МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра информационных систем управления

Исаченко
Дмитрий Александрович

CROSS-LANGUAGE ФУНКЦИОНАЛЬНОСТЬ АВТОМАТИЧЕСКОГО ПОИСКА В СЕТИ INTERNET РЕЛЕВАНТНЫХ ДОКУМЕНТОВ

Дипломная работа

Научный руководитель**:**
доктор технических наук,
профессор И.В. Совпель

Рецензент:
доктор технических наук,
гл.н.с. ГНУ «ОИПИ НАН БЕЛАРУСИ»
С.Ф. Липницкий

Допущена к защите

«___» _____ 2017 г.

Зав. кафедрой информационных систем управления
доктор технических наук, профессор В. В. Краснопрошин

Минск, 2017

ОГЛАВЛЕНИЕ

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ И СИМВОЛОВ

ЕЯ - естественный язык;

ПОД - поисковой образ документа;

Стоп-слова - слова, не несущие в себе смысловой и содержательной нагрузки, такие как междометия, предлоги и прочие;

Стемминг - процесс нахождения основы слова для заданного исходного слова;

TF– частота термина в контексте определённого документа;

IDF - обратная частота термина в корпусе документов;

IR – информационный поиск;

CLIR - разновидность информационного поиска, при котором язык извлечённой информации может отличаться от языка исходного запроса;

WSD - проблема, связанная с разрешением лексической многозначности.

РЕФЕРАТ

Дипломная работа, 51 с., 11 рис., 3 табл., 20 источников.

Ключевые слова: ПОИСК, ЕСТЕСТВЕННЫЙ ЯЗЫК, АВТОМАТИЧЕСКАЯ ОБРАБОТКА ТЕКСТОВ, ПЕРЕВОД, РАЗРЕШЕНИЕ ЛЕКСИЧЕСКОЙ МНОГОЗНАЧНОСТИ, МАШИННОЕ ОБУЧЕНИЕ, ТЕЗАУРУС.

Объект исследования – алгоритмы извлечения ключевой информации из документа, алгоритмы разрешения лексической многозначности при переводе.

Цель работы – исследование методов поиска релевантных документов в многоязычной информационной среде, разработка мобильного клиента для поиска актуальных данному документов, в том числе на отличных от исходного языках.

Методы исследования – методы теории вероятности, математической статистики, интеллектуальный анализ данных, машинное обучение.

Результатом является выполненный обзор существующих решений в области поиска релевантных документов в многоязычной информационной среде, разработанное мобильное приложение под ОС Android, позволяющее для некоторого документа/веб-страницы выполнить поиск релевантной информации в сети Internet, в том числе на отличном от исходного языке.

Область применения: информационный поиск.

РЭФЕРАТ

ABSTRACT

ВВЕДЕНИЕ

В наше время огромное количество информации доступно в электронном виде. Информационные системы, оперирующие большими объемами данных произвольной предметной области и успешно решающие различные прикладные задачи, становятся все более востребованными, как предприятиями и организациями, так и отдельными пользователями.

Информационный поиск(IR) представляет собой процесс извлечения релевантной информации среди огромного количества документов. Традиционные IR системы реализуются в основном для документов, написанных на одном языке, хотя интернет сам по себе является многоязычной информационной средой. По этой причине возникает языковой барьер между пользователем и доступной информацией, а также появляется необходимость в исследовании и разработке методов для повышения эффективности IR.

В большинстве случаев при поиске информации в интернете мы хотим, чтобы она была написана на нашем родном языке, однако такая информация не всегда является доступной. С учётом того, что большинство пользователей владеет одним или несколькими иностранными языками, они могут быть также заинтересованы в поиске информации, написанной на других языках. Так появляется необходимость в многоязычном поиске(CLIR), целью которого является сопоставления запроса, написанного на одном языке, с документами, написанными на других языках. CLIR снимает языковой барьер, благодаря чему пользователи могут отправлять запросы, написанные на их родном языке, а получать документы на других языках и наоборот. Например, запрос на русском языке вернёт релевантную информацию на английском языке. Из-за быстрого развития интернет-технологий потребность в CLIR значительно растёт, поскольку данный тип поиска позволяет реализовать обмен информацией между различными языками, устранить лингвистическое несоответствие между предоставляемыми запросами и документами, которые извлекаются из информационной сети. В связи с этим CLIR приобрел большое значение, как в качестве исследовательской дисциплины, так и в качестве технологии, которая будет востребована на рынке.

В дополнение к проблемам, встречаемым при одноязычном IR, в CLIR добавляется ещё одна – проблема перевода. Однако в данном случае перевод будет отличаться от полнотекстового машинного перевода. Причиной этому является отсутствие необходимости быть удобочитаемым для человека, перевод должен просто максимально подходить для поиска соответствующих документов. В основе CLIR могут лежать следующие варианты реализации перевода: перевод запроса, перевод документов, перевод запроса и документов

одновременно. Уже было опубликовано большое количество исследований по теме реализации CLIR. Многие вопросы, связанные с данной темой, также рассматриваются на различных конференциях, например, TREC, NTCIR, CLEF. Каждая из данных конференций охватывает определённые языки: TREC включает в рассмотрение испанский, китайский, немецкий, французский, арабский и итальянский языки; NTCIR включает японский, китайский и корейский языки, а CLEF - французский, немецкий, итальянский, испанский, голландский, финский, шведский и русский.

В дипломной работе сначала приводится описание подходов реализации поиска релевантных документов в одноязычной информационной среде. Затем выполняется анализ техник перевода, а также методов разрешения лексической многозначности для осуществления поиска в многоязычной информационной среде. Итогом проведенного в дипломной работе исследования является разработанное мобильное приложение, обладающее cross-language функциональностью при поиске релевантных документов в сети интернет.

