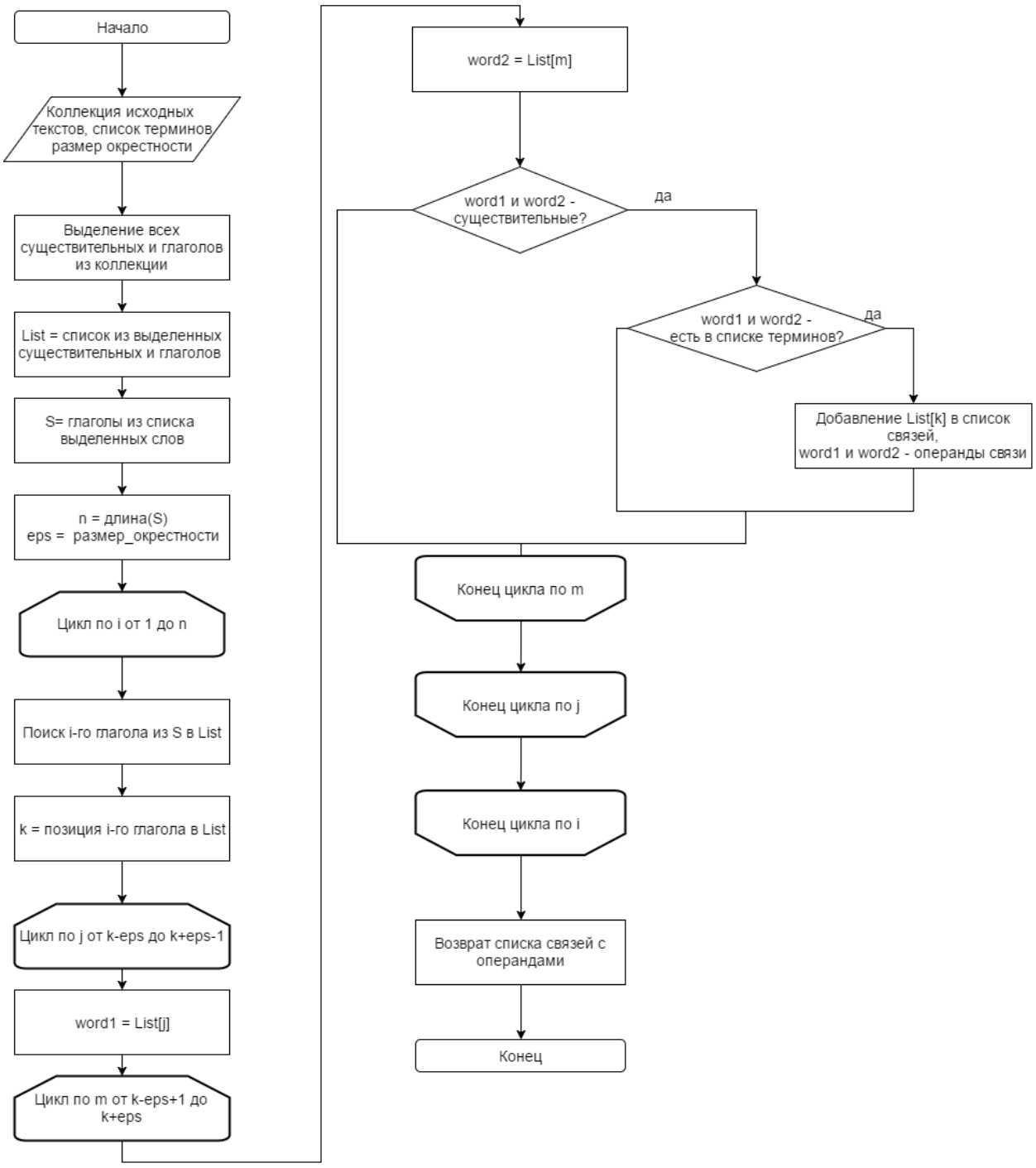


Рисунок 6. Составление списка связей между сущностями

При этом следует отметить следующую особенность алгоритма – в формируемый список существительных и глаголов слова помещаются в порядке появления. Это делается для того, чтобы можно было определять связи между терминами по контексту.

2.1.3. Выделение атрибутов для сущностей

Задача нахождения атрибутов для сущностей в общем случае может сводиться к нахождению атрибутивных отношений между словами в предложениях. При этом в случае использования конкретной выборки, направленной на более полное извлечение терминов для строящейся автоматически семантической сети, с высокой долей вероятности можно говорить о том, что такая выборка не будет обладать полным набором всех существующих атрибутов для конкретной сущности.

Исходя из предположения о том, что для конкретного термина может существовать достаточно большое число атрибутов, было принято решение о том, чтобы загружать набор атрибутов для известных сущностей в рамках данной предметной области из уже готовых источников. Данное решение может быть осуществлено засчет того, что термины, полученные на этапе выделения сущностей, будут переведены для корректной трансляции естественно-языковых запросов в запросы к базе данных. Таким образом, мы предполагаем повысить качество составляемой сети и уменьшить потери. Такое упрощение позволяет нам снять вопрос о выделении атрибутов для сущностей.

В заключении следует отметить, что генерируемая семантическая сеть будет обладать значительным количеством сущностей, связей между сущностями и атрибутами. Такая обширная сеть затруднит отладку реализуемой системы, поэтому принято решение ограничить генерируемую семантическую сеть и, соответственно, базу данных, ограниченным набором сущностей, полученных с помощью разработанного алгоритма.

2.2. Конструирование реляционной базы данных и семантической сети

При конструировании базы данных следует учитывать, что в нашей работе не осуществляется семантический анализ текста, поэтому выбор предметной области жестко определяет контекст работы разрабатываемой программы. При этом, следует также выбрать такую предметную область, в которой существует множество различных отношений между объектами.

Поэтому, в качестве предметной области было решено выбрать киноиндустрию, а в качестве сущностей актеров, режиссеров, фильмы и награды. Такой выбор сущностей обусловлен частотой запросов по таким ключевым словам (согласно [36] сущность **«фильм»** имеет частоту запросов 178525926 просмотров в месяц, **«актер»** - 13253923 в месяц, **«режиссер»** - 615083 в месяц, **«награда»** - 515170). Такая популярность запросов позволяет сконструировать «полезную» для конечного пользователя систему, при этом будет обеспечено удобство и скорость отладки такой системы. Поэтому в разработке прототипа системы преобразования естественно-языковых запросов принято решение пользоваться этими сущностями и относящимися к ним связями.

Между данными сущностями существует достаточно большое количество различных связей, что удовлетворяет нашим требованиям. На рисунке 7 можно увидеть структуры разработанной базы данных.

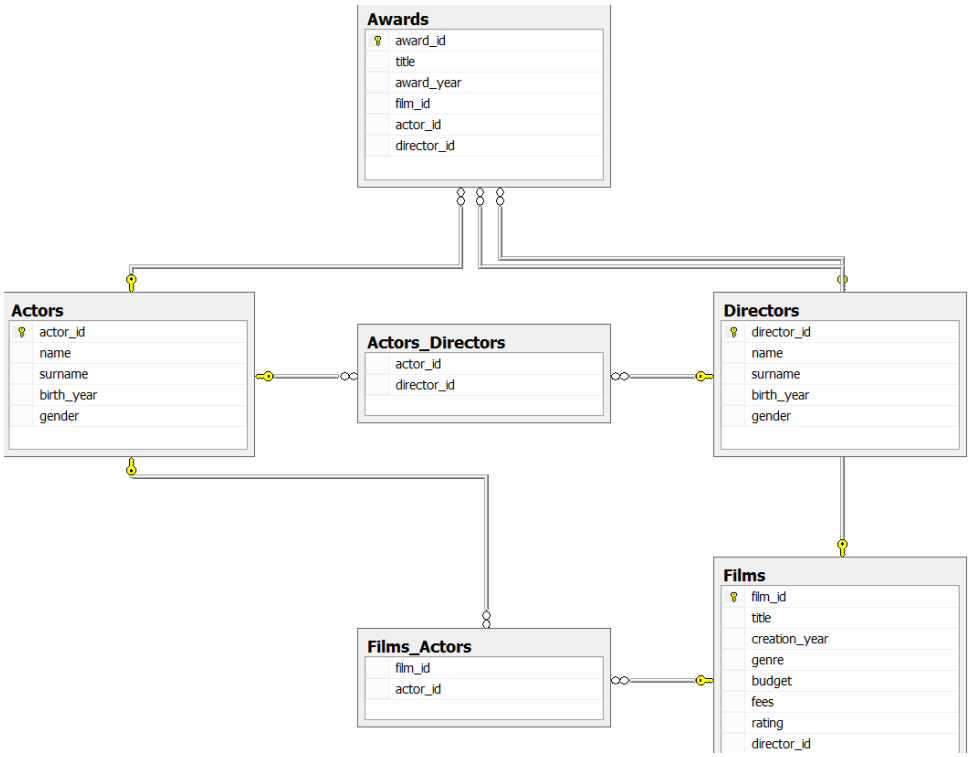


Рисунок 7. ER-диаграмма разработанной базы данных.

Примечание: так как перед нами стоит задача трансляции естественно-языковых запросов к базе данных, в разрабатываемом прототипе неважно

качество наполнения разработанной базы данных. Следовательно, мы можем ограничиться тестовыми записями в процессе отладки системы.

Следует отметить, что в работе используются таблицы **Films**, **Actors**, **Directors**, **Awards**, в то время как таблицы Films_Actors, Actors_Directors используются для реализации связи многие к одному и многие ко многим, но запросы к этим таблицам не составляются, так как они не имеют смысловой нагрузки с точки зрения запросов.

Рассмотрим отдельно каждую из таблиц разработанной базы данных. На каждое из полей таблиц накладываются ограничения, которые позволят в перспективе обработать некоторые исключения. Так, в таблицах Actors, Directors и Awards год рождения/присуждения ограничен сверху 2019 годом, а пол ограничен символами “m” или “f”. В таблице Films накладываются ограничения на год создания (возможный год с 1920 по 2016), бюджет и сборы (натуральное число + ноль), а также рейтинг (рациональное число от 0 до 10). Все идентификаторы – целые положительные числа, идентификаторы уникальны.

Таким образом, на основании описанных выше фактов, учитываемых при составлении базы данных, мы можем сделать вывод о том, в какой нормальной форме находится наша таблица. Предполагается, что строки в наших таблицах будут различны (это обеспечивается уникальными ключами), а ячейки в таблице имеют атомарную, неделимую структуру. Следовательно, разработанная база данных находится в первой нормальной форме. Однако, таблицы, используемые в работе, удовлетворяют условиям соответствия второй нормальной формы – нахождение в первой нормальной форме, а также любое поле таблицы, не входящее в состав первичного ключа, функционально полно зависит от первичного ключа. Иными словами, если таблицы приведены к первой нормальной форме и у них установлен уникальный идентификатор для каждой строки, то они находятся во второй нормальной форме.

Отдельно стоит упомянуть про таблицу фильмов. Эта таблица находится в третьей нормальной форме, т.к. имеется разделение таблиц актеров и

режиссеров в две таблицы. Разделение обусловлено тем, что один режиссер может снять несколько различных фильмов. Таким образом, при удалении фильмов из таблиц, сведения о режиссерах останутся. Аналогичным образом организовано разделение таблицы наград и остальных таблиц. А так как в таблицах актеров и режиссеров нет полей, не зависящих от ключа (человек с определенным идентификатором имеет собственные характеристики), то и эти таблицы находятся в третьей нормальной форме. Следовательно, наша база данных находится в третьей нормальной форме. Говорить о БКНФ применительно к нашей базе данных не представляется возможным, т.к. потенциальные ключи не вынесены в отдельные таблицы. Строки в некоторых таблицах зависимы друг от друга (наличие строки X в таблице фильмов предполагает наличие строки Y в таблице режиссеров). Исходя из вышеописанного, можно сделать вывод, что база данных не находится в четвертой нормальной форме, а значит, и дальнейший анализ нормальных форм бессмысленен [37].

Также, исходя из структуры описанной системы, следует, что для обработки запросов на естественном языке удобно использовать семантическую сеть. Главное ее отличие от реляционной базы данных в том, что основной упор в ней делается на отношения между сущностями, а не на сами сущности, как это делается в реляционных базах данных. Семантическая сеть представляет собой граф, в котором в качестве узлов являются сущности, а в качестве связей используются отношения между сущностями. Пример возможной семантической сети для заданного набора сущность приведен на рисунке 8:

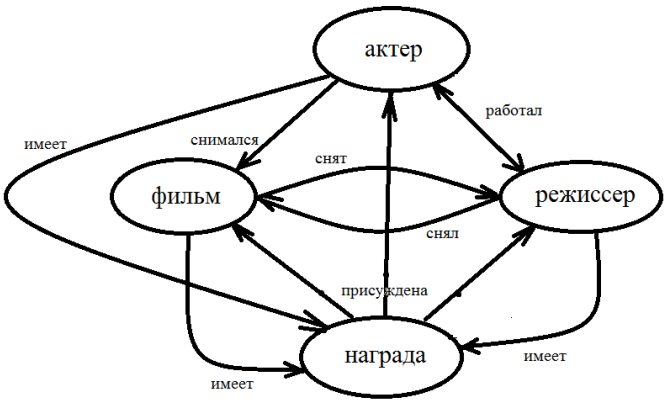


Рисунок 8. Семантическая сеть для выбранных сущностей.

Условно, трактовать отношения в данной сети можно так: какой-то актер может иметь награду, сняться в фильме и поработать с режиссером. Эти отношения очень удобно описывать средствами языков логического программирования. Например, на Prolog отношение между сущностями будет описано так: ***снимался(актер, фильм)*** и т.п.

Для удобства трансляции естественно-языковых запросов семантическая сеть переведена на английский язык вручную.

Представление данных в базе данных и семантической сети различны. В реляционной базе данных таблица представляется своеобразной структурой, где каждое поле является атрибутом или ключом. Семантическая сеть представляет собой базу знаний, а в базе знаний существуют лишь факты и правила. Таким образом, семантическая сеть представляется набором фактов, где главный функтор характеризует отношение между атрибутами (например, актер(Вася, Мышкин, 1972, м)).

2.3. Обработка запросов на ограниченном естественном языке

Обработка естественно-языковых запросов, как уже было описано ранее, осуществляется в три этапа: сначала исходный текст запроса проходит через лингвистический модуль, формируется запрос к базе знаний на Прологе и далее формируется запрос на выборку в зависимости от полученного запроса. Останавливаясь подробнее на описании первого этапа обработки естественно-языкового запроса, следует отметить, что написание собственного

лингвистического модуля – очень сложный и трудоемкий процесс, который осуществляется командой разработчиков в течение нескольких лет, что явно выходит за рамки рассмотрения нашей работы. Поэтому принято решение об использовании в работе уже имеющихся технических средств, позволяющих осуществить требуемую работу на данном этапе. Условно, первый этап, получение запроса к базе знаний, может быть описан следующим алгоритмом (рисунок 9):

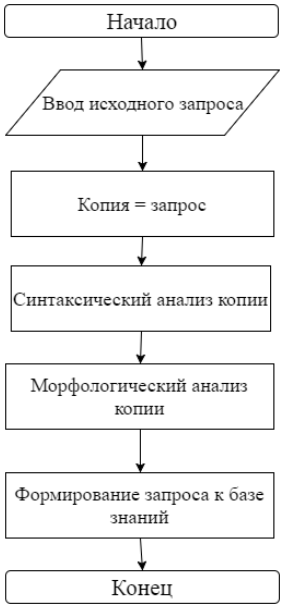


Рисунок 9. Схема алгоритма работы лингвистического модуля.

В работе алгоритма делается копия исходного запроса с целью обеспечения проведения нескольких видов анализа за один прогон модуля независимо друг от друга. Поэтому для нас не имеет особого значения строгий порядок выполнения различных видов анализа. Следует отметить, что в данной версии алгоритма опускается графематический анализ текста, направленный на выделение словоформ в предложении. Это связано с ограничениями на естественно-языковые запросы, о которых будет рассказано далее.

Также стоит отдельно упомянуть принципы работы каждого из оставшихся видов анализа применительно к нашей задаче. Синтаксический анализатор, принимая текст исходного запроса, строит синтаксическое дерево зависимостей, в котором определены отношения между частями предложения. Так как русский язык обладает большой степенью омонимии, результатом

синтаксического разбора в зависимости от контекста могут быть несколько деревьев [38].

На этапе морфологического анализа проверяется валидность входящих в предложение словоформ посредством поиска в имеющемся словаре, формируется список словарных статей с информацией о каждой словоформе, затем этот список возвращается на верхний уровень работы программы. Данный список словоформ нужен для корректного формирования запроса к базе знаний.

Этап формирования запроса к базе знаний может осуществляться несколькими способами, о которых будет рассказано далее. В качестве результата работы этой части алгоритма планируется получать текст запроса к базе знаний и список инициализированных параметров.

После того, как произведен этап преобразования запроса на естественном языке, возникает проблема приведения полученных данных к конечному результату – запросу на формальном языке. С учетом модели реализуемой базы данных, имеется два пути обработки полученного синтаксического дерева:

1) с помощью ***семантических сетей***, отражающих связи между объектами, а также связи между связями;

2) с помощью ***средств логического программирования***;

Первый способ очень удобен при реализации запросов к графовым базам данных, так как под семантической сетью понимается граф информационной модели, который может быть переложен на базу данных. Второй способ пригоден для работы как с графовыми базами данных, так и с реляционными базами данных, так как средства логического программирования позволяют представить синтаксическое дерево как в виде семантической сети, так и в виде различных таблиц, свойств и атрибутов, а также связей между таблицами. Более того, анализ доступных информационных источников показал, что на данный момент имеются весьма эффективные трансляторы из языка логического программирования в формальный язык работы с базами данных, что существенно упрощает процесс составления запросов [39]. Однако,

представленный в данной работе транслятор имеет ряд ограничений, которые впоследствии могут быть отброшены при соответствующей модификации.

Стоит также отметить, что использование языка логического программирования в качестве промежуточного средства преобразования запросов необходимо, так как преобразование с его помощью дает простор для последующей модификации.

С учетом описанных выше фактов и достоинств, можно сказать, что использование языка логического программирования применительно к данной задаче весьма оправдано.

Таким образом, можно сделать вывод о том, что мы описали два равновозможных в использовании метода преобразования получаемой после анализа предложения на естественном языке информации, которые условно могут быть представлены следующей схемой (рисунок 10):

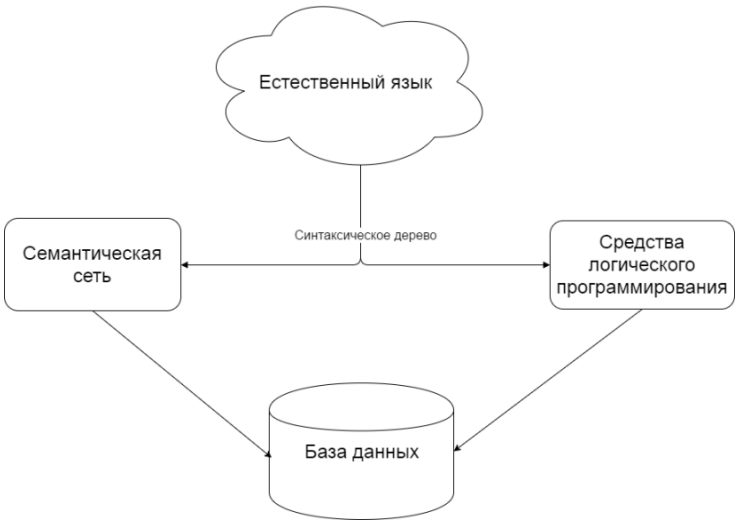


Рисунок 10. Условная схема методов получения запросов к базам данных.

При этом в своей работе мы отмечаем, что основная реализация будет вестись с использованием методологии логического программирования.

2.4. Формирование запросов к базе данных

2.4.1. Запросы на естественном языке

Условно, формирование запросов к базе данных можно разделить на три этапа в соответствии с выбранной структурой системы:

- обработка исходного естественно-языкового запроса;
- преобразование в запрос на языке логического программирования;
- транслирование запроса на языке логического программирования в запрос к реляционной базе данных;

При формировании запросов к нашей базе данных, мы сталкиваемся с проблемой произвольности порядка слов в предложении на выбранном естественном языке. Так, в английском языке мы можем наблюдать очень строгую последовательность слов (например, вопрос на английском языке всегда начинается со вспомогательного слова – «Who are you?», «How old are you?» и другие). Такой строгий порядок позволяет в достаточно четкой и корректной форме приводить имеющиеся естественно-языковые запросы к удобному для работы программы виду. Однако, выбранный нами естественный язык проявляет гибкость в отношении порядка слов в предложении без нарушения смысловой нагрузки («Я люблю тебя.», «Тебя я люблю.» и «Люблю я тебя.» - конструкции, которые могут в равной степени существовать в тексте). Поэтому, так как в данной работе опускается семантический анализ предложений на естественном языке, было принято решение ограничить рамки вводимых запросов различными «схемами» предложений, которые позволят добавить предсказуемости в трансляции запросов к конечной базе данных и сделать корректное преобразование. Под «схемой» предложения понимается определенный порядок слов в предложении, наиболее часто употребляемый в повседневной речи.

Для того, чтобы определить, какие именно конструкции в речи наиболее часто используются пользователями, было решено провести социологический опрос. Контрольная группа составляла 20 человек. Такое число респондентов позволяет с определенной вероятностью говорить о каких-либо фактах. Немногочисленность числа респондентов обусловлена малой степенью заинтересованности в развитии исследования.

Группа откликнувшихся респондентов состояла из людей различных профессий – программистов, психологов, работников авиационной промышленности и некоторых других областей. Такой состав участников, а именно вариативность сфер деятельности, в которых задействованы респонденты, обусловлена тем, чтобы получить максимально эффективную выборку результатов опроса.

Возрастной состав участников составляет преимущественно 18-22 года, так как эти люди наиболее подкованы в предметной области с точки зрения современных сведений и тенденций выбранной предметной области, но также в полученной выборке имеются результаты и людей более солидного возраста.

Половой состав участников был поддерживаемым на уровне «50 на 50» - 50% представителей обоих полов, однако есть небольшие отклонения от этого правила в результирующей выборке, которые не должны существенно повлиять на эффективность получаемых результатов.

Также с целью корректности и эффективности получения интересующей нас выборки, от респондентов скрывалась истинная цель получения результатов, но была предложена схожая цель. Так, в полученных респондентами анкетах, были поставленные исходные данные, приближенные к решению нашей задачи, описывалась структура имеющейся базы данных, но в анкете она интерпретировалась так, что эта база данных заложена внутрь некого робота-переговорщика, с которым можно напрямую пообщаться на произвольные темы. Однако, было поставлено ограничение, которое разрешало респондентам общаться строго в рамках выбранной при решении нашей задачи предметной области – было сказано, что робот может общаться только на те темы, которые знает. Далее, респондентам было предложено задать все возможные вопросы, которые их интересуют, но с учетом поставленных ограничений. Все участвовавших респонденты остались анонимными для того, чтобы раскрепостить их в общении с «роботом». Пример такой анкеты приведен в приложении А.

После того, как все респонденты ответили на поставленный в анкете вопрос, был проведен их анализ. Под анализом понимается синтаксический разбор всех предложений в полученной анкете. Данный анализ был выполнен вручную для получения корректных результатов и контроля обработки. После проведения анализа были выделены следующие наиболее часто употребляемые синтаксические конструкции:

- **дополнение дополнение сказуемое подлежащее** (пример: какие награды получил Аль Пачино?);
- **сказуемое дополнение дополнение** (пример: покажи фильмы Стивена Спилберга.);
- **дополнение подлежащее сказуемое дополнение** (пример: в каком году Леонардо ДиКаприо получил Оскар?);

Также в ходе анализа полученных результатов были получены разнообразные конструкции, содержащие причастные обороты, но они нами не рассматриваются в силу ограничений, описанных в работе.

Таким образом, на данном этапе работы мы ограничиваемся такими конструкциями в обработке запросов. При этом стоит отметить, что для реализации прототипа разрабатываемой системы достаточно использовать одну схему на данном этапе работы. С учетом архитектуры программы, последующее добавление конструкций предложений для проверки не должно вызывать затруднений. Таким образом, для отладки работы разрабатываемого транслятора используем схему «**СКАЗУЕМОЕ – ДОПОЛНЕНИЕ - ДОПОЛНЕНИЕ**»

2.4.2. Правила обработки естественно-языковых запросов

Получив в ходе предыдущего этапа различные классы вопросов к разрабатываемой системе, возникает проблема их обработки. Как получить необходимое корректное преобразование естественного языкового запроса? Для решения этой задачи можно воспользоваться двумя методами:

- осуществление парсинга получаемого в ходе работы лингвистического модуля синтаксического дерева с помощью **алгоритма обратной польской записи;**
- **позиционный парсинг** исходного предложения с выделением сущностей, отношений, атрибутов и прочей полезной для запросов к базе данных;

Особенностью первого метода является то, что при применении этого метода прослеживается четкая аналогия с решением различных алгебраических уравнений – использование стека в алгоритме обратной польской записи позволяет отделить операторы и операнды, основываясь при этом на их приоритете и ассоциативности. Однако в нашей задаче в качестве операндов выступают значения сущностей, а в качестве операторов – названия отношений. Например, во время работы алгоритма обратной польской записи применительно к математике можно получить такой результат: $a + b = ab +$. Воспользовавшись алгоритмом обратной польской записи в нашей предметной области и определив соответствующие отношения можно получить такой результат: **«Вася снимался в Матрице» => снимался(Вася, Матрица).**

Такой конечный запрос получается с соблюдением синтаксических норм языка логического программирования (в частности, Prolog). Однако, для того, чтобы получить такой результат, недостаточно одного лишь алгоритма обратной польской записи – необходимо средство перевода, которое позволяло бы учитывать необходимые синтаксические нормы языков программирования. В качестве такого средства было решено выбрать определенного вида словарь, который необходим для:

1. определения характеристик слов, так как лингвистические анализаторы работают достаточно долго и обращаться всякий раз к морфологическому анализу невыгодно с точки зрения времени обработки запроса;

2. перевода функционально значимых единиц на английский язык. Перевод на английский язык необходим для генерации SQL-запроса, так как SQL-

запросы не могут содержать названий базы данных, таблиц, столбцов на русском языке.

Примечание: так как мы ограничиваемся конкретным набором сущностей (актер, режиссер, фильм, награда), а также принимая во внимание тот факт, что при разработке прототипа системы мы концентрируемся на одном классе запросов, то словарь можно хранить внутри системы. В конечном итоге предполагается включить в проект таблицы базы данных, имитирующие работу словаря.

Он имеет следующую структуру:

Слово-форма	Тип	Функция	Способ-ность	М.	Заменяемый текст
Аль	СУЩЕСТВИТЕЛЬНОЕ	Подл.	Константа	0	‘Al’

Таблица 2. Структура словаря для преобразования запросов и пример использования.

Под ***словоформой*** понимается сама исходная словоформа, какой она предстает в исходном естественно-языковом запросе.

Тип позволяет узнать, является ли исходная словоформа существительным, прилагательным, глаголом, числом, местоимением. Данное поле называется именно типом, потому что здесь могут быть не только названия частей речи, а также и специфические названия (например, для словоформы 2009 это будет тип NUM_WORD – числовое слово).

Функция говорит нам о том, какую функцию в предложении выполняет данная словоформа – подлежащего, сказуемого, дополнения и т.д. Однако, здесь стоит отметить, что одна словоформа с учетом наших выводов о «схемах» предложений, может выполнять как функцию подлежащего, так и функцию дополнения (например, имя любого актера попадает под это правило). В таком случае в словаре делается приоритет для подлежащего, т.е. будет отведена одна запись с функцией подлежащего – это делается из соображений о том, что все-таки в повседневной речи мы чаще пользуемся подлежащими. Однако, при

последующей работе над данным исследованием это правило должно быть доработано с целью обеспечить гибкость конструкций.

Под ***способностью*** словоформы понимают то, что она является в конечной семантической сети или реляционной базе данных – сущность, свойство, отношение. Такая характеристика может потребоваться при анализе различных «схем» предложений.

Под ***«М.»*** в таблице понимается признак множественности числа: 0 – единственное число, 1 – множественное число, -1 – нет характеристики числа.

Заменяемый текст – текст, который будет участвовать в формировании запроса на языке логического программирования. При этом необходимо учитывать различные синтаксические нормы таких языков – постановку скобок, запятых и прочего. Также следует отметить, что в зависимости от того, с чем именно мы работаем – конкретно с базой знаний, имитирующей реляционную базу данных или же с самой базой данных, заменяемый текст будет отличаться. Более того, так как не существует русского интерфейса запросов к реляционным базам данных, необходимо переводить запросы на русском языке и все, что с ними связано (например, сущности, отношения) на английский язык. Для этого требуется более простой словарь с однозначным соответствием слов на русском и английском языках.

В конце разговора о словаре стоит отметить, что он является расширяемым, т.е. данные, которые приходят с морфологического разбора предложения, преобразуются в структуру словаря и добавляются в сам словарь. По умолчанию было принято решение в разрабатываемой системе на данном этапе работы обозначать все отсутствующее в словаре данные, приходящие с морфологического анализа, как существительные и константы. Такое решение связано с наибольшей частотой употребления таких словоформ и избавляет нас от более сложного морфологического и синтаксического анализа, хотя стоит отметить, что это может привести к некоторым ошибкам. Например, дата, отсутствующая в словаре, будет некорректно помечена. Но, вследствие

описанных в работе ограничений, налагаемых используемыми техническими средствами, эта проблема решается нами.

Второй метод основывается на знании того, как расположены слова в предложении и делается соответствующее преобразование. Так, «**Вася снимался в Матрице**» с выделенным отношением «снимался» подразумевает при себе константу слева и справа в предложении. Исходя из этого, строятся схемы проверки естественно-языковых запросов. Например, запрос «**покажи всех актеров.**» может быть представлен схемой «**вспомогательное слово – вспомогательное слово – сущность – признак конца**». Для запросов на выборку по атрибутам дополнительно используется схема «**атрибут - константа**», которая может встречаться несколько раз в предложении. Запятые и предлоги в схемах игнорируются.

В нашей работе была предпринята попытка воспользоваться первым методом обработки естественно-языковых запросов, однако была постигнута неудача вследствие некачественной синтаксической обработки исходного текста. Причины, способствующие этому факту, заключаются в не очень качественных используемых технических средствах (конкретно, в лингвистическом модуле). Поэтому было решено воспользоваться вторым способом и определить жестко схемы предложений, которые представляют собой своеобразную «маску», на которую для сравнения накладывается исходный запрос и при нахождении соответствия между запросом и схемой, запускается определенный порядок действий по формированию запроса на языке логического программирования.

Если мы работаем с семантической сетью, то далее полученный запрос просто запускается на выполнение и выдается результат в соответствии с правилами работы программ на языке логического программирования. Если же мы работаем с реляционной базой данных, то полученный запрос, при условии корректного составления, транслируется в запрос на языке работы с реляционными базами данных (например, Transact-SQL). Далее этот запрос

может быть запущен на выполнение с целью получения выборки из базы данных.

3. Технологический раздел

В этом разделе представлены основные моменты реализации разрабатываемого программного обеспечения, выбор технических средств, производится обоснование выбора таких технических средств. Осуществляется выбор формата входных данных, а также языка программирования, с помощью которого предполагается кодирование программного обеспечения. Представлена структура и интерфейс разработанной программы, а также описаны принципы работы с реализованным программным обеспечением.

3.1. Выбор исходных данных для построения семантической сети

Построение семантической сети, как было описано в конструкторском разделе, условно, разделяется на три этапа: выделение сущностей, связей между сущностями, а также атрибутов сущностей. При этом, в работе приведено упрощение о загрузке атрибутов из внешних источников. Таким образом, основной упор в семантической сети делается на выделение сущностей на заданную тематику из различных источников.

Согласно [28], наилучший результат в выделении специализированных однословных терминов (порядка 89% от выделенных экспертами терминов) достигается при использовании структурированных текстов, например, словарные статьи, которые приведены в [28]. Следовательно, необходимо выбрать такой корпус текстов, который бы удовлетворял требованиям по структурированности.

В данной работе было рассмотрено несколько различных источников информации, таких как:

1. Национальный Корпус Русского Языка;

2. Толковый Словарь Ожегова (онлайн-версия) [40];

3. Глоссарий кинематографических терминов [41];

4. Расширенный словарь кинематографических терминов [42];

При изучении содержания данных корпусов текстов и принципов работы было выявлено, что в национальном корпусе русского языка нет специально

подобранной коллекции словарных статей на необходимую тематику. Более того, доступ к корпусу ограничен для коммерческой разработки и разрешен при заключении лицензионного соглашения, что не очень удобно для работы конечного пользователя и разработки продукта для коммерческого использования. Однако, данный корпус содержит тексты со снятой морфологической и лексической омонимией, что очень удобно для выделения терминов. Также данный корпус содержит большую выборку текстов, что может понадобиться при выделении связей для сущностей, так как коллекция текстов может содержать незначительное число связей, как, например, в коллекции словарных статей.

Онлайн-версия толкового словаря Ожегова содержит необходимые структурированные словарные статьи, однако в этом словаре нет возможности собрать коллекцию текстов в соответствии с выбранной предметной областью, что затрудняет процесс обработки текстов и вносит очень большую зашумленность в исходные данные, что влияет на время и качество выделения.

Глоссарий кинематографических терминов, представленный в [41] представляет собой веб-ресурс, страницы которого содержат список словарных статей на тему кинематографии. Такая коллекция текстов представляет собой для нас наибольший интерес в соответствии с выбранным методом и требованиями по входным данным. Однако, этот ресурс содержит значительное количество недостатков, среди которых неоднозначное оформление словарных статей (много брака в заголовках статей, некоторые статьи не содержат четко выделенных заголовков), что не позволяет произвести качественное формирование образов документов и требует дополнительной обработки.

Словарь кинематографических терминов, представленный в [42] по структуре похож на глоссарий, описанный выше, но при этом он устраняет отмеченные недостатки. Более того, весь набор словарных статей размещен на одной странице, что позволяет упростить парсинг исходных данных засчет загрузки всего списка терминов за один запрос. Коллекция статей в данном словаре имеет средние размеры (порядка 350 статей), что позволяет получить

качественную выборку терминов за сопоставимое время. Таким образом, с учетом описанных преимуществ, в качестве исходных данных было решено выбрать данный глоссарий терминов.