ГЛАВА 1
ИССЛЕДОВАНИЕ ПОДХОДОВ РЕАЛИЗАЦИИ ПОИСКА В СЕТИ ИНТЕРНЕТ РЕЛЕВАНТНЫХ ДОКУМЕНТОВ

Задача автоматизации поиска в сети интернет документов релевантных данному относится к классическим задачам информационного поиска и её можно решать одним из двух следующих способов:

- реализовать собственную поисковую систему при разработке приложения;
- воспользоваться уже существующей поисковой системой при разработке приложения.

1.1 Реализация собственной поисковой системы

Работу поисковой системы можно представить следующим образом:

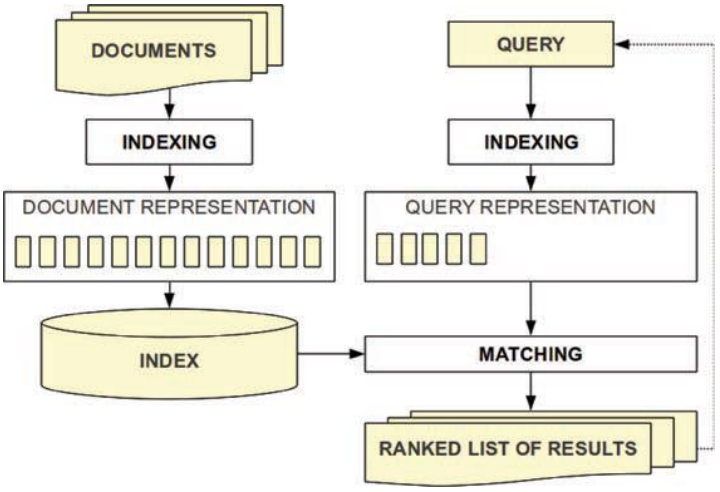

Рисунок 1.1 – Схема работы поисковой системы

Основными её составляющими являются: поисковый робот, индексатор, поисковик.

Поисковый робот — составная часть поисковой системы, основной функцией которой является перебора страниц Интернета. Данный перебор осуществляется для сохранения информации о страницах в базе данных поисковика. Поисковой робот исследует содержимое страницы и затем сохраняет поисковой образ на сервере поисковой машины, которой принадлежит, после этого исследуются следующие страницы, которые доступны по ссылкам с текущей. Большие сайты зачастую проиндексированы поисковой машиной не целиком, так как обычно для поисковой машины глубина

проникновения внутрь сайта и максимальный размер сканируемого текста ограничены. Переходы между страницами реализуются с помощью ссылок, которые содержаться на исходных страницах. В зависимости от алгоритмов информационного поиска определяются порядок обхода страниц и частота визитов, так же предотвращается возможность зацикливания.

Современны поисковые системы дают пользователю возможность ручного добавления сайта в очередь для индексирования, с целью ускорения процесса индексирования сайта. Если на сайт невозможно попасть по внешним ссылкам, то это вообще оказывается единственной возможностью уведомить поисковую систему о существовании сайта.

Для сбора информации о сайте выполняется процесс индексирования, в ходе которого робот поисковой системы помещает в базу данных сведения о сайте (ключевые для сайта слова, ссылки, изображения, аудио…), которые затем используются при поиске. Индексирование страницы осуществляется непосредственно с помощью индексатора, в обязанности которого входит анализ страницу, при этом каждый элемент веб-страницы анализируется отдельно. Полученные индексатором данные о веб-страницах помещаются в индексную базу данных для возможности использования их в последующих запросах. Это существенно ускоряет поиск информации по пользователя.

Поисковый запрос - последовательность символов, которую пользователь вводит в поисковую строку, для обнаружения релевантной информации. Формат поискового запроса зависит от 2-х вещей: от типа информации для поиска и от устройства поисковой системы. Обычно поисковый запрос представляет собой набор слов или фразы.

Работу поисковой системы можно разбить на следующие шаги: сначала исходный контент принимается поисковым роботом, затем согласно контенту, в ходе процесса индексирования определяется доступный для поиска индекс, после чего можно обнаруживать с помощью поисковой системы исходные данные. Данные шаги выполняются каждый раз при обновлении поисковой системы.

В большинстве случаев для поисковых систем основным источником для анализа и получения информации о веб-странице является HTML страница, соответствующая ей. Основное внимание при извлечении информации уделяется заголовкам и метатегам.

Поисковые гиганты, такие как Google, имеют возможность полностью сохранять контент исходной страницы целиком или только часть её(кэш). Последнее позволяется значительно увеличить скорость поиска информации на ранее посещённых страницах(кэшированные). Текст запроса пользователя обычно сохраняется вместе с кэшированной страницей, чтобы сохранить

актуальность в случае обновления исходной. Пользователь формирует запросы для поисковика, который затем обрабатывает их, анализируя данные полученные в ходе процесса индексации, и затем возвращает результаты поиска. Запросы пользователя зачастую представляют собой набор ключевых слов. В тот момент, когда пользователь вводит запрос, поисковая система уже начинает анализировать имеющиеся индексы, после чего пользователь получает наиболее релевантные веб-страниц, также поисковая системы может возвращать их вместе с краткой аннотацией, которая представляет собой заголовок документа и возможно некоторый отрывок из текста. Поисковая система характеризуется следующими двумя оценками: оценка точности найденных релевантных страниц и оценка полноты найденных релевантных страниц. Для того, чтобы в начале списка результатов были наиболее актуальные для пользователя, многие поисковые систем используют методы ранжирования, которые в свою очередь определяют, какие страницы более релевантны, а также очередь отображения результатов.

В связи с огромной трудоёмкостью реализации собственной поисковой системы при разработке приложения будет использоваться поисковая система Google. Таким образом задача автоматизации поиска в сети интернет документов релевантных данному сведётся к формированию поискового запроса. Поисковым запросом для Google будет являться поисковой образ документа, который формируется из ключевых для исходного текста слов. Количество слов в запросе можем варьироваться в зависимости от размера документа. Согласно рекомендациям Google, поисковой запрос должен состоять из ключевых слов, оптимальное количество ключевых слов должно находиться в диапазоне 6-9.