3.2. Выбор средств программной реализации

В качестве целевой платформы, для которой будет вестись разработка, была выбрана ОС Windows, так как на данный момент ее использует достаточно большое количество пользователей и, так как в конечном итоге разрабатываемый программный продукт должен быть ориентирован на конечного пользователя, то разумно выбрать именно данную платформу. В качестве СУБД был выбран Microsoft SQL Server из соображений популярности и скорости работы. Используемым языком программирования является C#, так как он обладает рядом преимуществ, среди которых удобный объектный интерфейс для взаимодействия программы и базы данных. Более того, к основным достоинствам этого языка можно отнести следующие:

1. **Расширяемость системы**. Используя данный язык программирования, можно свободно загружать дополнительные исполняемые файлы, импортировать объекты и классы из других программ;
2. **Сложность разработки и сопровождения**. C# является достаточно читаемым и хорошо документированным;
3. **Открытая документация**, исходные тексты библиотек;
4. **Защищенность и контроль версий** подключаемых алгоритмов;
5. **Трудоемкость написания**;
6. **Унифицированная система типизации;**
7. Подлинная **объектная-ориентированность;**

Описанные выше преимущества позволяют достаточно быстро и качественно реализовать разработанную систему. Однако, у данного языка есть недостаток, связанный с обработкой больших данных – программы, написанные на этом языке, достаточно долго обрабатывают большие данные. Этот недостаток можно устранить при помощи использования небольшой

выборки исходных данных, а также засчет предварительной подготовки образов документов с помощью программ на других языках. В нашей работе было принято решение осуществить предварительное формирование образов документов и дальнейшее использование готовой коллекции на входе алгоритма формирования семантической сети.

3.2.1. Выбор парсера исходных данных

Так как выбранный глоссарий представляет собой веб-страницу, то для того, чтобы подготовить образы документов – словарных статей, необходимо выбрать соответствующий парсер веб-страниц. На сегодняшний день для платформы .NET широко используется **HTMLAgilityPack** – парсер веб-страниц, который обладает следующими преимуществами [45]:

1. **Поддержка DOM и XPath**. Позволяет находить путь к соответствующим элементам страницы, вводя путь к селектору с помощью строки;
2. **Предоставление интерфейса работы с HTML-документами** аналогично интерфейсу, определенному в System.XML;
3. Возможность **обработки плохо сформированных веб-страниц**;
4. **Возможность обработки страниц «из сети»,** т.е. напрямую загруженных из Интернета, а также загруженная как текстовый файл.

Поддержка XPath и возможность загрузки страницы прямо из Интернета позволяют упростить процесс подготовки образов документов до нескольких команд. При этом использование оптимизированной для работы потоков библиотеки позволяет ускорить по времени обработку дерева страницы. Поэтому решено использовать данную библиотеку в качестве парсера исходных данных.

3.2.2. Лингвистический модуль

Для того, чтобы производить обработку естественно-языковых запросов на русском языке, необходимо выбрать средство, предоставляющее данную возможность, учитывая при этом тот факт, что выбираемое средство должно

обеспечивать все виды необходимого анализа. На сегодняшний день существуют следующие технические средства обработки текстовой информации [43]:

1. графематический анализ:

Название	Метод	Языки	Лицензия	Платформа
AOT	словарный	русский, английский	LGPL	Linux, Windows
Lemmatizer	словарный	русский, английский	GPL	Linux
FreeLing	правила	русский, английский, итальянский, испанский, португальский, астурийский, валийский, галисийский, каталанский	GPL+коммерческая	Linux
Solarix	правила	русский, английский	коммерческая	Linux, Windows
Tokenizer	правила	русский, английский, немецкий	GPL	C
Greeb	регулярные выражения	русский, английский	MIT	Ruby
AskNet	правила	русский, английский	коммерческая	Windows

Таблица 3: Основные средства графематического анализа русского языка

2. синтаксический анализ:

Название	Метод	Языки	Лицензия	Платформа
AOT	грамматика HPSG	русский, английский, немецкий	LGPL	Linux, Windows
Solarix	правила	русский, английский	коммерческая	Linux, Windows
MaltParser	машинное обучение	русский, английский	собственная	Java
Link Grammar Parser	грамматика связей	русский, английский	BSD	Linux, Windows
AGFL	грамматика аффиксов над конечной решеткой	русский, английский, французский, испанский	GPL	Linux, Windows
ABBYY Compreno	правила	русский	коммерческая	Windows
AskNet	правила	русский, английский	коммерческая	Windows
DictaScope	правила	русский	коммерческая	FreeBSD, Windows
ЭТАП-3	правила	русский, английский	н/д	Windows
Синтактико-семантический анализ русского языка	функциональная модель языка	русский	коммерческая	Веб-сервис

Таблица 4: Основные средства синтаксического анализа русского языка

3. морфологический анализ

На сегодняшний день существует очень много различных средств морфологического анализа языка. Вот некоторые из них:

- AOT (словарный метод обработки).
- Snowball (алгоритм Портера).
- Stemka (словарный метод обработки).
- Pymorphy (словарный метод обработки).
- Myaso (алгоритм Витерби).
- Eureka Engine (машинное обучение).
- Pymystem3 (разрешение омонимии).
- SVMTool (метод опорных векторов).
- TreeTagger (деревья принятия решений).
- Solarix (словарный метод обработки).
- AskNet (словарный метод обработки, правила).
- и другие;

Разнообразие технических средств обработки текстовой информации позволяет нам разработать наиболее удобную и корректную систему обработки естественно-языковых запросов. Однако, так как мы уже выбрали целевую платформу, язык программирования и соответствующую СУБД, в своем выборе мы должны опираться на это ограничение. Таким образом, мы получаем следующий набор текстовых анализаторов, позволяющий нам проводить обработку запросов на начальном этапе:

Название	Метод	Языки	Лицензия	Платформа
AOT	словарный, грамматика HPSG	русский, английский, немецкий	LGPL	Linux, Windows
Solarix	правила	русский, английский	коммерческая	Linux, Windows
AskNet	словарный,	русский,	коммерческая	Windows

	правила	английский		

Таблица 5: Результирующий список технических средств обработки текстовой информации.

Однако, анализируя полученный список технических средств, мы узнаем, что у AskNet нет бесплатной версии для разработчиков, что не очень удобно в рамках разработки системы, а AOT не предоставляет API для C#, что в значительной степени усложняет работу над проектом. Таким образом, в качестве целевого средства обработки текстов был выбран Solarix. Он имеет следующие достоинства [44]:

- словарь, содержащий лексикон и тезаурус, содержит более 205 тысяч словарных статей, также вместо русской словарной базы можно выбрать английский словарь, содержащий 210 тысяч словарных статей;
- содержит все необходимые модули обработки для проекта – токенизатор, модуль морфологического и синтаксического разбора текста;
- скомпилированная библиотека для доступа к базе грамматического словаря для выбранной x32 или x64 платформы Windows или Linux, и .NET обертка для Windows, имеющая большой набор функций для выполнения склонения, спряжения, проверки согласования, определения морфологических атрибутов слова, и т.д.;

Вместе с имеющимися достоинствами, бесплатная версия Solarix обладает следующими ограничениями и недостатками:

- Solarix не воспринимает неслучайные последовательности.
- Solarix имеет проблему распознавания имен собственных (например, Леонардо ДиКаприо), а также слов, которые не содержатся в словаре (например, некоторые неологизмы).
- Solarix распознает причастия как прилагательные.
- Длительное время работы вследствие не очень эффективного алгоритма.
- Не всегда корректное построение синтаксического дерева отчасти из-за некорректного распознавания имен собственных. Например, запрос

«покажи всех актеров», Solarix преобразует как «показать(актер(весь))», что является корректным преобразованием с точки зрения дальнейшей работы, однако более сложный запрос с неизвестными словарю именами собственными будет обработан некорректно. Например, запрос «покажи фильмы с Леонардо ДиКаприо» будет выглядеть так: «показать(фильм(с(Леонардо)) ДиКаприо)», хотя в качестве результата ДиКаприо должен быть на одном уровне синтаксического дерева с Леонардо, но т.к. модуль ничего не знает о фамилии актера, запрос обрабатывается некорректно. Этот недостаток вносит существенные коррективы в обработку исходных запросов.

- Ограничение по числу словарных статей (порядка 20000).

Принципы работы составляющих данное техническое средство компонентов более подробно описаны в [44], в том числе, принципы работы синтаксического и морфологического анализаторов.

3.2.3. Выбор словаря для перевода сущностей

Для того, чтобы перевести запросы на естественном языке в запросы к выбранной реляционной базе данных, необходимо использовать специальный словарь для перевода терминов, связей и атрибутов на английский язык. Это связано с причинами, описанными в конструкторском разделе.

Существует большое число различных словарей, однако на данном этапе разработки программного обеспечения было решено ограничиться внутренним словарем, содержащим однозначные соответствия слов на русском и английском языках.

В дальнейшем предлагается воспользоваться внутренним словарем компании Google, который содержит значительно число различных переводов, и который находится в открытом доступе. К данному словарю можно обращаться через запрос, таким образом, можно присоединить данный словарь к системе, не загружая ее значительными «по весу» данными.

3.2.4. Выбор модуля математического обеспечения

Модуль математического обеспечения необходим для осуществления преобразования над терм-документной матрицей, содержание которой описано в разделе 2. При разработке компонента, отвечающего за автоматическую генерацию семантической сети, было решено использовать библиотеку MATLAB в качестве основного математического обеспечения. Такое решение обусловлено следующими преимуществами данной библиотеки:

1. Наличие **огромного набора пользовательских функций** обеспечивает простоту и гибкость вычислений;
2. **Высокая скорость обработки** больших массивов данных;
3. Возможность **разработки собственных функций** и библиотек на основе стандартных;
4. **Поддержка интерфейса для C++ и C#.**

При этом использование библиотеки MATLAB имеет некоторые особенности, которые будут описаны далее.

3.2.5. Выбор средств отображения результатов генерации семантической сети

Несмотря на опциональность данного компонента применительно к нашей системе, было принято решение внедрить в разработку проекта средства отображения различных графов и сетей. В качестве такого средства решено использовать **Graphviz v2.38**, так как он предоставляет достаточный функционал для отображения полученных результатов, не занимая при этом значительного места в памяти. Более того, он позволяет сгенерировать компактный наглядный рисунок с возможностью последующего масштабирования. Использование данного средства обусловлено также необходимостью автоматизации процесса формирования семантической сети.

3.2.6. Выбор технических средств логического программирования

Использование средств логического программирования в этой работе позволяет произвести обработку синтаксического дерева, полученного на выходе лингвистического модуля, представленным запросом к семантической сети, путем приведения его к запросу реляционной базы данных. На данный момент существует несколько различных языков логического программирования (среди которых выделяют Mercury, Planner, Q-LISP), однако наиболее популярным и удобным среди всех языков является Prolog. Данный язык обладает рядом преимуществ [46]:

1. **Единообразие представления элементов**: фактов, правил, сложных процедур, управляющих структур;
2. **Простота программной реализации** и самих программ, являющаяся следствием единства механизма вычислений;
3. **Возможность доступа к дереву доказательства**, механизмам ввода-вывода и возврата, а также к составляющим правилам;
4. **Возможность доказательства правил** и управления выполнением правил.

Существует много различных версий языка Prolog: Turbo Prolog, Visual Prolog (содержит структуру доменов – некое подобие типизации в языке), SWI-Prolog и другие. Однако, было принято решение использовать SWI-Prolog, т.к. эта реализация является наиболее приближенной к исходному стандарту логического программирования. Более того, SWI-Prolog – свободная реализация, часто используемая для построения семантических сетей, а также предоставляет богатый набор возможностей: библиотеки для многопоточности, юнит-тестирования, GUI, интерфейсы к Java, ODBC, C# (что является наиболее важным с точки зрения автоматизации обработки запросов), поддерживает платформы Unix, Windows, Macintosh.

Также, в SWI-Prolog имеется возможность добавления транслятора запросов к базе знаний Prolog в запросы к базе данных SQL [47]. Данный транслятор требует при себе определенной организации фактов в базе знаний, а именно а с помощью главных функторов relation и attribute:

relation(actors,5,'Actors').

attribute(1,'Actors','actor_id',int).

attribute(2,'Actors','name',string).

attribute(3,'Actors','surname',string).

attribute(4,'Actors','birth_year',int).

attribute(5,'Actors','gender',string).

Такое представление знаний в базе знаний имитирует представление таблиц в реляционной базе данных. Однако, с точки зрения нашей работы, такое представление не очень удобно для описания семантической сети, поэтому было принято решение разделить программу на две части и показать подробно работу с семантической сетью и базой знаний, и, отдельно от нее показать работу с реляционной базой данных.

Транслятор, представленный в данной работе, был впервые разработан Кристофом Дракслером в 1993 году [39], затем был адаптирован под современные версии SWI-Prolog Крисом Мунголлом [48] с добавлением следующих модификаций:

- Специальные типы для MySQL.
- SELECT DISTINCT.
- Перезапись запроса для оптимизации.
- Дополнительные операторы сравнения.

Оригинальный транслятор Дракслера имеет функциональные возможности по переводу следующих термов:

- Простая задача базы данных.
- Арифметическое выражение.
- Оператор сравнения.
- Негативная конъюнкция задачи.
- Терм агрегатной функции.

Транслятор, описанный в [39], страдает от некоторых недостатков. Первый присущ любому компилятору, который транслирует более многозначительный язык в менее многозначительный.

- комплексные структуры данных не могут использоваться в запросах доступа к базе данных, т.к. SQL не разрешает данные с комплексной структурой (кроме строк).

Прочие недостатки и ограничения данной конкретной реализации:

- **Транслятор не реализует весь язык SQL**. Целый набор арифметических функций и предикатов отсутствует. Более того, компилятор не производит условия HAVING и ORDER BY. Тем не менее, они могут быть просто добавлены:

  - сравнения, включающие свободные переменные и сравнения для условия HAVING.

  - проекционный терм для генерации условия ORDER BY.
- В настоящей форме **компилятор не поддерживает пользователя никаким образом**. Нет сообщений об ошибках, нет доступной помощи, нет индикации местонахождения ошибки.
- Транслятор в данной реализации **не может обрабатывать вложенные запросы.**

Ограничения накладываются на входные аргументы транслятора. Тем не менее, эти ограничения не уменьшают значительную активность языка доступа к базе данных.

- Все переменные проектных термов должны быть связаны во время всей задачи базы данных.

  Проекционный терм определяет условие SELECT, которое может содержать только квалифицированные атрибуты и константные значения. Только подтвержденные переменные могут быть переведены в квалифицированные атрибуты.

- Все явные количественные вычисления переменных (с помощью оператора **^/2**) должны предшествовать задаче базы данных.

  Экзистенциально количественно определенные переменные должны быть в словаре транслятора до перевода задачи базы данных, потому что переменные будут считаться универсально количественно определенными во время перевода, если они не находятся в словаре.
- Аргумент задачи терма агрегатной функции может быть только конъюнкцией задач базы данных.

  Агрегатные функции SQL не определены для UNION из запросов.
- Операторы отрицания должны прямо предшествовать операторам сравнения.

3.3. Описание основных моментов программной реализации

3.3.1. Использование библиотеки MATLAB в C#

Интеграция библиотеки MATLAB в проект, написанном на языке C#, имеет ряд способов. Первый основан на импорте компонента MATLAB Engine непосредственно в проект. Такой способ предполагает наличие предустановленного пакета MATLAB на компьютере. Его разумно использовать, когда осуществляется доступ к большому числу функций библиотеки, а также используются внутренние компоненты библиотеки, такие как средство отображения графиков.

Второй способ основан на использовании компонента MATLAB Runtime и внутреннего компилятора, позволяющего на основе написанного кода сгенерировать библиотеку, которая может быть подключена в проект. Runtime-библиотеки распространяются бесплатно и занимают меньшее пространство по сравнению с целой библиотекой. Однако вариативность набора функций и возможность использования средства отображения результатов приводят к выбору первого способа.

3.3.2. Формат хранения разреженных матриц

Так как в нашей работе происходит формирование терм-документной матрицы, важным аспектом в распределении памяти является формат хранения разреженных матриц. Предполагается, что составляемая матрица будет сильно разреженной, так как далеко не все слова встречаются во всех образах документов. Поэтому с точки зрения экономии памяти было принято решение использовать формат хранения матрицы, при котором известны индексы ненулевых элементов, по которым можно составить исходную матрицу. Составление исходной матрицы необходимо для передачи ее математическому модулю для осуществления сингулярно-векторного разложения.

3.3.3. Реализация функций лингвистического модуля

Функции лингвистического модуля представляют большой интерес в данной системе, так как они позволяют формировать данные, необходимые для дальнейшего преобразования в конечный запрос. Согласно разделам 1 и 2, для нас важны функции морфологического и синтаксического анализов. Алгоритм работы этих функций представлен на рисунке 11:

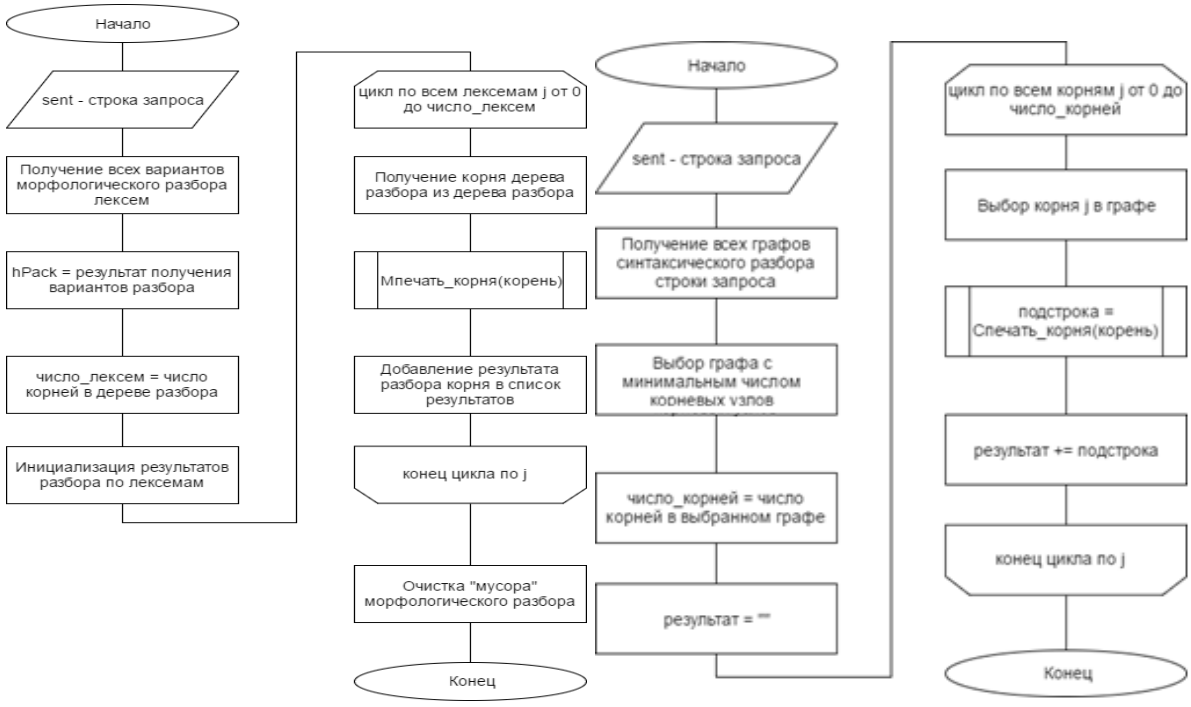


Рисунок 11. Схема алгоритма морфологического анализа естественно-

языкового запроса (слева) и синтаксического анализа (справа)

Отдельного внимания в данных схемах заслуживают функции ***Мпечать_корня*** и ***СПечать_корня***. Это функции, осуществляющие печать узлов деревьев морфологического (МПечать_корня) и синтаксического (СПечать_корня) разборов.

Алгоритм печати узла дерева морфологического разбора направлен на выявление всех проекций каждого слова. Так как мы считаем семантическую неоднозначность снятой, то берем первую проекцию каждого слова. При этом, если не находится ни одной проекции для заданного слова, его параметры инициализируются значениями по умолчанию. В качестве значений по умолчанию будем считать, что всякое неизвестное слово является существительным и константой, так как вероятность встречи такой неизвестной конструкции крайне высока. Следует отметить, что при построении семантической сети значением по умолчанию будет считаться тип «**НЕКОРР_СЛОВО**». Он позволит отсеять несуществующие слова.

Схема алгоритма печати узла морфологического разбора представлена на рисунке 12:

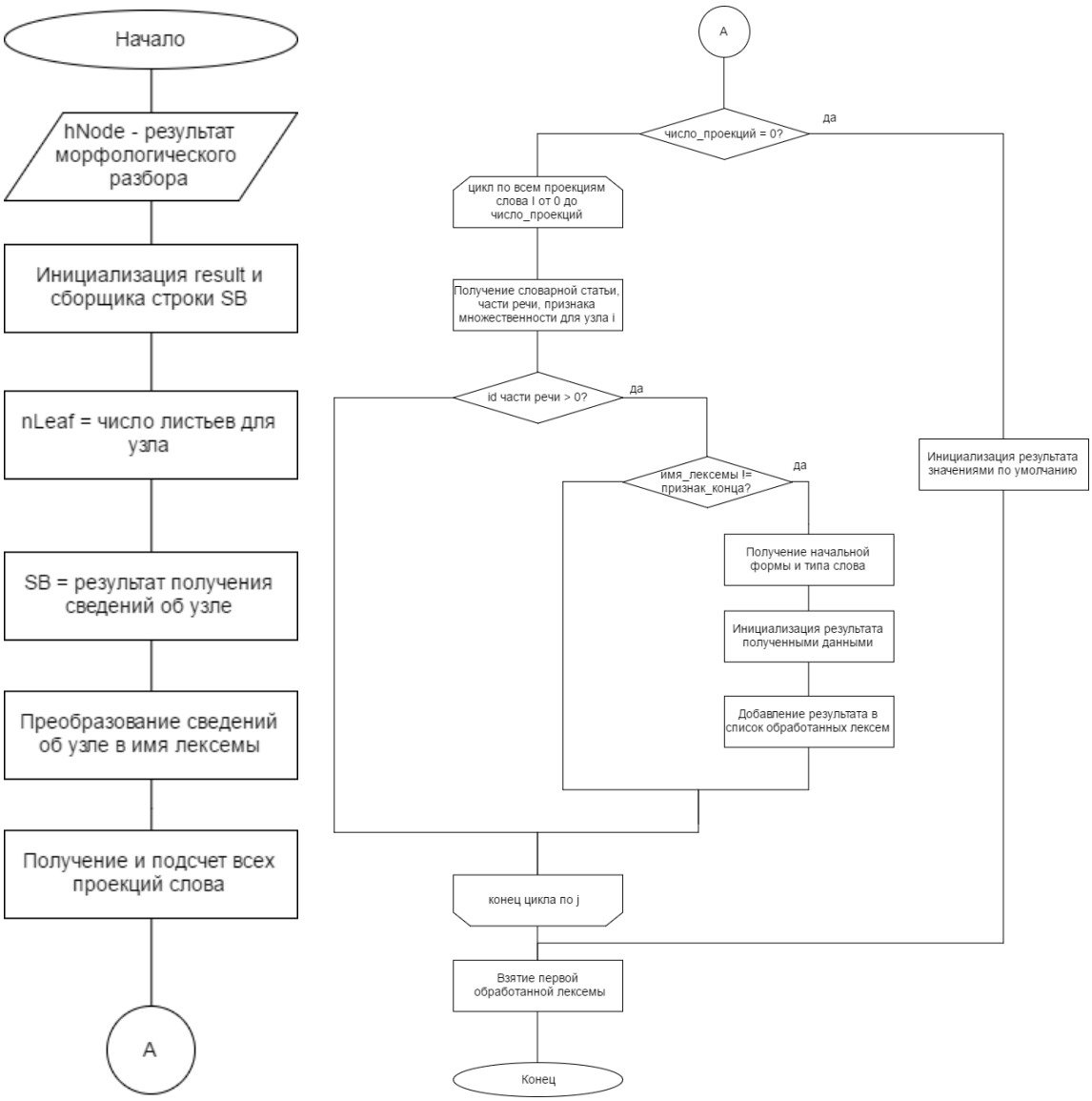


Рисунок 12. Схема алгоритма печати узла дерева морфологического разбора

Алгоритм печати узла синтаксического дерева направлен на формирование скобочной структуры, отражающей уровни дерева разбора. Полученная скобочная структура будет направлена на преобразование в Пролог-запрос. Схема алгоритма печати узла дерева синтаксического разбора представлен на рисунках 13,14:

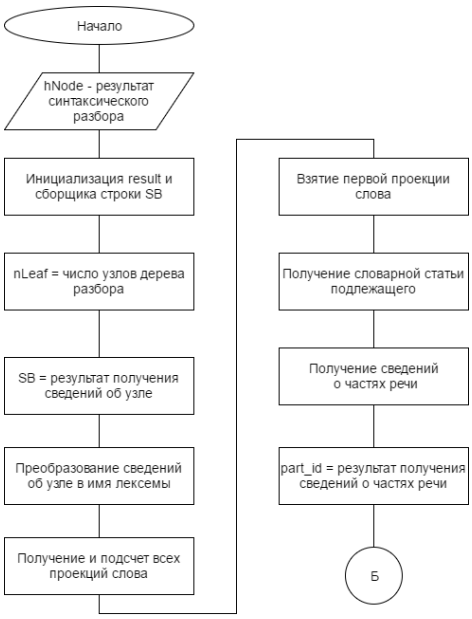


Рисунок 13. Схема алгоритма печати узла дерева синтаксического разбора (часть 1)

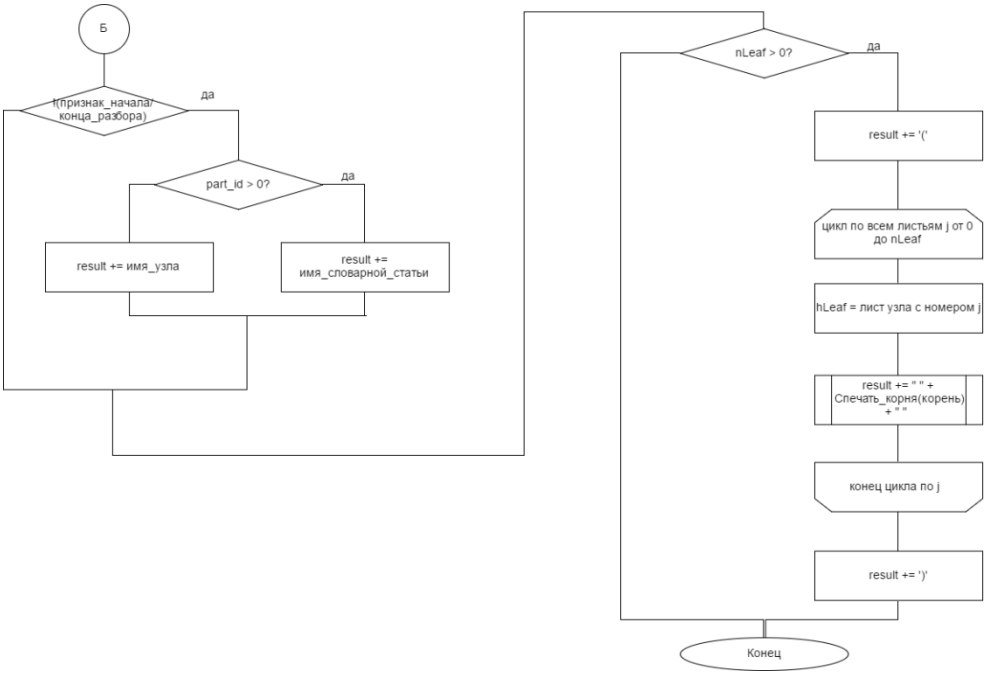


Рисунок 14. Схема алгоритма печати узла дерева синтаксического разбора (часть 2)

3.3.4. Работа с базой данных

Для корректного функционирования системы, в ней необходимо учесть способ наполнения данными, который может быть осуществлен пользователем. Поэтому в данной работе предусмотрен раздел работы с базой данных, где пользователь может самостоятельно через формы добавлять и удалять записи в

таблице. При этом осуществляются соответствующие проверки на наличие данных в таблицах.

3.4. Структура разработанного программного обеспечения

С учетом описанных выбранных программных средств, описанных в разделе 3.2. а также методов реализации программного обеспечения, описанных в разделах 1 и 2, разрабатываемая система принимает следующий вид:

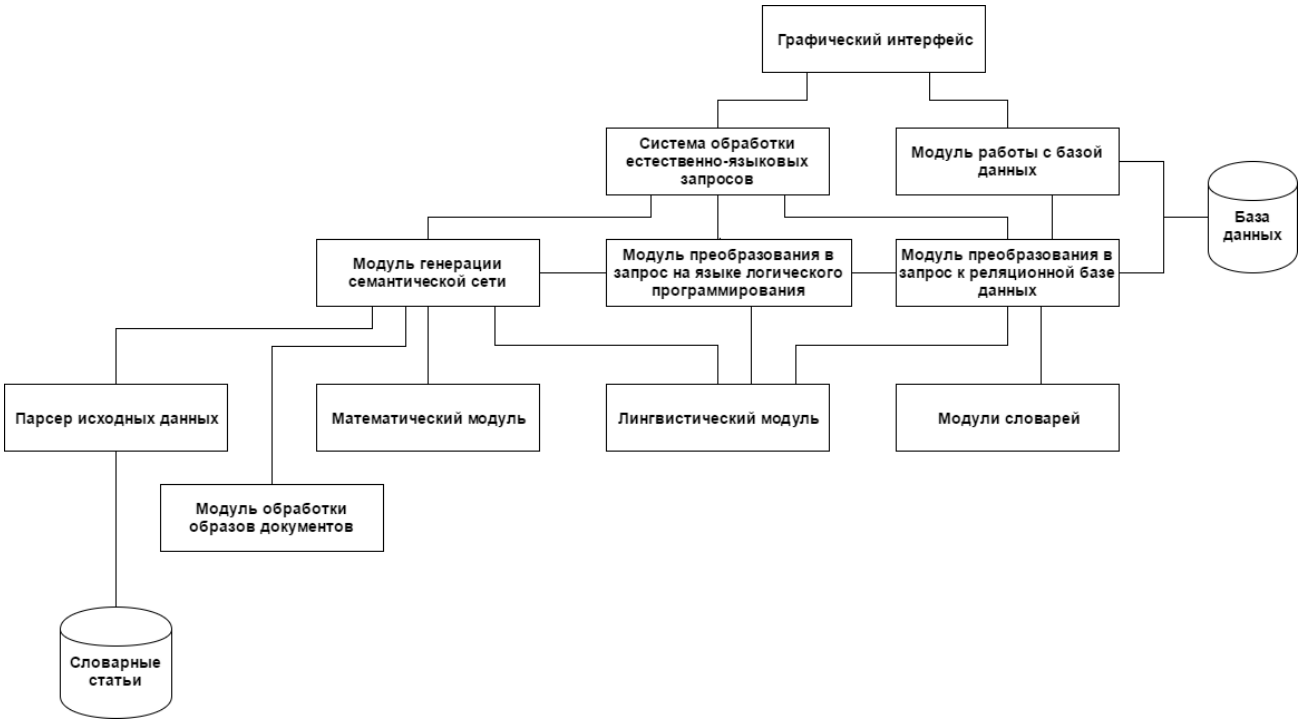


Рисунок 15. Структура разработанного программного обеспечения

При этом запрос, поступающий через графический интерфейс, будет поступать на вход системы обработки естественно-языковых запросов, где будет последовательно проходить все этапы обработки: преобразование в запрос к базе знаний и последующая его трансляция с помощью модуля преобразования в запрос к реляционной базе данных в необходимый запрос.

Диаграмма классов разработанного программного обеспечения представлена на рисунке 16:

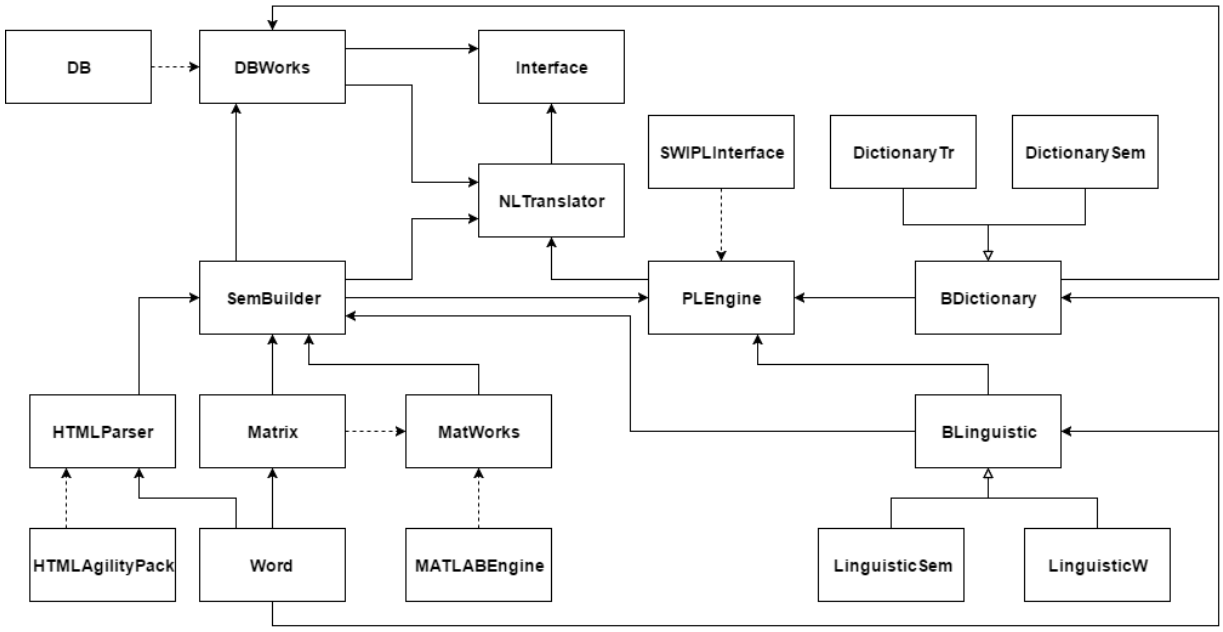


Рисунок 16. Диаграмма классов разработанного программного обеспечения

Абстрактный класс **BLinguistic** используется для представления работы лингвистического модуля. Данный класс используется на всех основных этапах формирования запроса к реляционной базе данных – генерации семантической сети и формировании запроса к базе знаний. Выделение базового абстрактного класса обусловлено схожим поведением объектов лингвистического модуля для операций построения сети и запроса к базе знаний. Однако, реализации классов, наследующих данный базовый класс будут отличаться принципом инициализации значениями по умолчанию (для класса **LinguisticSem** значением по умолчанию будет слова с типом «НЕКОРР_СЛОВО», в то время как для класса **LinguisticW** это слово будет иметь тип «СУЩЕСТВИТЕЛЬНОЕ»), что было описано в разделе 2.