1.2 Поиск релевантных документов в одноязычной информационной среде

1.2.1 Предварительная обработка документа

В ходе предварительно обработки документа происходят следующие действия: токенизация, удаление стоп-слов, стемминг и расширение терминов.

Токенация служит для распознавания и изолирования различных языковых единиц, присутствующих в исходном тексте. Двумя основными процедурами процесса токенизации являются сегментация слов и декомпозиция слов. Сегментация обычно выполняется при работе с восточноазиатскими языками, в то время как декомпозиция с европейскими.

Сегментация - это просто процесс разбития исходного текста на составляющие единицы. Данный процесс легко реализовать для языков, в

которых явно выделены границы слов, например, с помощью пробельного символа в английском и французском языке, но значительно труднее для таких языков как китайский, где разделители между словами отсутствуют.

Один из подходов реализации сегментации использует алгоритм максимального соответствия, используя список известных слов. Очевидно, что такой подход не работает для слов, которые отсутствуют в исходном списке. Альтернативой данному подходу являются подходы, основанный на n-граммах, наиболее распространёнными из которых являются подходы, использующие биграммы.

В некоторых языках, таких как русский и немецкий, часто употребляются сложные слова, которые состоят из нескольких слов и в ходе процесса токенизации должны считаться одной языковой единицей. Для обнаружения сложных слов можно использовать специальные словари, содержащие их список. Текст будет разбит на минимальное количество слов, присутствующих в данном словаре. Если алгоритм обнаружил два (или более) возможных вариантов составного слова в определённом отрывке текста, то должен выбраться наиболее вероятный для данного контекста. Вероятность можно вычислять предварительно, обучив системы на корпусе документов.

Предлоги, местоимения, союзы, общие глаголы и незначащие слова обычно удаляется из исходного текста до составления ПОД. Фильтрация этих терминов осуществляется зачастую с использованием списка стоп-слов.

Список стоп слов для английского языка:

a, about, above, after, again, against, all, am, an, and, any, are, aren't, as, at, be, because, been, before, being, below ,between, both, but, by, can't, cannot, could, couldn't, did, didn't, do, does, doesn't, doing, don't, down, during, each, few, for, from, further, had, hadn't, has, hasn't, have, haven't, having, he, he'd, he'll, he's, her, here, here's, hers, herself, him, himself, his, how, how's, i, i'd, i'll, i'm, i've, if, in, into, is, isn't, it, it's, its, itself, let's, me, more, most, mustn't, my, myself, no, nor, not, of, off, on, once, only, or, other, ought, our, ours, ourselves, out, over, own, same, shan't, she, she'd, she'll, she's, should, shouldn't, so, some, such, than, that, that's, the, their, theirs, them, themselves, then, there, there's, , these, they, hey'd, they'll, they're, they've, this, those, through, to, too, under, until, up, very, was, wasn't, we, we'd, we'll, we're, we've, were, weren't, what, what's, when, when's, where, where's, which, while, who, who's, whom, why, why's, with, won't, would, wouldn't, you, you'd, you'll, you're, you've, your, yours, yourself, yourselves.

Одним из способов повышения эффективности работы систем информационного поиска является предоставление поисковым системам способа обнаружения различных форм одного и того же слова. Для реализации процесса обнаружения различных форм можно воспользоваться стеммерами.

Стемминг также используется в информационном поиске для уменьшения размера индексных файлов. Таким образом сначала для исходного текста выделяются границы слов. Затем для обнаружения различных форм одного и того же слова необходимо выполнить процесс нахождения основы слова для всех исходных слов текста – стемминг. Стемминг представляет собой морфологический разбор слова, в ходе которого обнаруживается общая для всех его грамматических форм основа, обрубаются окончания и суффиксы.

Существует несколько критериев оценки стеммеров: корректность, эффективность поиска и производительность сжатия.

При реализации стемминга нужно найти баланс между следующими двумя проблемами: чрезмерный стемминг, что приводит к объединению несвязанных терминов и соответственно это понижает точность поиска, так как извлекаются нерелевантные документы; основа слова выделяется слишком слабо, в связи с чем будет понижаться полнота поиска.

В зависимости от необходимой точности/полноты поиска, а также скорости работы можно выбрать один из следующих стеммеров.

Для английского языка на данный момент одним из самых распространённых стеммеров является стеммер Портера в силу его быстрой скорости работы, отсутствия необходимости в предварительной обработке корпуса документов и использования каких-либо баз основ. В основе данного стеммера лежит алгоритм усечения окончаний, использующий для своей работы небольшой набор правил, например, если слово оканчивается на “ет”, то удалить “ет” и так далее. Алгоритмы усечения окончаний достаточно эффективны на практике, но в то же время обладают некоторыми недостатками. Алгоритмы усечения окончаний неэффективны в случае изменения корня слова, например, изменения или выпадения гласной. Данные алгоритмы эффективны для тех частей речи, которые имеют хорошо известные окончания и суффиксы. Стеммер Портера основывается на том, что количество словообразующих суффиксов в языках ограниченно. Благодаря этому алгоритм может выполняться с помощью установленных вручную определённых правил. Алгоритм выделения основы слова стеммера Портера для английского языка включает в себя пять шагов. На каждом их которых проверяется будет ли получившаяся в результате убирания словообразующего суффикс часть соответствовать заранее установленным правилам. В случае, если правила удовлетворены осуществляется переход на следующий шаг алгоритма, иначе выбирается другой суффикс для отсечения. Из описания хода работы алгоритма видно, что у стеммера Портера существует недостаток: он может обрезать слово больше необходимого, что в свою очередь затруднит получение правильной основы слова и соответственно уменьшит точность извлечение релевантной информации. Ещё одним недостатком

стеммера Портера является отсутствие возможности работать при изменении корня слова, например, в случае выпадающих беглых гласных.