Абстрактный класс **BDictionary** используется для представления работы модуля словарей. Так как в нашей работе будет поддерживаться два вида словарей – словарь для перевода (**DictionaryTr**) и хранения данных о словах (**DictionaryW**), но вместе с тем они будут обладать схожим поведением, то выделение базового класса в данной ситуации обосновано

Класс **SemBuilder** отражает работу модуля генерации семантической сети. Содержит в себе основные компоненты – парсер веб-страниц

HTMLParser, разреженную матрицу, представленную классом **Matrix**, а также математический модуль **MatWorks.**

Класс **Word** представляет собой основную функциональную единицу системы – слово. Содержит в себе необходимую информацию о встречаемости в документах, а также исходную форму в тексте и преобразованную словарем форму.

Класс **PlEngine** используется для представления работы модуля преобразования запроса к базе знаний в запрос к реляционной базе данных. Содержит метод загрузки скрипта на языке Пролог для инициализации используемого транслятора и структуры базы знаний и базы данных.

Класс **DBWorks** представляет работу модуля базы данных. Реализует функции добавления/удаления в реляционной базе данных, а также принимает на вход обработанную строку запроса в соответствии с предложенными методами.

Класс **NLTranslator** используется для сборки воедино всех разрозненных компонентов разработанной системы. Сосредотачивает в себе всю логику работы системы и позволяет отделить интерфейс от внутренних процессов в системе.

Классы **HTMLAgilityPack**, **MATLABEngine**, **SWIPLInterface** отражают работу заимствованных библиотек и объектов. Внутренняя реализация этих классов не изменяется в процессе работы над нашей системой.

3.5. Интерфейс программы

В начале работы с программой пользователю доступно исходное окно приложения, содержащее меню, в котором есть возможность выбора работы: добавление/удаление элементов в базу данных, а также преобразование естественно-языковых запросов в запросы к реляционной базе данных и работа с семантической сетью. Окно представлено на рисунке 17.

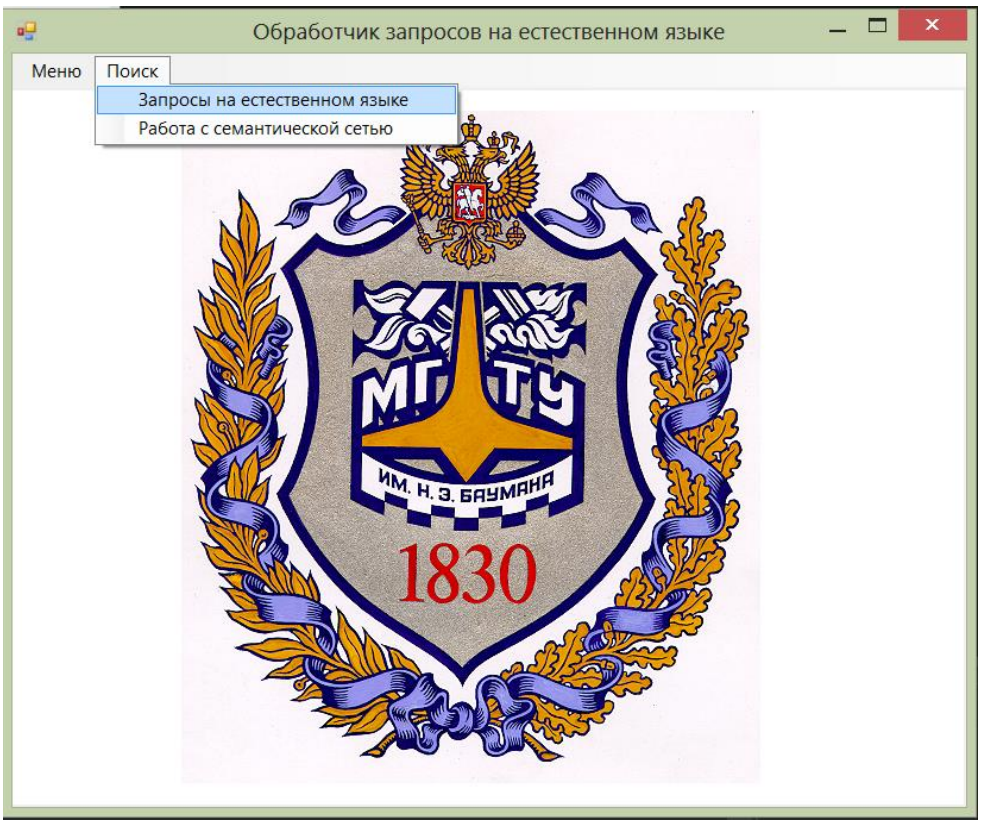


Рисунок 17. Главное окно приложения.

Далее, если была выбрана опция **Добавить элемент** во вкладке **Меню**, появляется окно добавления элементов в базу данных (рисунок 18).

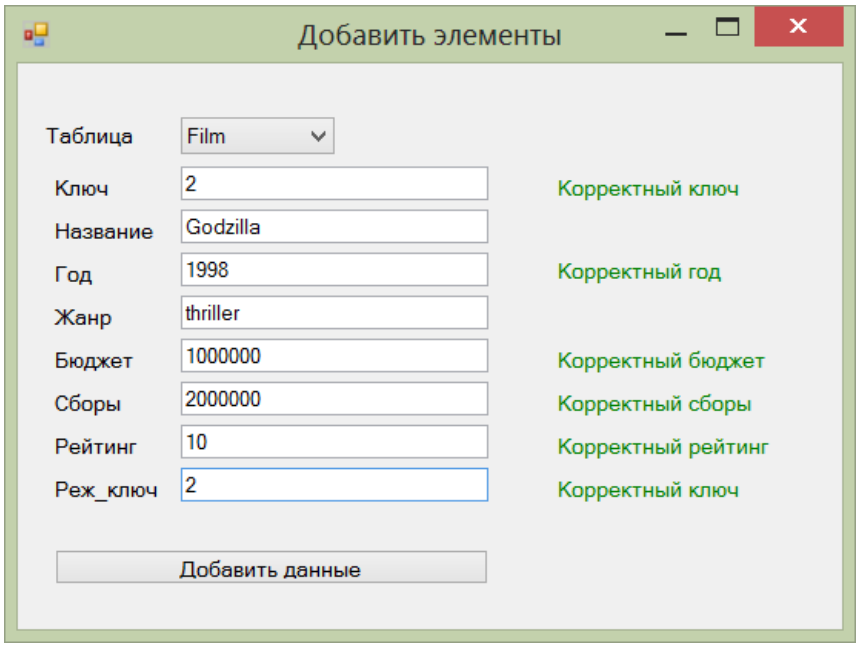


Рисунок 18. Окно добавления элементов.

Окно добавления элементов содержит выпадающий список, который позволяет менять конфигурацию добавляемого элемента, а именно, вводимые поля будут меняться в зависимости от выбранной сущности (рисунок 19).

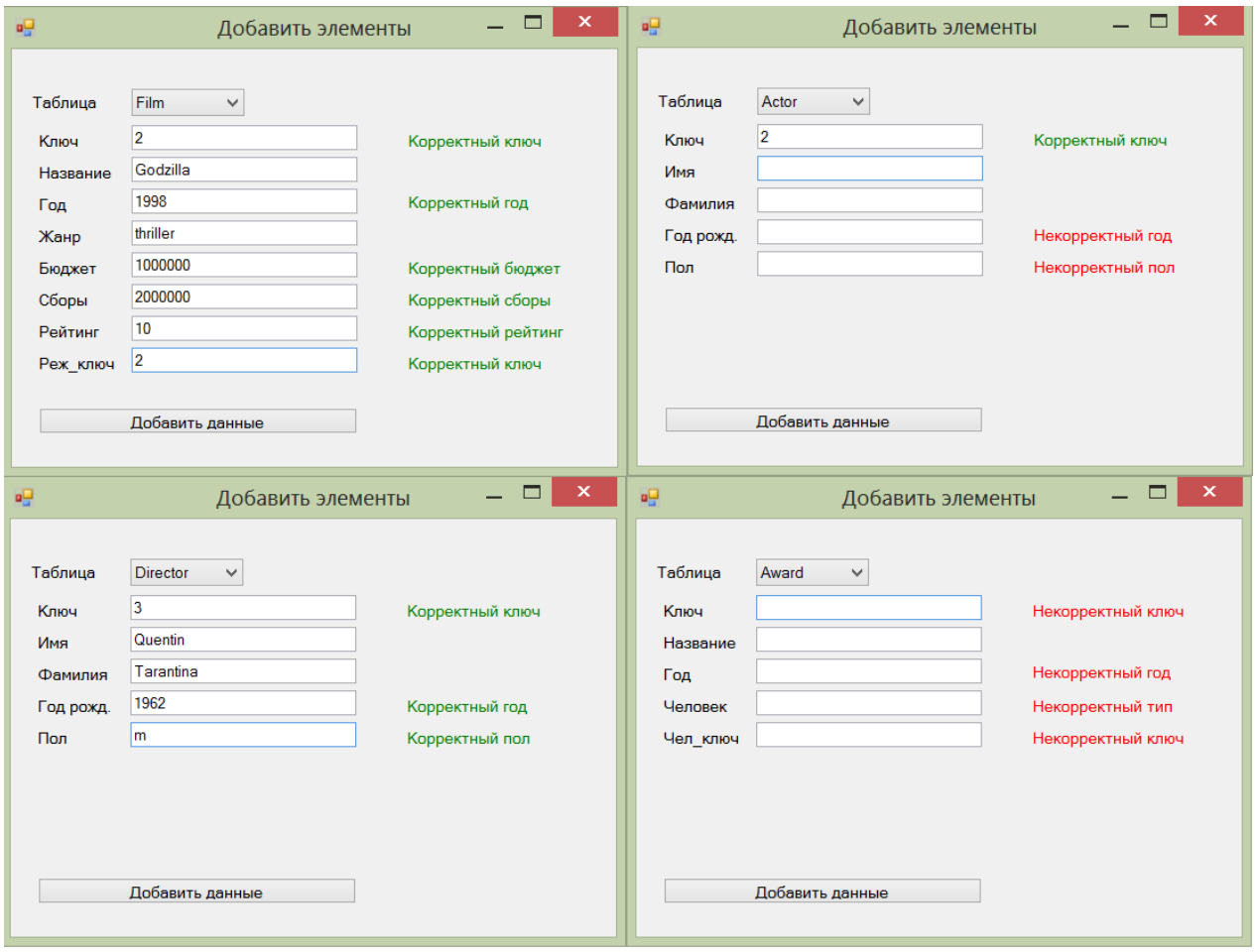


Рисунок 19. Варианты окна добавляемого элемента.

При работе в данном окне пользователь вводит необходимые данные в таблицы. При этом проверяется валидность всех вводимых полей в процессе ввода в соответствии с обозначенными выше ограничениями. Отдельно стоит упомянуть поле Person при выбранной сущности Award. Это поле воспринимает название одной из трех сущностей – Film, Director, Actor. В противном случае оно выдает ошибку. При условии соответствия всех полей можно произвести успешное добавление в таблицы базы данных. При этом выдастся диалоговое сообщение об успешном добавлении.

Опция **Удалить элемент** предоставляет пользователю программного продукта возможность удалять записи из таблиц по идентификатору. Удаление будет происходить независимо от того, есть ли запись с данным идентификатором в таблице или нет. Окно удаления элементов представлено на рисунке 20. После удаления выдается сообщение.

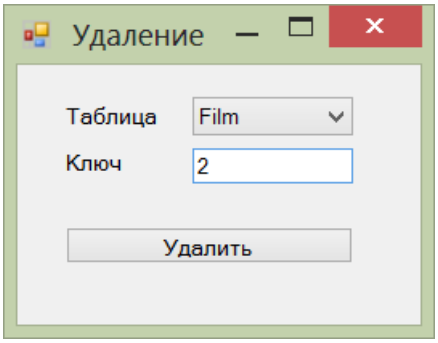


Рисунок 20. Окно удаления элемента.

Опция **Работа с семантической сетью** позволяет пользователю организовывать запросы на естественном языке к семантической сети, представленной базой знаний. Окно работы с семантической сетью представлено на рисунке 21.

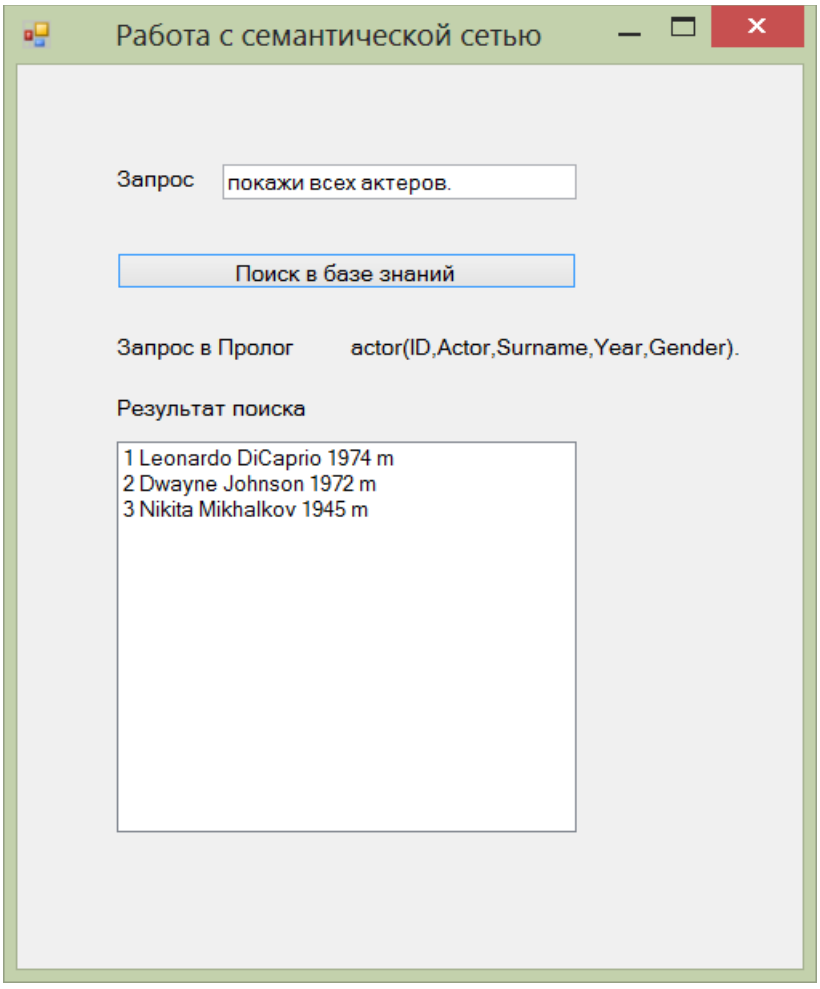


Рисунок 21. Окно работы с семантической сетью.

В данном окне пользователю предоставляется возможность ввести запрос на естественном языке в соответствии с наложенными на них в работе ограничениями. В поле вывода выводится результат работы полученного

запроса к базе знаний, а также сам запрос к базе знаний. В случае некорректного ввода запроса будет выдано сообщение об ошибке. Сообщение об ошибке формируется посредством модуля интерфейса языка логического программирования на язык C#.

Также в разработанном программном продукте реализована опция, позволяющая формировать запросы на естественном языке к реляционной базе данных. Соответствующее окно представлено на рисунке 22.

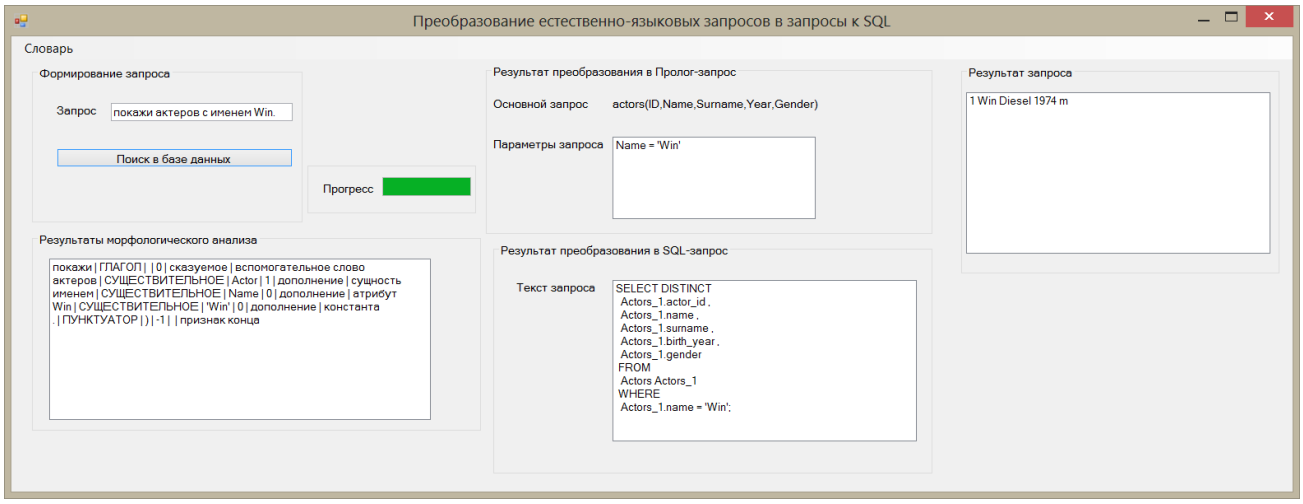


Рисунок 22. Окно работы с реляционной базой данных.

В данном окне предусмотрены вывод запроса к базе знаний, который впоследствии будет транслирован в запрос к реляционной базе данных, проинициализированные параметры запроса, текст преобразованного запроса на языке T-SQL, а также результаты морфологического анализа предложения. Также предусмотрен вывод на экран результатов работы запроса к реляционной базе данных и возможность посмотреть словарь во вкладке **Словарь** -> **Показать словарь**. Шкала прогресса показывает то, в какой стадии обработки запроса находится программа.