Также можно воспользоваться стеммером, использующим таблицы поиска флективных норм. Трудностью при реализации данного стеммера является необходимость перечислять все флективные формы в таблице. Если какая-то из форм будет отсутствовать, то она обрабатываться не будет. В связи с этим получается, что таблица поиска может иметь большой размер. В качестве плюсов можно выделить простоту подхода, скорость работы и простоту обработки исключений. Таблицы поиска, которые используются в стеммерах, обычно генерируются в полуавтоматическом режиме. Чтобы избежать проблемы, когда разные слова относятся к одной лемме (ошибка лемматизации), при реализации алгоритма поиска можно использовать предварительную частеречную разметку.

Можно улучшить проход к выделению основы слова посредством определения части речи слова и затем в зависимости от результата применения соответствующих для каждой части речи правил нормализации.

Основной недостаток классических стеммеров – они не различают слова, имеющие схожий синтаксис, но абсолютно разные значения, например, в английском языке “news” и “new” для данных стеммеров будут различными формами одного и того же слова. С целью разрешения данных проблем было реализованы стеммеры на основе корпусов текстов. Ключевой идеей стеммеров на основе корпусов является создании классов эквивалентности для слов классических стеммеров, которые после разделят некоторые слова, объединенные на основе их встречаемости в корпусе. Такие алгоритмы работают с базой данных основ, которые не обязательно соответствуют обычным словам и зачастую представляют собой. Для определения основы слова алгоритм сопоставляет его с основами из базы данных, используя различные ограничения, такие как длина искомой основы в слове относительно длины самого слова и т.п.

1.2.2 Составление поискового образа документа

Поисковый образ документа(ПОД) - текст, выражающий на информационно поисковом языке основное содержание документа и в последующем используемый для информационного поиска. Для формирования ПОД необходимо выделить из документа ключевую информацию.

Любой алгоритм извлечения ключевых слов/словосочетаний реализует одну или несколько систем распознавания образов, разбивающих входное множество слов на два класса (ключевые и прочие). По наличию элементов обучения выделяют необучаемые, обучаемые и самообучаемые методы извлечения ключевых слов. Более простые необучаемые методы подразумевают

контекстно-независимое выделение ключевых слов/словосочетаний из отдельного текста на основе априорно составленных моделей и правил. Они подходят для гомогенных по функциональному стилю корпусов текстов, увеличивающихся со временем в объемах, например, научных работ или нормативных актов. Обучаемые методы предполагают использование разнообразных лингвистических ресурсов для настройки критериев принятия решений при распознавании ключевых слов. Здесь большое значение имеет корректное выделение ключевых слов в выборке, используемой для обучения. Среди методов с обучением можно выделить подкласс самообучаемых, если обучение ведется без учителя или с подкреплением (на основе пассивной адаптации). По второму признаку классификации, прежде всего, следует выделить статистические и структурные методы извлечения ключевых слов. Статистические методы учитывают относительные частоты встречаемости морфологических, лексических, синтаксических единиц и их комбинаций. Это делает создаваемые на их основе алгоритмы довольно простыми, но недостаточно точными, т.к. признак частотности ключевых слов не является превалирующим.

Для извлечения ключевых словосочетаний из текста выполняется анализ коллокаций, которые обнаруживаются во время лексического анализа текста.

Коллокация состоит из нескольких слов, представляющих собой синтаксически и семантически целостную единицу. При извлечении коллокаций анализируют является ли появление лексических единиц случайным или нет.

В нашем случае ПОД, будет состоять из ключевых слов исходного документа, и являться запросом для поисковой системы Google.

1.2.3 Анализ методов извлечения ключевых слов

Существуют следующие категории методов выделения ключевых слов: статистические, лингвистические, и гибридные, которые являются их комбинацией.

В основе лингвистических методов лежат значения слов, семантические данные о слове, а также используются онтологии, которые формализуют знания из некоторой области с помощью концептуальной схемы. При использовании данных подходов возникает трудность, связанная с реализацией онтологий, что само по себе очень трудоёмкий процесс. Часть операций, которая при лингвистическом анализе текстов выполняется вручную, усложняют процесс анализа документов из-за дополнительной возможности возникновения ошибок и неточностей.

Наиболее популярными лингвистическими методами при обработке естественного языка являются лингвистические методы в основе которых лежат графы. Главная задача данных методов представляет собой построение семантического графа. Семантический граф является взвешенным графом. Термины исходного документа будут вершинами в графе. Между вершинами графа есть ребро в том и только в том случае, если присутствует семантическая связь между терминами. Вес в семантическом графе равен значению семантической близости связанных ребром терминов. Поиск ключевых слов осуществляется с помощью алгоритмов обработки графа. Определяющими характеристиками лингвистических методов, основанных на графах, являются способ отбора множества терминов, а также алгоритм определения весов рёбер (семантической близости терминов

Статистические методы базируются на численных данных о встречаемости слова в тексте. Основными их преимуществами являются относительная простота реализации, универсальность алгоритмов поиска ключевых слов, а также не нужно выполнять трудоемкие операции построения лингвистических баз знаний. Максимальную точность и полноту имеют алгоритмы, в основе которых лежат статистические исследования корпусов документов. Алгоритмы, которые предварительно не обрабатывают никаких документов, кроме того, ключевые слова которого необходимо извлечь, обладают сравнительно более низкой точностью. Классическими подходами в области статистической обработки естественного языка можно считать использование метрики TF-IDF и ее модификаций при поиске ключевых слов, а также анализ коллокаций при поиске ключевых словосочетаний. Одним из самых простых статистических методов выделения ключевых слов в тексте является построение множества кандидатов путем ранжирования по частоте встречаемости в исходном документе всех его словоформ или лексем. Фильтрация в данном случае осуществляется через отбор в качестве ключевых наиболее частотных словоформ/лексем.

Если в качестве параметра для автоматического обнаружения ключевых слов использовать только частоты слова в документе, то в данном случае вычисление частоты словоформ реализуется следующим образом: полученная в результате частота ключевых слов вычисляется посредством сравнения словоформ, приведённых к одной форме, как правило, к основе или лемме. Выделение основы у словоформы представляет собой разновидность задачи морфологического анализа, которая является достаточно трудоёмкой. При реализации статистических подходов для поиска ключевых слов задействованы различные эвристические алгоритмы, обычно приводящие словоформу к ее квази-основе, что достигается посредством выделения у словоформы некоторого

количества букв. Данные алгоритмы(стемминг-алгоритмы) обсуждались выше при описании предварительной обработке документа. В ходе алгоритмов стемминга выделялись основы слов, которые затем ранжировались по частоте. Словоформы с наибольшей частотой считаются ключевыми. Статистические методы, обученные для повышения точности поиска ключевых слов на корпусе текстов, достаточно популярны. Но в тоже время необходимо наличие таких корпусов для каждой определённой предметной области, что значительно затрудняет возможность данных методов. С целью повышения точности описания контента документа разрабатываются методики, у которых мерой релевантности является вес лексемы, полученный посредством определённой комбинации значений различных параметров лексем, таких как, расположения в тексте, статистика совместной принадлежности слов одному и тому же документу и т.п.

Положительными сторонами использования статистических методов является универсальность и относительная простота реализации алгоритмов извлечения ключевых слов, которая связана с тем что не нужно выполнять трудоемкие и занимающие огромное количество времени операции для создания лингвистических баз знаний. Однако методы выделения ключевых слов, в основе которых лежит только статистический подход иногда не обеспечивают желаемого качества результатов, особенно невысокие результаты получаются при работе с языками с богатой морфологией, например, с русским языком, в котором лексемы характеризуются огромным количеством словоформ с невысокой частотностью в отдельно рассматриваемом тексте.

Для оценки важности слова в контексте документа рассмотрим более подробно статистическую меру TF-IDF, которая является произведением двух статистик: частоты термина в данном документе и обратной частоты термина в корпусе документов. Существуют различные способы определения данных статистик.

Введём следующие обозначения:

1. D - корпус документов;
2. N - размер корпуса документов;
3. t - термин, важность которого хотим определить в документе d;
4. n(t) = 1 + количество документов,в которых встречается t;
5. f(t,d) = частоте термина t в документе d.

Способы определения статистики TF:

- по частоте встречаемости (raw frequency) формула (1.1);
- логический (boolean frequency) формула (1.2);
- логарифмически нормализованный (logarithmically scaled frequency) формула (1.3);

- нормализованный по максимальной частоте слова (augmented frequency) формула (1.4).

$$tf(t,d) = f(t,d) \tag{1.1}$$

$$tf(t,d) = \begin{cases} 1, & f(t,d) > 0 \\ 0, & f(t,d) = 0 \end{cases} \tag{1.2}$$

$$tf(t,d) = \begin{cases} 1 + \log(f(t,d)), & f(t,d) > 0 \\ 0, & f(t,d) = 0 \end{cases} \tag{1.3}$$

$$tf(t,d) = 0.5 + \frac{0.5 * f(t,d)}{\max\{f(t',d) : t' \in d\}} \tag{1.4}$$

Способы определения статистики IDF представлены формулами (1.5) - (1.8).

$$idf(t,D) = 1 \tag{1.5}$$

$$idf(t,D) = \log\left(\frac{N}{1 + n(t)}\right) \tag{1.6}$$

$$idf(t,D) = \log\left(\frac{\max\{n(t'), t' \in d\}}{1 + n(t)}\right) \tag{1.7}$$

$$idf(t,D) = \log\left(\frac{N - n(t)}{n(t)}\right) \tag{1.8}$$

Различные варианты схемы взвешивания TF-IDF часто используются поисковыми системами в качестве основного инструмента при ранжировании по релевантности документов для данного поискового запроса. Так же TF-IDF может быть успешно использован при фильтрации стоп-слов в различных предметных областях.

Для повышения точности автоматического обнаружения ключевых слов в тексте используются гибридные методы, представляющие собой комбинацию статистических методов обработки документов, дополненных несколькими лингвистическими процедурами, такими как морфологический, синтаксическим, и семантический анализ, а также различными лингвистическими базами знаний. В основе гибридных методов поиска ключевых слов в документе, может лежать обучение на корпусе текстов. Например, метод Кена Баркера, осуществляет поиск в исходном тексте базовых именных групп(БИГ) посредством морфосинтаксического анализа с использованием словарей и расчётом релевантности БИГ. Именные группы, обладающие показателем релевантности

выше заданного порога, относятся к ключевым. Одной из разновидностей гибридных методов поиска ключевых слов являются методы на основе машинного обучения, в которых выделение ключевых слов представляет собой задачу классификации. Как известно, для построения обучающей выборки, по которой будет обучен классификатор, необходимы корпуса документов, в которых выделены ключевые слова. Выделенные ключевые слова играют роль положительного примера, остальные слова – отрицательного примера. После этого для всех слов тренировочного текста вычисляется их релевантность, посредством сопоставления каждого из слов с вектором значений различных параметров. Запоминается разница между значениями векторов данных параметров для ключевых и не являющихся таковыми слов. Затем происходит обучение модели посредством расчёта вероятности принадлежности каждого слова к группе ключевых и задания соответствующего порога. Поиск ключевых слов во входном документе осуществляется с помощью классификатора, путем расчёта актуальности слов в соответствии с построенной моделью.

Проанализировав вышеописанные методы, было замечено, что схема выделения ключевых слов в тексте схожа для каждого из них и её можно разбить на следующие шаги:

1. Предварительная обработка текста, призванная представить текст в формате, удобном для последующего распознавания. В неё входят следующие операции: фильтрация из исходного текста стоп-слов, не несущих смысловой нагрузки (предлоги, союзы, частицы, местоимения, междометия и т. д.), выделение основы слова;
2. Отбор кандидатов: выделяются все возможные слова, фразы, термины или понятия (в зависимости от поставленной задачи), которые потенциально могут быть ключевыми;
3. Анализ свойств: для каждого кандидата нужно вычислить свойства, которые указывают, что он может быть ключевым. Например, кандидат, появляющийся в названии книги, скорей всего является ключевым;
4. Отбор ключевых слов из числа кандидатов, посредством вычисления весов важности ключевых слов/словосочетаний в контексте документа.