4. Экспериментальный раздел

4.1. Оценка качества выделения терминов

4.1.1. Оценка качества выделения терминов с помощью разработанного алгоритма

Так как в общем случае задачу выделения однословных терминов можно отнести к задаче классификации, то мы можем воспользоваться метриками оценки, используемыми в задачах подобного рода. Согласно [49] существует множество различных критериев качества семантической сети, которые в частном случае могут быть использованы для оценки качества выделения терминов – узлов графа, описывающего семантическую сеть. Таким образом, воспользуемся метриками, частично описанными в [49] и более подробно в [50].

Методика эксперимента, использующая данные метрики, осуществляется следующим образом: предположим, что мы имеем набор «образцовых» терминов, которые семантически близки заданному (например, «кинофильм»). Используя сведения из этого набора, а также сведения об общем числе существительных в выбранной коллекции документов, мы можем определить следующие характеристики списка, выделенных с помощью разработанного метода:

Название характеристики	**Описание**
tp (true positive)	Число слов, которые ожидали увидеть и которые оказались в результирующем списке терминов
fp (false positive)	Число слов, которые не ожидали увидеть в списке терминов, но которые появились в нем
tn (true negative)	Число слов, которые не ожидали увидеть в списке терминов и которые не попали в итоговый список
fn (false negative)	Число слов, которые ожидали в списке терминов, но они не оказались в нем

Таблица 6. Сведения о подсчитываемых характеристиках

Примечание: согласно [50] данные характеристики суммарно определяют общее число рассматриваемых слов в коллекции документов (в нашем случае, все существительные). Данный факт позволяет контролировать правильность расчета критериев в каждом эксперименте.

Имея эти характеристики, можно определить необходимые критерии качества: правильность (A), точность (P), полноту (R) и f-меру (F), которые определяются следующими формулами [50]:

$$A = \frac{tp + fp}{N}$$

$$P = \frac{tp}{tp + fp}$$

$$R = \frac{tp}{tp + fn}$$

$$F = 2 * \frac{P * R}{P + R}$$

$$N = tp + fp + tn + fn$$

Правильность определяет процент правильно выделенных терминов (соответствующих набору «идеальных» терминов) по отношению к общему числу слов в коллекции документов. Позволяет понять долю ошибок в работе алгоритма.

Точность показывает, какая доля «идеальных» терминов была выделена с помощью алгоритма.

Полнота характеризует способность алгоритма определять термины «наугад». Другими словами, полнота позволяет охарактеризовать алгоритм, который с определенной долей вероятности может случайно выделить нужный термин.

F-мера, согласно [50] является критерием, сочетающим в себе как точность, так и полноту и принимает наибольшее значение при абсолютном совпадении составляющих.

Использование подобных критериев качества обладает следующими преимуществами:

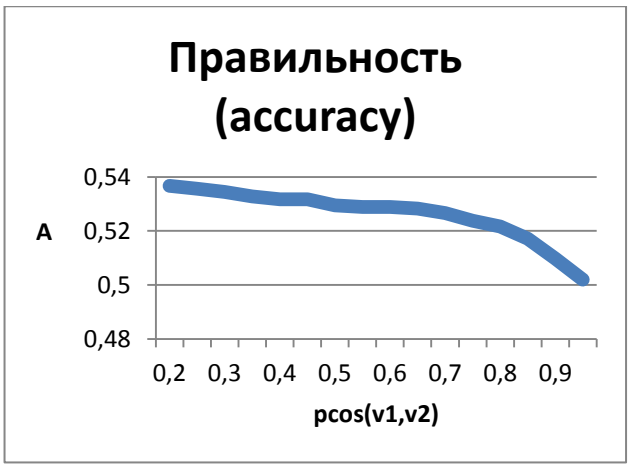
- позволяют произвести объективную оценку качества;
- позволяют сравнить предлагаемый метод с множеством других;
- данный метод прост в реализации и можно повторять многократно;

Данная методика предполагает при себе набор терминов, соответствующих «идеальной» семантической сети. Эту проблему применительно к нашей задаче было решено устранить, используя экспертную оценку: для этого, выбранному эксперту в области русского языка, а также в предметной области, связанной с нашей работой, был предложен общий список существительных, сформированный на основе образов документов, подготовленных в процессе работы метода. Далее эксперт вручную выделил набор терминов, близких по смыслу к заданному (для термина «кинофильм» был выделено 304 слова, соответствующих «идеальному» набору).

На основании вышесказанного, можем отметить, что эксперимент проводился следующим образом: с помощью разработанного метода выделения терминов генерировался список с заданной мерой косинусного расстояния, затем этот список сравнивался с «идеальным» и производился подсчет необходимых характеристик и критериев. Косинусное расстояние считалось для векторов, полученных с помощью сингулярно-векторного разложения, описанного в разделе 2.

В экспериментальной сессии использовалось два интервала: [0.2,0.95] с шагом 0.5, [0.99,0.999] с шагом 0.01, а также дискретный набор значений расстояния {0.99, 0.999, 0.9999, 0.99999, 0.999999, 0.9999999}.

Результаты для первого интервала представлены на следующих графиках:



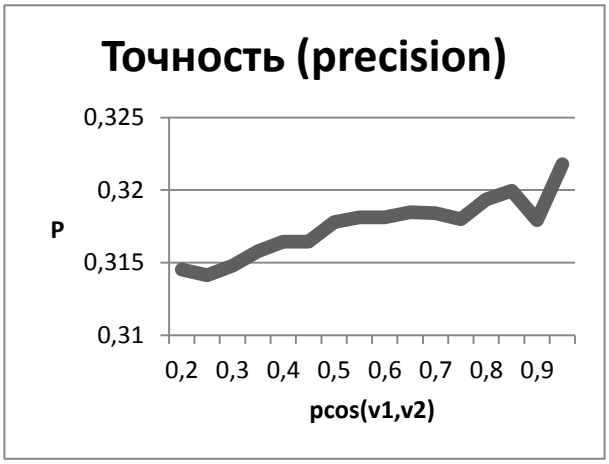
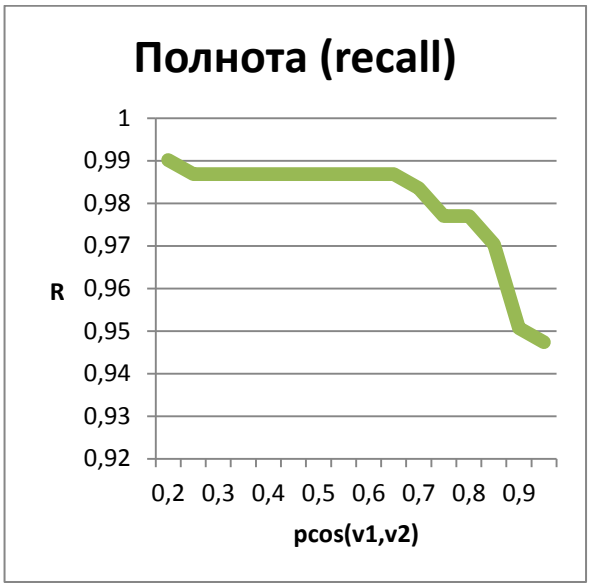


Рисунок 23. Оценка правильности и точности выделения терминов в интервале [0.2,0.95]



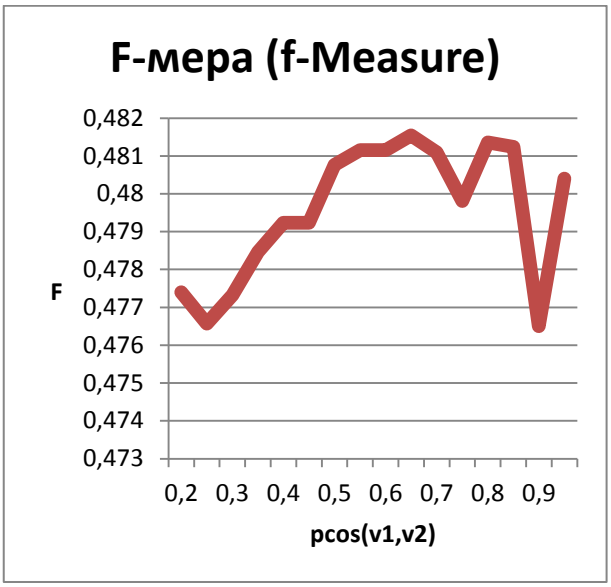


Рисунок 24. Оценка f-меры и полноты выделения терминов в интервале [0.2,0.95]

На основании полученных результатов, можно отметить, что с увеличением косинусного расстояния возрастает точность выделения терминов. Вместе с возрастанием точности происходит падение полноты выделения терминов, что говорит о возрастании числа терминов, ошибочно отнесенных к неправильным. На участке от 0.3 до 0.65 мы наблюдаем плато в графике полноты, что говорит нам о неизменности числа неправильных терминов при возрастающей точности.

Дополнительно следует отметить, что параметр точности задает общий тренд графика f-меры, который усиливается критерием полноты. На данном

интервале для заданной выборки наиболее оптимальным является косинусное расстояние в 0.65, так как здесь наблюдается лучшее значение f-меры и не происходит резкого убывания полноты и правильности при сравнительно оптимальной точности. Максимум графика f-меры говорит о наиболее оптимальном соотношении полноты и точности.

Для второго интервала были получены следующие результаты:

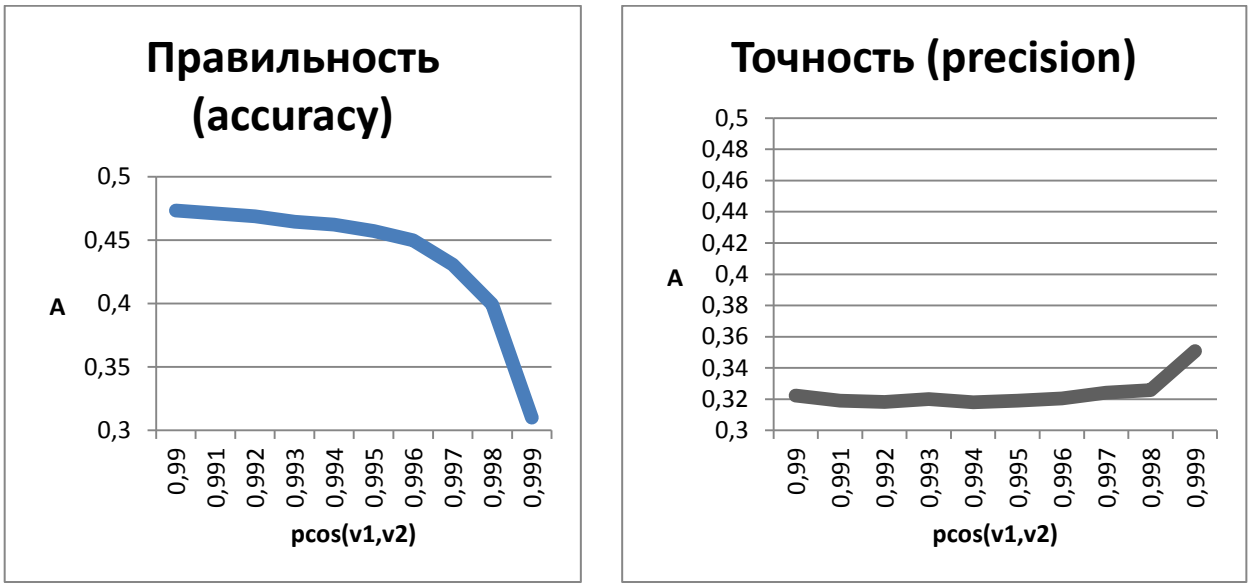


Рисунок 25. Оценка правильности и точности выделения терминов в интервале [0.99,0.999]

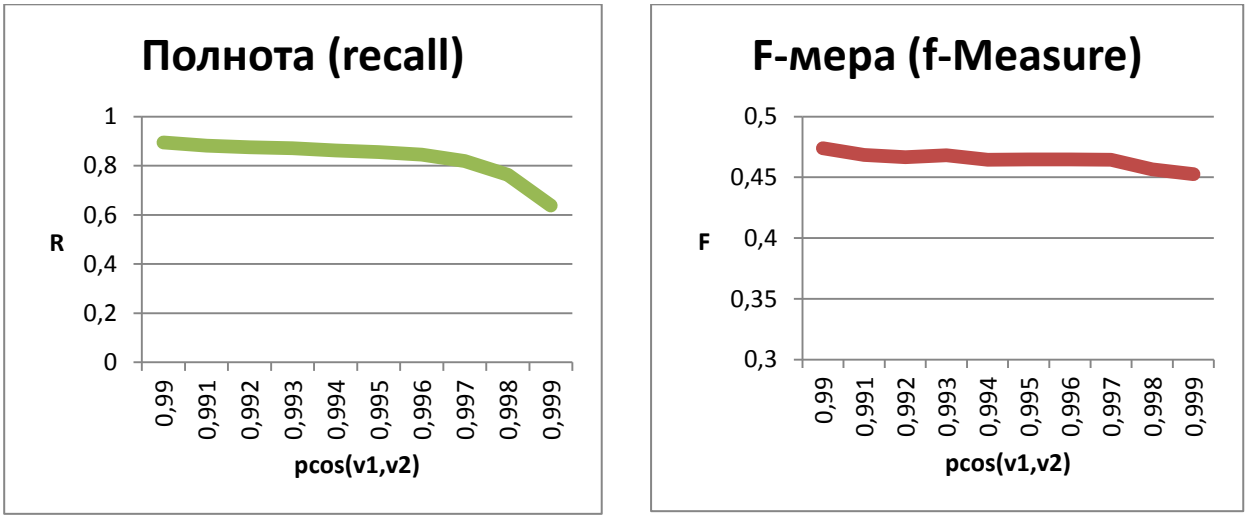


Рисунок 26. Оценка f-меры и полноты выделения терминов в интервале [0.99,0.999]

Из полученных результатов видно, что на интервале, близком к единице, происходит устойчивый рост точности, а при приближении к 1 происходит

резкое возрастание точности. При этом происходит резкое падение полноты на всем интервале, что приводит к падению f-меры. Такое падение говорит о неоптимальном соотношении точности выделения терминов и их полноты. Иными словами, мы получаем более точную выборку терминов, в которой в значительной степени не будет хватать различных необходимых слов.

Эксперимент на дискретном наборе значений позволяет охарактеризовать работу алгоритма в прямом соответствии с гипотезой о близости слов в предложениях. Его результаты представлены на следующих графиках:

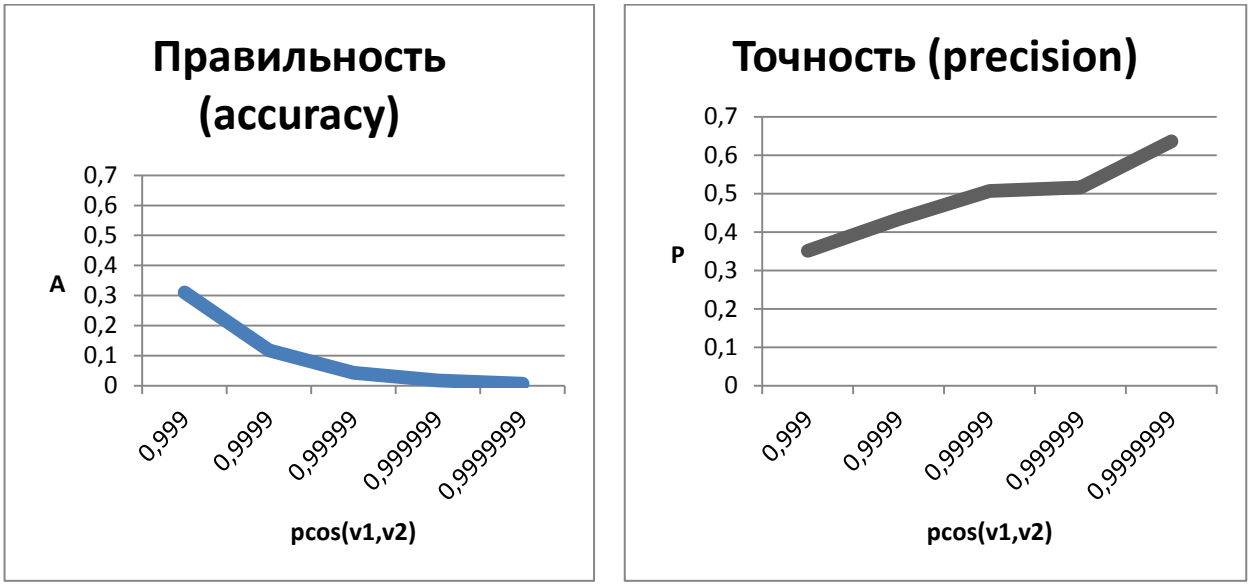


Рисунок 27. Оценка правильности и точности выделения терминов на дискретном наборе значений косинусного расстояния

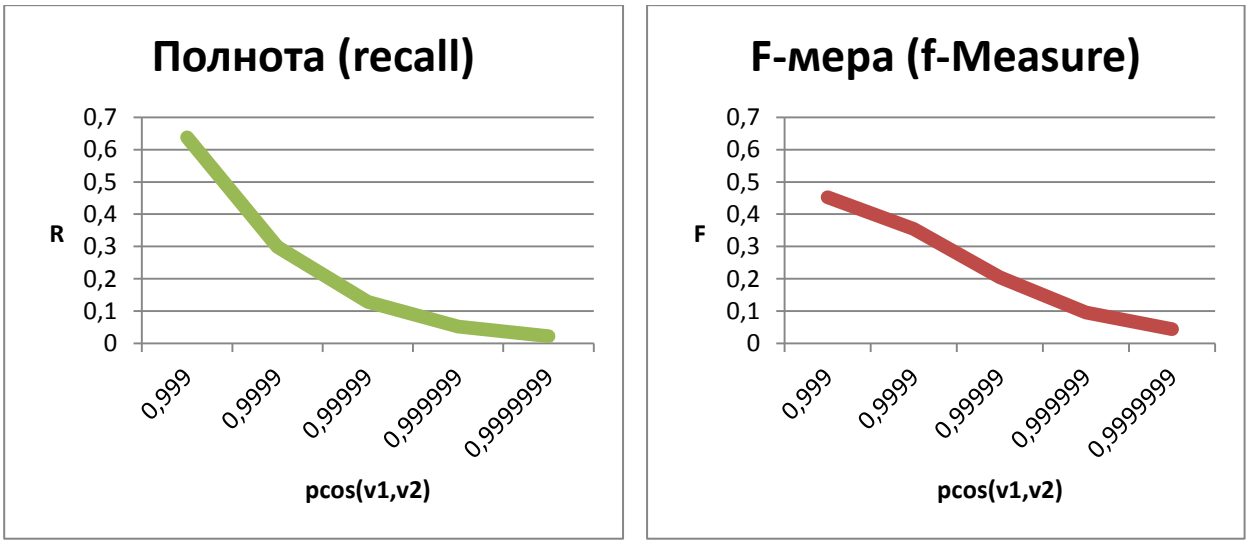


Рисунок 28. Оценка f-меры и полноты выделения терминов на дискретном наборе значений косинусного расстояния

Данные результаты говорят нам о том, что при стремлении угла между векторами к нулю, конечная выборка состоит практически из всех терминов, находящихся в «идеальном» списке, но их количество очень мало, что не позволяет говорить о качестве такого результирующего списка.