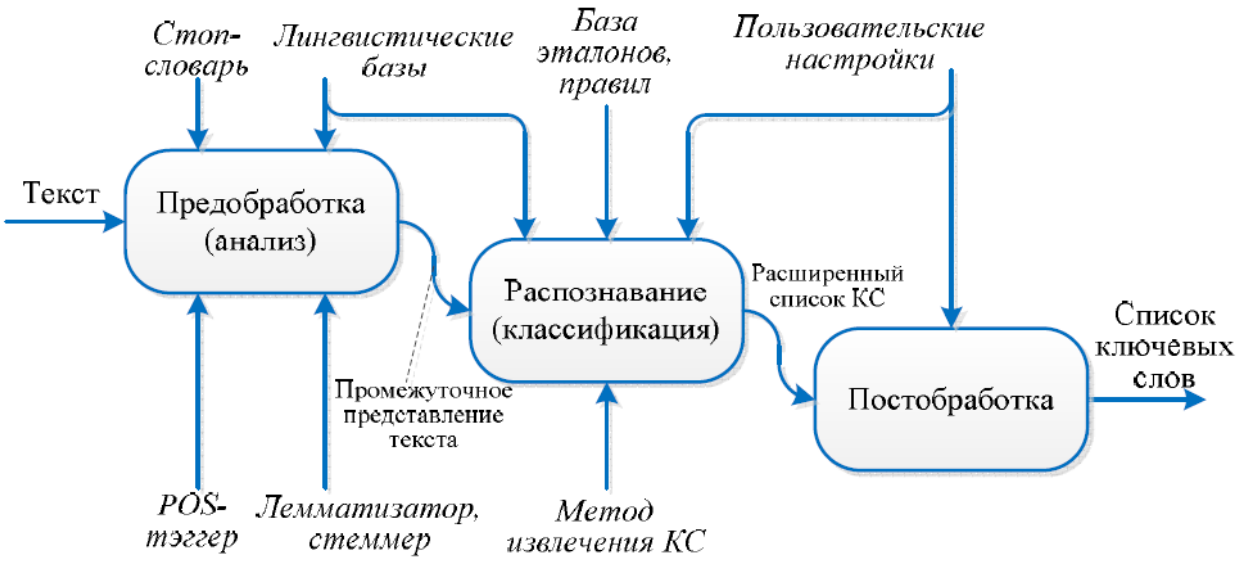

Рисунок 1.2 – Процесс извлечения ключевых слов

В связи с трудоёмкостью реализации собственного лингвистического процессора и недостаточной точностью методов выделения ключевых слов, в основе которых лежит только статистический подход, в данной работе для выделения ключевых слов при формировании поискового запроса воспользуемся сторонним сервисом, использующим гибридный подход для извлечения ключевых слов. Анализ существующих сервисов наиболее популярных в IT сообществе для решения данной задачи будет приведён в следующей главе.

1.3 Поиск релевантных документов в многоязычной информационной среде

При поиске информации в многоязычной информационной среде необходимо сопоставлять запросы и документы, написанные на разных языках. Для разрешения несоответствия языков используется перевод запроса и/или документов перед выполнением поиска. Поэтому правильность перевода одна из главных задач при CLIR.

При разработке собственной поисковой системы, поддерживающей обнаружение информации на языке отличной от языка запроса, можно было бы перевести все имеющиеся документы на все возможные языки запросов.

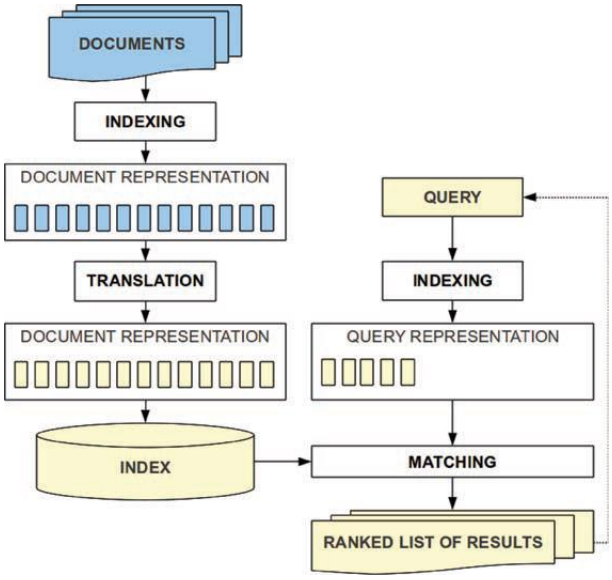

Рисунок 1.3 – CLIR с переводом документов

Данный подход является вычислительно затратным, а также возникает необходимость хранить переводы всех документов системы на всевозможные языки. Следующий подход реализации многоязычного поиска предлагает все документы и поисковой запрос переводить на промежуточный язык.

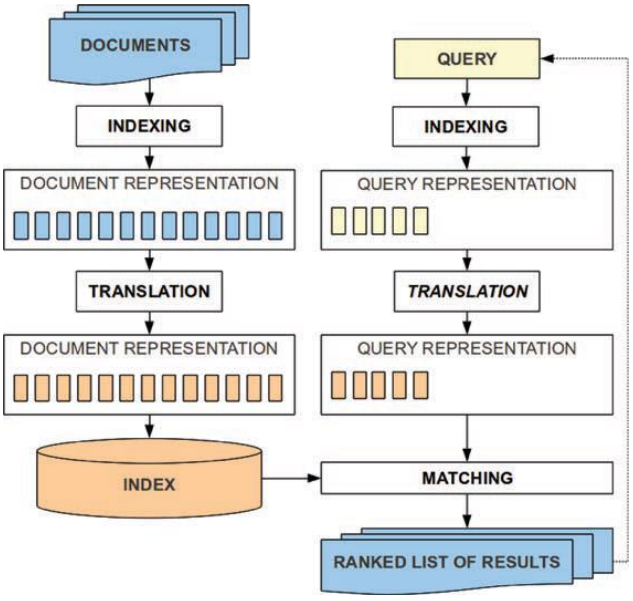

Рисунок 1.4 – CLIR с переводом документов и запроса на промежуточный язык

Так же существует 3-й подход, основанный только на переводе запроса и являющийся наиболее актуальным для нас, так как мы не будем разрабатывать

собственную поисковую систему, а воспользуемся уже существующей. Данный подход, в то же время, является наиболее предпочтительным для реализации с точки зрения CLIR сообщества.

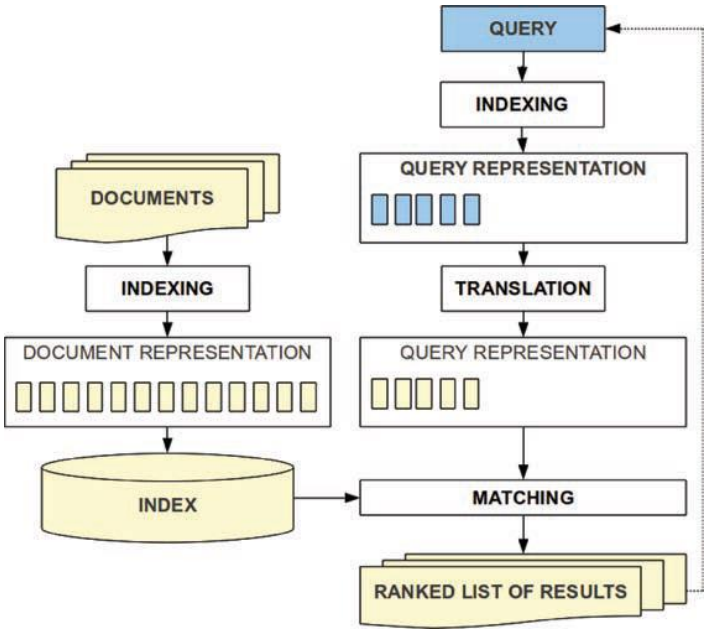

Рисунок 1.5 – CLIR с переводом запроса

Согласно схеме изображённой выше(см. Рисунок 1.5) в нашем случае запросом будет является исходный текст, процесс индексации – процесс построения поискового образа документа. Сравнительная характеристика 3-х подходов к реализации перевода приведена ниже (см. таблица 1.1).

Параметры	**Перевод запроса**	**Перевод документа**	**Перевод документа и запроса**
Неоднозначность	Высокая	Низкая	Средняя
Дополнительное пространство для хранения	Не требуется	Необходимо	Не требуется
Время перевода	Низкое	Высокое	Высокое
Поиск информации	Двуязычный	Двуязычный	Двуязычный и многоязычный
Гибкость	Высокая	Низкая	Низкая

Таблица 1.1 - Сравнение трех подходов к переводу

Таким образом для исходного текста сначала будем составлять ПОД, а затем выполнять его перевод. Для разрешения лексической многозначности слов при переводе будем использовать тезаурус синсетов.

1.3.1 Лексические базы данных, Wordnet

Тезаурус – словарь, охватывающие понятия, определения и термины специальной области знаний. Слова в тезаурусах упорядочены по смысловой близости, не по алфавиту.

Наиболее распространёнными типами смысловых отношений между словами в тезаурусах являются:

- синонимия, базирующаяся на критерии, что два выражения являются синонимичными в том случае, когда замена одного из них на другой в предложении не изменяет смысл данного предложения, например, быстрый – шустрый, бортпроводница – стюардесса;
- антонимия, основанная на смысловом противопоставлении, например, тёплый – холодный, светло – темно;
- гипо-гиперонимия, представляющая собой отношение общего и частного, например, машина – самосвал;
- меронимия, т.е. отношение часть-целое, например, компьютер – процессор, тетрадь – страница.

Синсетом называется множество слов, связанных отношением синонимии. Синсеты разбивают множество всех лексических единиц на классы эквивалентности. Если для некоторого слова не существует синонимов, то соответствующий ему синсет будет состоять только из одного слова. При работе со словом учитываются все его значения, особенно те, в которых это слово является синонимом к другим словам. Многозначные слова, рассматриваемые в разных значениях, входят и в разные синсеты: золотая (монета) – сделанная из золота и золотой (работник) – хороший.

WordNet - это огромная лексическая база знаний для английского языка. WordNet является семантической сетью, узлы которой представляют собой синсеты, связанные различными отношениями, такими как гипонимия, гиперонимия, голонимия, меронимия и т.п. WordNet приобрёл популярность благодаря его содержательным и структурными характеристиками. Принстонский WordNet и все последующие варианты для других языков направлены на отображение состава и структуры лексической системы языка в целом, а не отдельных тематических областей. Для каждого синсета имеется описание на естественном языке, а так же примеры использования входящих в него слов. Лексемы, входящие в состав тезауруса, могут относиться к четырем частям речи: существительное, прилагательное, наречие и глагол. Лексемы

различных частей речи хранятся отдельно, и описания, соответствующие каждой части речи, имеют различную структуру.

Существительные, прилагательные, глаголы, наречия сгруппированы в наборы когнитивных синонимов (синсеты), каждый из которых выражает отдельное значение. Синсеты взаимосвязаны между собой посредством концептуально-семантических и лексических отношений. WordNet является свободно распространяющейся и соответственно общедоступной для загрузки базой знаний. Поэтому WordNet предсиваляет собой полезный инструмент для вычислительной лингвистики и обработки естественного языка.