Объединив три набора значений воедино, мы получим общую картину зависимости качества выделения терминов от косинусного расстояния:

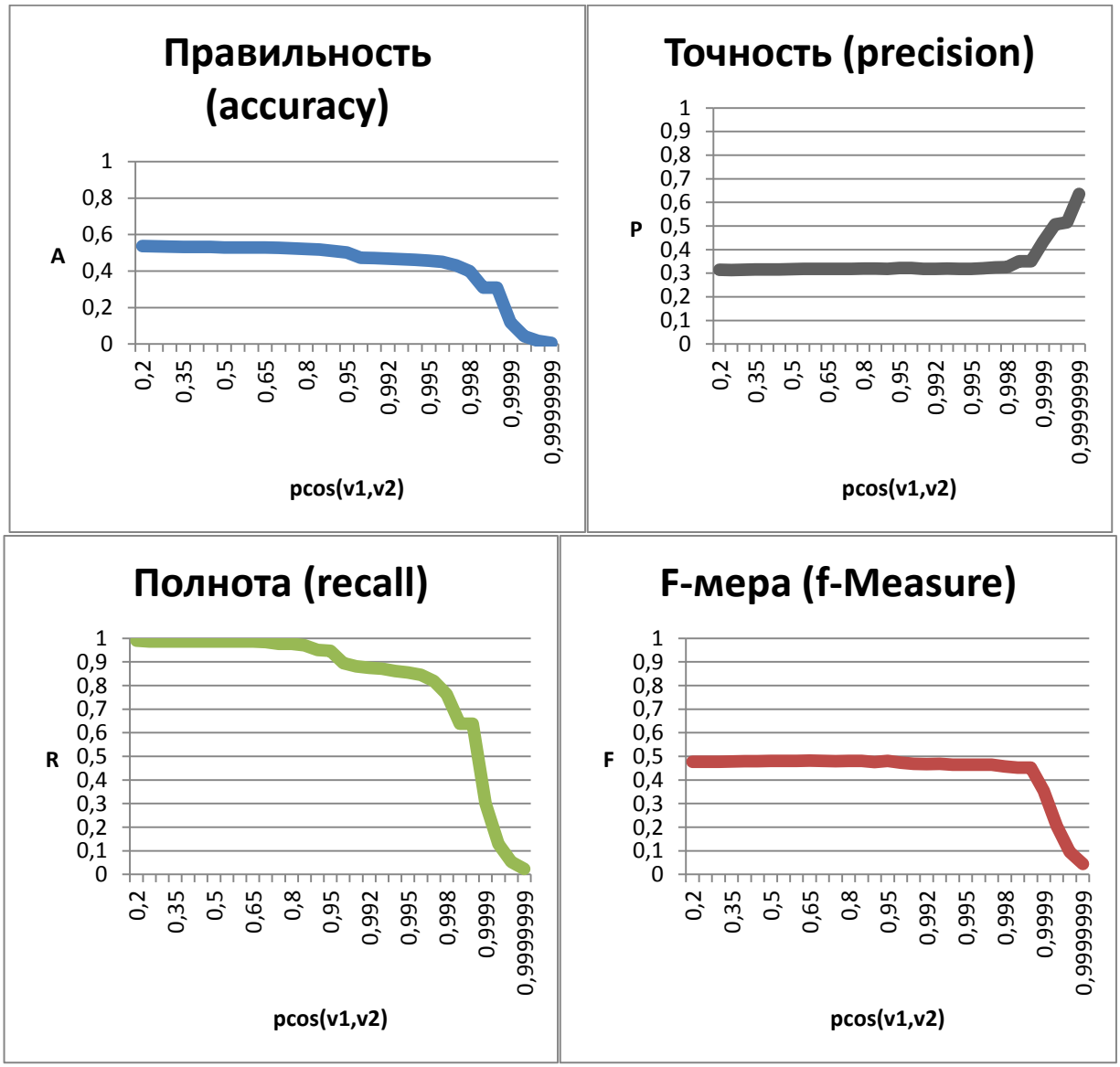


Рисунок 29. Зависимость полноты, правильности, точности, f-меры от косинусного расстояния

Заметим, что наиболее точная выборка при сравнительно высокой полноте достигается при значении косинусного расстояния в 0.999, что подтверждает гипотезу о семантической близости слов в зависимости от угла между векторами встречаемости двух слов в тексте.

Таким образом, мы получили следующие количественные результаты:

- наиболее оптимальная выборка достигается при значении косинусного расстояния 0.65;
- наиболее точная выборка с сохранением полноты достигается при значении косинусного расстояния, равном 0.9999;
- при значении косинусного расстояния, меньшем 0.65, наблюдается рост f-меры, после – падение;
- полнота вносит наибольший вклад в общий тренд на всем интервале значений косинусного расстояния.

4.1.2. Сравнение алгоритма выделения терминов и Word2Vec

Word2Vec – программное средство компании Google, которое представляет собой семантический анализатор и использующий векторное представление слов и дистрибутивную семантику. В основе работы данного средства лежит модель Continuous Bag of Words(CBOW), а также алгоритмы, связанные с обучением нейронной сети.

Для сравнения Word2Vec и разработанного алгоритма воспользуемся методикой, описанной в 4.1.1. В качестве средства, реализующего работу Word2Vec, воспользуемся средством, представленным в [51]. В качестве входной выборки используем координаты векторов, полученные с помощью сингулярно-векторного разложения для терм-документной матрицы, полученной нашим алгоритмом. Список для сравнения с «идеальным» формировался путем взятия первых tp+fp значений из результирующего списка Word2Vec, в котором слова отсортированы по убыванию близости. Результаты сравнения представлены на рисунках 30-33:

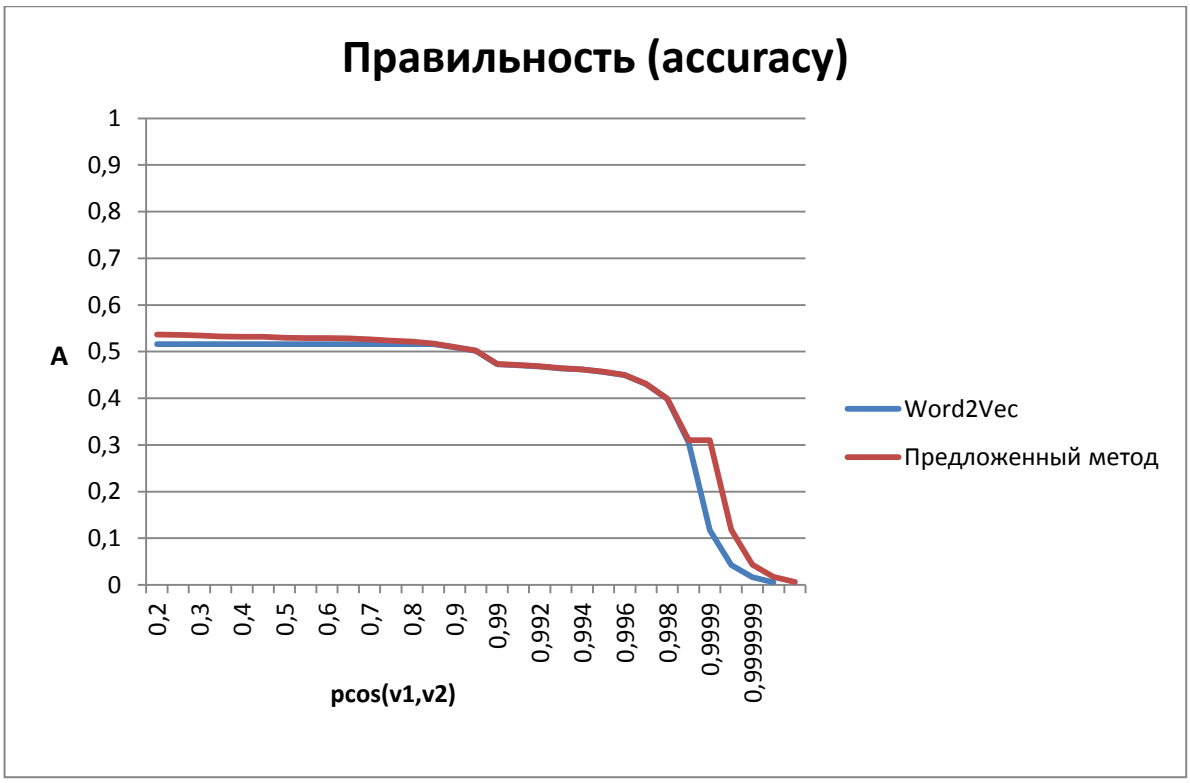


Рисунок 30. Сравнение правильности в зависимости от косинусного расстояния для Word2Vec и внутреннего алгоритма

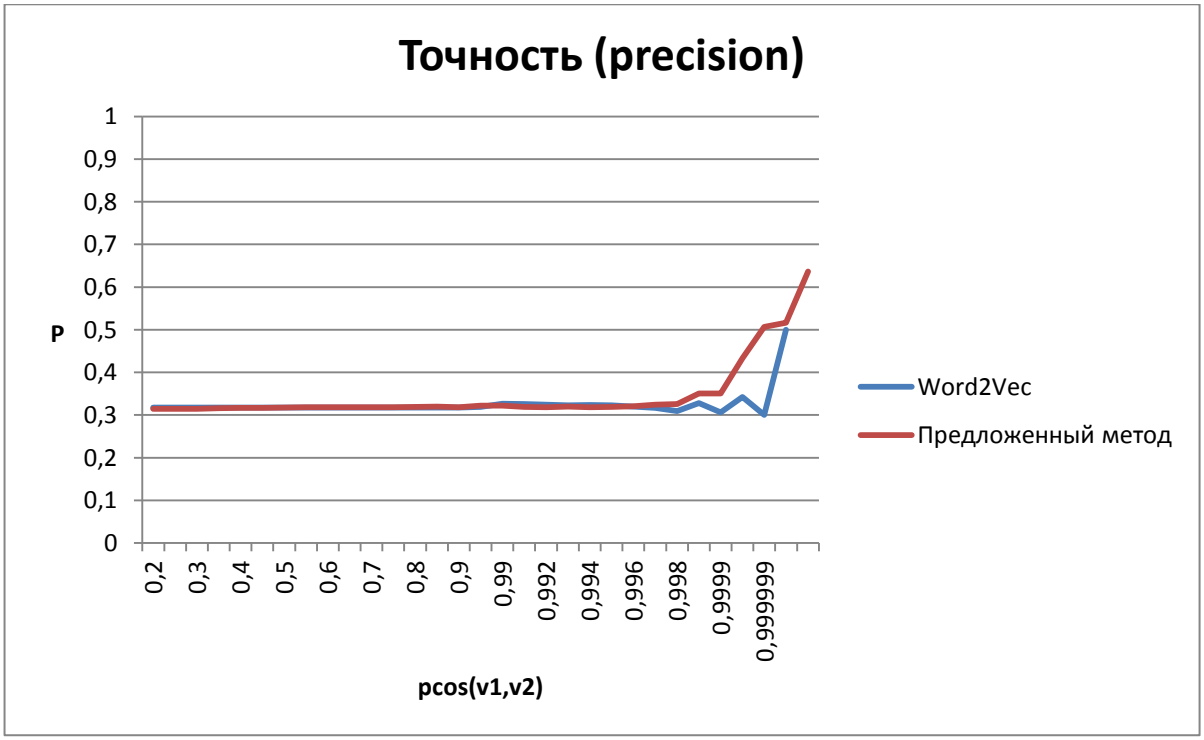


Рисунок 31. Сравнение точности в зависимости от косинусного расстояния для Word2Vec и внутреннего алгоритма

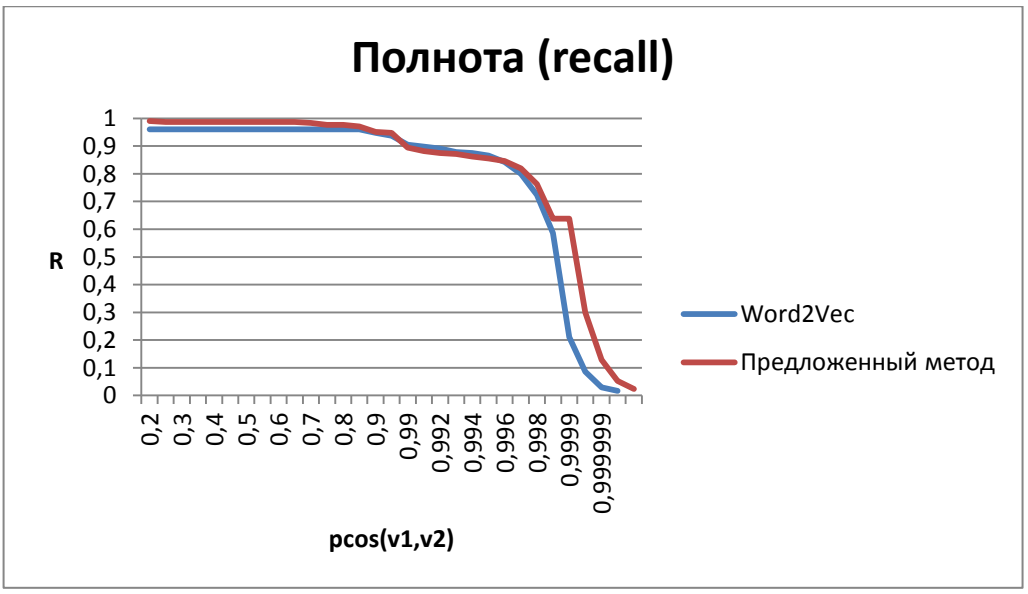


Рисунок 32. Сравнение полноты в зависимости от косинусного расстояния для Word2Vec и внутреннего алгоритма

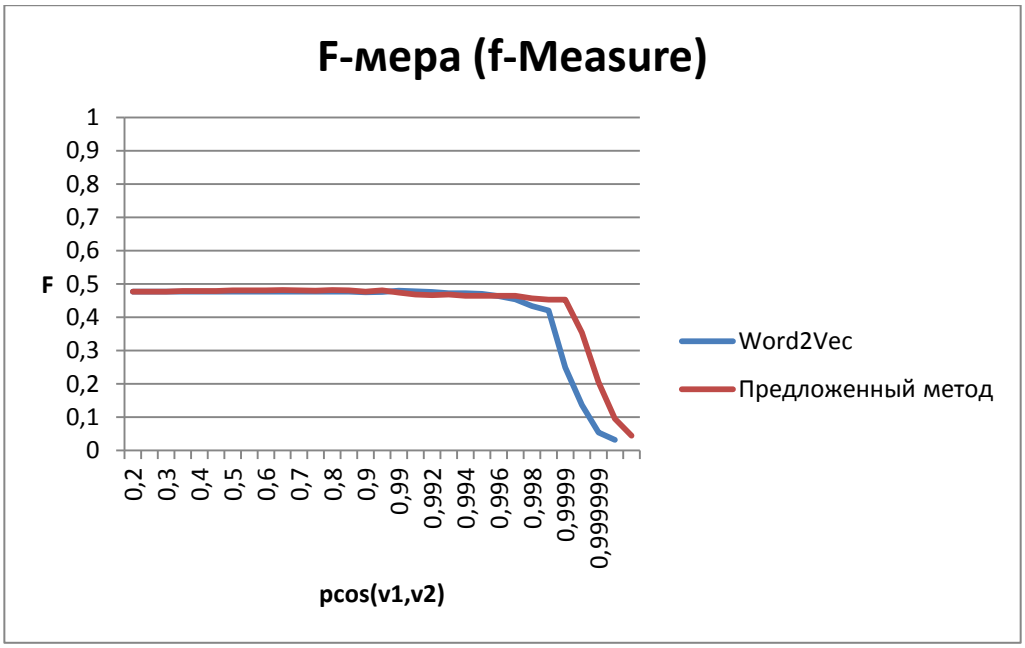


Рисунок 33. Сравнение f-меры в зависимости от косинусного расстояния для Word2Vec и внутреннего алгоритма

Как видно из полученных результатов, мера сравнения близости векторов, применяемая в нашем алгоритме, для данной выборки текстов в целом представляет лучшие результаты, за исключением интервала [0.99,0.995]. При этом общий тренд графиков сохраняется, что говорит о подобии методов, применяемых в нашей работе и Word2Vec.

4.2. Исследование преобразования естественно-языкового запроса

4.2.1. Исследование времени преобразования естественно-языкового запроса в запрос к реляционной базе данных

Для того, чтобы оценить время преобразования естественно-языковых запросов в запросы к реляционной базе данных было разработано множество естественно-языковых запросов со все увеличивающимся числом словоформ. Все запросы организовывались к таблице актеров. Таблица актеров наполнена 1000 записей схожего характера, но с разными идентификаторами.

Таблица сформированных запросов представлена далее. В ней определяется каждая словоформа, независимо от того, пунктуатор это или слово, считается как независимая.

Текст запроса	Длина запроса
покажи актеров	2
покажи актеров.	3
покажи всех актеров.	4
покажи всех актеров с именем name	6
покажи всех актеров с именем name.	7
покажи всех актеров с именем name, фамилией surname	9
покажи всех актеров с именем name, фамилией surname.	10
покажи всех актеров с именем name, с фамилией surname.	11
покажи всех актеров с именем name, с фамилией surname, полом мужским	13
покажи всех актеров с именем name, с фамилией surname, полом мужским.	14
покажи всех актеров с именем name, с фамилией surname, с полом мужским.	15

Таблица 7. Набор преобразуемых естественно-языковых запросов.

Время работы функции преобразования запроса замерялось при прогонке функции в течение 100 раз, а затем общее время работы делилось на количество

прогонок функции. Тестирование проводилось на компьютере с процессором AMD A8-550M APU with Radeon™ HD Graphics с частотой 2,10 ГГц. Временные характеристики приведены на следующем графике:

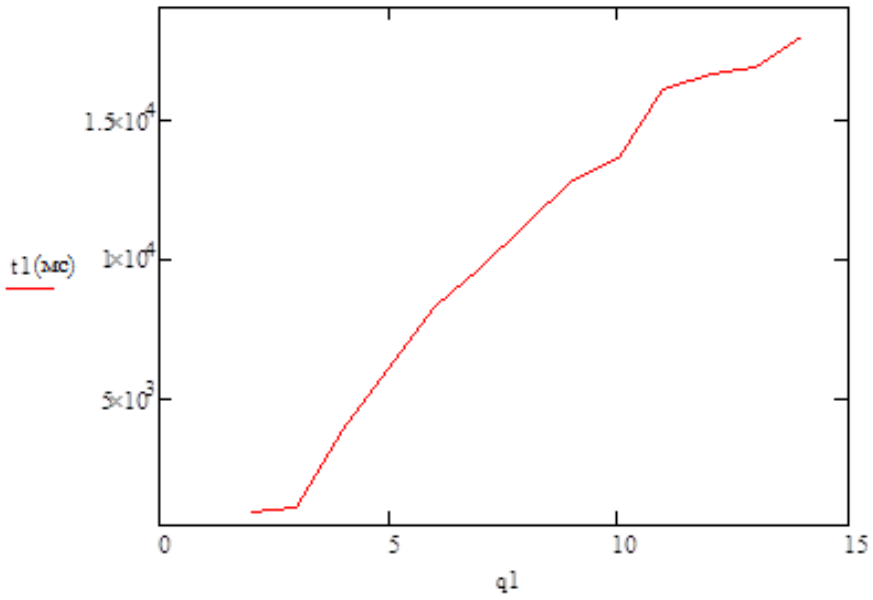


Рисунок 34. Время выполнения преобразования естественно-языкового запроса (мс) в зависимости от длины запроса.

По результатам проведенных исследований достаточно сложно выявить какую-либо зависимость в изменении времени преобразования запроса в зависимости от длины запроса. С некоторой долей вероятности данные результаты можно аппроксимировать, как логарифмическую, но для более точных выводов требуется дополнительное, более глубокое исследование.

4.2.2. Сравнение времени преобразования запроса с использованием семантической сети и основного модуля

Аналогично пункту 4.1.1. было проведено исследование времени обработки естественно-языкового запроса для семантической сети. Далее было решено сравнить времена обработки на схожих запросах. Ввиду ограничений, налагаемых на запросы, и особенностей работы двух модулей, список запросов ограничивается самым простым набором, представленным в таблице 6:

Время выполнения (основной модуль, мс)	Текст запроса	Время выполнения (семантическая сеть, мс)
5417	покажи всех актеров.	5174
4918	покажи всех режиссеров.	4673
4613	покажи все фильмы.	4543
5339	покажи все награды.	5141

Таблица 8. Сравнительная характеристика времени работы модуля работы с семантической сетью и основного модуля.

Как можно заметить из результатов измерений, время преобразования запроса с использованием семантической сети в целом быстрее, чем преобразование с помощью основного модуля. Это может быть связано с необходимостью дополнительной трансляции в основном модуле. Таким образом, для более быстрого преобразования запроса может быть удобным пользоваться исключительно семантической сетью.

Заключение

В результате проделанной выпускной квалификационной работы была разработана, спроектирована и реализована система преобразования естественно-языковых запросов к реляционным базам данных. В процессе ее создания были достигнуты следующие результаты:

1. Разработан и реализован метод автоматической генерации семантической сети с использованием структурированных текстов;
2. Выделены наиболее часто используемых схемы естественно-языковых запросов;
3. Спроектированы и реализованы методы лингвистической обработки запросов с использованием стороннего программного обеспечения;
4. Спроектирована структура программы, реализующая работу транслятора естественно-языковых запросов к базе данных;
5. Реализовано программное обеспечение, демонстрирующее работу системы, а также метода автоматической генерации семантической сети;
6. Проведено исследование работы полученного программного обеспечения, произведены оценки качества и скорости работы системы;

С учетом полученных результатов можно констатировать факт соответствия разработанного программного обеспечения техническому заданию.

Разработанное программное обеспечение обладает следующими достоинствами:

1. Данное программное обеспечение является полностью бесплатным засчет использования открытых компонент;
2. Возможность расширения системы для обработки большего числа вариантов запросов по сравнению с конкурентами;
3. Автоматическая настройка системы, использующая в качестве исходных данных коллекцию текстовых документов;

К основным недостаткам разработанного программного обеспечения можно отнести зависимость от ресурсоемких сторонних компонент (например, библиотеки MATLAB), использование несовершенных компонент, к которым можно отнести лингвистический анализатор, следствием чего является долгий процесс генерации семантической сети, а также отсутствие возможности обрабатывать вложенные запросы. Мы не относим к недостаткам обработку одного класса запросов, так как схемы запросов могут быть расширены.

С учетом полученных результатов и оценки текущих достоинств и недостатков можно выделить следующие направления развития:

1. Поиск/реализация более качественного лингвистического анализатора;
2. Модификация транслятора с языка логического программирования для возможности обработки сложных и вложенных запросов;
3. Расширение числа схем обработки языковых запросов;
4. Оптимизация процесса генерации семантической сети, что позволит ускорить настройку системы;
5. Модификация словарей для охвата большего числа запросов.

Список использованных источников

1. Евдокимова И.С. Естественно-языковые системы: курс лекций. – Улан-Удэ: Изд-во ВСГТУ, 2006. 92 с.
2. Компьютерная лингвистика. Энциклопедия «Кругосвет» [Электронный ресурс]. URL: http://files.school-collection.edu.ru/dlrstore/0032e870-dadd-5ab2-a83e-85e032c459b2/1009220A.htm (дата обращения 19.03.2019).
3. Большакова Е. И., Клышинский Э. С., Ландэ Д. В., Носков А. А., Пескова О. В., Ягунова Е. В. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика: учебное пособие. М.: МИЭМ, 2011. 272 с.
4. Седунов А.А.. Модель графематического анализа в системе обработки естественного языка. М.: ВГУ, 2007.
5. Jackson P., Moulinier I. Natural Language Processing for Online Applications. M: John Benjamins Publishing, 2002. 237 с.
6. Хабаров С. Представление знаний в информационных системах. Тема 5. Семантические сети. [Электронный ресурс]. URL: http://www.habarov.spb.ru/bz/bz05.htm (дата обращения 20.03.2019)
7. Искусственный интеллект как ключевой фактор цифровизации глобальной экономики. [Электронный ресурс]. URL: http://json.tv/ict_telecom_analytics_view/iskusstvennyy-intellekt-ii-artificial-intelligence-ai-kak-klyuchevoy-faktor-tsifrovizatsii-globalnoy-ekonomiki-20170222045241 (дата обращения 20.03.2019)
8. START. Natural Language Question Answering System [Электронный ресурс]. URL: http://start.csail.mit.edu/start-system.html (дата обращения 20.03.2019)
9. Конференция AINL 2013: Искусственный интеллект, естественный язык. Владимир Веселов [Электронный ресурс]. URL: http://nlpseminar.ru/ainl/program/vladimir-veselov/ (дата обращения 20.03.2019)
10. Мозговой М.В. Простая вопросно-ответная система на основе семантического анализатора русского языка. СПб: Вестник СПбГУ, 2005. вып. 2.

11. RAZOOM. Вопросно-ответная система. [Электронный ресурс] URL: http://www.razoom.org (дата обращения 20.03.2019)

12. Вопросно-ответная система ITFRU [Электронный ресурс] URL: http://www.itfru.ru/index.php/qas/qas-itfru (дата обращения 20.03.2019)

13. Жигалов В.А., Соколова Е.Г. InBASE: технология построения ЕЯ интерфейсов к базам данных // Труды Международного семинара Диалог'2001 по компьютерной лингвистике, Том 2, Аксаково, Июнь 2000, с. 123-135.

14. Найханова Л.В., Евдокимова И.С. Методы и алгоритмы трансляции естественно-языковых запросов к базе данных в SQL-запросы: Монография – Улан-Удэ, изд-во ВСГТУ, 2004. – 148 с.

15. IBM Research. DeepQA Project [Электронный ресурс]. URL: https://www.research.ibm.com/deepqa/deepqa.shtml (дата обращения 21.03.2019)

16. Manning C.D. Foundations of statistical natural language processing. Cambridge: The MIT Press, 1999

17. Ванюшкин А.С., Гращенко Л.А. Методы и алгоритмы извлечения ключевых слов. М: Новые информационные технологии в автоматизированных системах, вып. 19, 2016. С. 85-94.

18. Калиниченко А.В. Сущность проблемы анализа текста в полнотекстовых поисковых системах. Подходы и пути решения. [Электронный ресурс]. URL: http://www.jurnal.org/articles/2010/inf12.htm (дата обращения 21.03.2019)

19. Воронина И.Е., Кретов А.А., Попова И.В., Дудкина Л.В. Функциональный подход к выделению ключевых слов: методика и реализации. // Вестник Воронежского Государственного университета. – 2009. - №1 – С.68-72

20. Коршунов А.В. Извлечение ключевых терминов из сообщений микроблогов с помощью Википедии. Труды Института системного программирования, РАН. 2011. Т.20 С.269-282.

21. НОУ Интуит. Основы генетических алгоритмов. [Электронный ресурс] URL: http://www.intuit.ru/studies/courses/14227/1284/lecture/24168?page=12 (дата обращения 22.03.2019)

22. Гринева М., Гринев М. Анализ текстовых документов для извлечения тематически сгруппированных ключевых терминов. Труды Института системного программирования, РАН. 2009. Т.16 С.155-165.

23. Можарова В.А. Автоматическое извлечение именованных сущностей методами машинного обучения. [Электронны ресурс] URL: https://www.slideshare.net/msucsai/ss-57201175 (дата обращения 22.03.2019)

24. Шалаев М.М. Распознавание именованных сущностей с использованием алгоритмов машинного обучения, основанных на принципе минимальной длины описания [Электронный ресурс] URL: http://seminar.at.ispras.ru/wp-content/uploads/2012/07/shalaev-presentation.pdf (дата обращения 22.03.2019)

25. Андриенко А.С. Выделение именованных сущностей в текстовых документах [Электронный ресурс] URL: http://nauchkor.ru/cloud_storage/documents/587d362f5f1be77c40d588c7.pdf (дата обращения 22.03.2019)

26. Кузнецов С.Д. Исследование и разработка методов выделения и классификации именованных сущностей в системах интеллектуального анализа текстов. [Электронный ресурс] URL: http://www.rfbr.ru/rffi/ru/project_search/o_42384 (дата обращения 22.03.2019)

27. Захаренков А.И., Соколов А.В. Метод извлечения информации из неструктурированных текстов. // Инновации в информационно-аналитических системах: сб. научных трудов. Вып. 5 – Курск: Науком, 2013. – 92 с. ISBN 978-5-4297-0009-0

28. Арчакова Н.А., Каневский Е.А., Боярский К.К. Извлечение низкочастотных терминов из специализированных текстов. [Электронный ресурс] URL: http://ceur-ws.org/Vol-1752/paper24.pdf (дата обращения 22.03.2019)

29. Создание собственной модели извлечения информации из текста с помощью webAPI от Meanotek. [Электронный ресурс] URL:

https://habrahabr.ru/company/meanotek/blog/258211 (дата обращения 22.03.2019)

30.. Манучарян Л.А. Извлечение информации из текста: прогнозирование связей между двумя сущностями. [Электронный ресурс] URL: https://www.science-education.ru/pdf/2011/6/manucharyan.pdf (дата обращения 22.03.2019)

31.Потапенко А. Векторное представление слов и документов [Электронный ресурс]. URL: http://www.machinelearning.ru/wiki/images/2/2a/Seminar_1.pdf (дата обращения: 07.05.2019)

32. Schutze H., Pedersen J. A vector model for syntagmatic and paradigmatic relatedness. // In Making Sense of Words: Proceedings of the conference, pp. 104-113, Oxford, England, 1993.

33. Gibianski A. Cool Linear Algebra: Singular Value Decomposition [Электронный ресурс]. URL: http://andrew.gibiansky.com/blog/mathematics/cool-linear-algebra-singular-value-decomposition/ (дата обращения: 07.05.2019)

34.Белоус В.В., Домников А.С. Анализ интеллектуальных данных в электронных обучающих системах. // М: Инженерный вестник. Изд-во: МГТУ им. Н.Э. Баумана, №12, декабрь, 2013

35.Синтаксис. Словосочетание. Синтаксические связи слов в словосочетании. [Электронный ресурс]. URL: http://russkiy-na-5.ru/articles/436 (дата обращения: 10.05.2019)

36.Подбор ключевых слов (для поисковых запросов) [Электронный ресурс]. URL: https://wordstat.yandex.ru/ (дата обращения: 16.05.2019)

37.6 нормальных форм БД [Электронный ресурс]. URL: http://i-novice.net/6-normalnyx-form-bd (дата обращения 24.05.2016).

38.Лебедев А.А., Москин Н.Д., Варфоломеев А.Г. К проблеме анализа неоднозначности синтаксического разбора предложения. // Петрозаводский Государственный университет, 2016.

39. Draxler C. A powerful Prolog to SQL compiler [Электронный ресурс]. URL: www.ling.gu.se/dialoglab/PeoMariaKod/pl2sql/doc/pl2sql.ps (дата обращения 27.04.2016).

40. Толковый словарь Ожегова (онлайн-версия) [Электронный ресурс] URL: http://slovarozhegova.ru/ (дата обращения: 21.05.2019)

41. Словарь кинематографических терминов [Электронный ресурс] URL: http://www.educationmode.ru/waedus-583-9.html (дата обращения: 21.05.2019)

42. Глоссарий терминов. [Электронный ресурс] URL: http://snimifilm.com/glossary (дата обращения: 21.05.2019)

43. NLPub [Электронный ресурс]. URL: https://nlpub.ru (дата обращения 21.05.2019).

44. Документация по Solarix [Электронный ресурс]. URL: http://www.solarix.ru/grammatical-dictionary-api.shtml (дата обращения 21.05.2019).

45. Документация по HTML Agility Pack.[Электронный ресурс]. URL: http://html-agility-pack.net/?z=codeplex (дата обращения: 21.05.2019)

46. Лорьер Ж.-Л. Системы искусственного интеллекта (перевод с французского) – М:Мир, 1991. – 410-411 с.

47. Документация по SWI-Prolog [Электронный ресурс]. URL: http://www.swi-prolog.org/pack/list?p=sql_compiler (дата обращения 21.05.2016).

48. SWI-Prolog interface for C#. [Электронный ресурс] URL: http://www.swi-prolog.org/contrib/CSharp.txt (дата обращения: 20.05.2016)

49. Панченко А., Филиппович Ю.Н. Построение семантической сети из разнородных данных. Критерии оценки качества семантической сети. [Электронный ресурс]. URL: http://it-claim.ru/Persons/Panchenko/presentation2010_sept_final.pdf (дата обращения: 28.05.2019)

50. Нековаль С. Немного о Precision и Recall. [Электронный ресурс]. URL: http://blog.gramant.ru/2012/06/06/f1-measure/ (дата обращения: 28.05.2019)

51. NWord2Vec. C# library for working with Word2Vec models. [Электронный ресурс]. URL: https://github.com/richorama/NWord2Vec (дата обращения: 28.05.2019)