Слова в WordNet группируются вместе на основе их значений. WordNet внешне напоминает тезаурус, однако есть некоторые важные различия. Во-первых, WordNet связывает не только словоформы, но и слова со схожим смыслом. Во-вторых, WordNet отмечает семантические отношения между словами, тогда как группировки слов в тезаурусе не следуют какой-либо явной схеме, кроме сходства.

Основным отношением между словами в WordNet является синонимия. Синонимы - слова, которые обозначают одну и ту же концепцию и являются взаимозаменяемыми во многих контекстах. Они группируются в неупорядоченные наборы (синсеты). Каждый из 117 000 синсетатов WordNet связан с другими синсетами с помощью небольшого числа смысловых отношений. Кроме того, синсет содержит краткое определение и, в большинстве случаев, одно или несколько коротких предложений, иллюстрирующих использование слов из данного синсета. Формы слов с несколькими различными значениями представлены в виде множества различных синсетов.

Синонимы обязаны быть взаимозаменяемы хотя бы в некотором непустом множестве контекстов. Для отношения синонимии не требуется заменимость всех синонимов во всех контекстах, в противном случае количество синонимов было бы слишком малым в языках. Существительные в WordNet могут иметь следующие семантические отношения: синонимия, антонимия, гипонимия/гиперонимия, меронимия.

Наиболее часто встречающимся отношением между синсетами является гиперонимия и гипонимия. Гиперонимия связывает более общие синсеты, такие как мебель, с более специфическими, такими как кровать. Таким образом, согласно WordNet в категорию мебели входит кровать, которая, в свою очередь, включает в себя двухъярусную кровать. Наоборот, понятия типа кровати и двухъярусной кровати составляют категорию мебели. Все иерархии существительных в конечном счете поднимаются на корневой узел. Отношение гипонимии является переходным: если кресло является своего рода стулом, а стул есть мебель, то кресло является своего рода мебелью. Синсет А –

гипоним синсета B, в том случае, когда существуют предложения типа A есть (является разновидностью) B. И соответственно наоборот Синсет A – гипероним синсета B, в том случае, когда существуют предложения типа A имеет разновидность B.

Меронимия или другими словами отношение «часть-целое» имеет место между синсетами, такими как, например, стул и спинка, стул и ножки. В WordNet определены три подвида отношения часть-целое: быть частью, быть элементом, быть сделанным из. Части у различных сущностей могут иметь одинаковое название, например, острие может быть у иголки, карандаша, стрелы, ножа, булавки и т.д. Таким образом A является меронимом B в том случае, если предложения вида A содержит B и A является частью B являются естественными для A и B, интерпретируемых как родовые понятия.

Так же в WordNet выделяют 2 категории глаголов согласно их смысловому значению: глаголы, обозначающие действия (действия и события), и глаголы состояния. Среди глаголов действий и событий выделяют следующие 14 групп: контакта, движения, коммуникации, восприятия, изменения, соревнования, познания, создания, эмоций, потребления, обладания, относящиеся к социальному поведению и глаголы ухода за телом. Однако, в связи с тем, что нельзя однозначно отнести многие глаголы к той или иной группе, границы между группами точно не установлены. Отношение логического следования устанавливается между синсетами глаголов A и B, если из того что выполняется A, следует, что выполняется B. Например, из того, что девушка говорит, следует, что девушка издаёт звуки.

Для установления иерархических отношений между глаголами было введено отношение тропонимии. То есть делать A означает делать B в определённой форме. Например, “Шептать – это тихо разговаривать". Отношение тропонимии – особый вид отношения следования. Отношение причины связывает два глагольных синсета, один из синсетов называется результатив, а второй каузатив. Отношение причины также может являться особым случаем отношения следования. Если A влечёт за собой B, то из B также логически следует A.

Большинство отношений WordNet связывают слова, являющиеся одной частью речи. Таким образом можно сказать, что WordNet действительно состоит из четырех подсетей, по одной для существительных, глаголов, прилагательных и наречий, с несколькими перекрестными POS-указателями.

В данной работе знания, полученные из тезаурусов, будут использоваться для разрешения лексической многозначности при переводе, что позволит избежать самостоятельного обучения системы на большом корпусе размеченных

документов. Более подробно алгоритм использования тезаурусов при переводе будет описан в главе 2.

1.4 Постановка задачи

Требуется разработать мобильное приложение под ОС Android, снимающее с пользователей языковой барьер при поиске интересующей его информации в интеренете.

Для решения сформулированной задачи в силу проведенных исследований необходимо решить следующие основные подзадачи:

1. Исследовать существующие сервисы/инструменты для извлечения ключевой информациии из текста, с которыми впоследствии будет взаимодействовать приложение;
2. Исследовать существующие алгоритмы для разрешения лексической многозначности при переводе слов, проанализировать преимущества и недостатки каждого из них;
3. Разработать мобильное приложение на основании 1 и 2 пункта. Приложение должно обладать удобным для пользователя интерфейсом.

1.5 Выводы

В ходе исследования предметной области пришли к следующим заключениям:

1. При реализации CLIR можно разработать собственную поисковую систему, а можно по входным для поиска данным формировать запрос для уже существующих поисковых систем;
2. Для извлечения ключевой информации из текста используются статистические, лингвистические и гибридные методы. Каждый из них обладает определёнными преимуществами, связанными с оценками точности и полноты извлечения, а также простотой реализации;
3. Правильность перевода одна из главных задач при поиске релевантных документов на языках, отличных от языка используемого запроса.

ГЛАВА 2
АЛГОРИТМИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ЗАДАЧИ

Для начала необходимо установить, что будет являться входными и выходными данными для разрабатываемой системы поиска релевантных документов в сети интернет, обладающей cross-language функциональностью.

Входные данные - текст либо адрес веб-страницы. Язык исходного текста/веб-страницы будет принадлежать множеству языков, равному пересечению множества языков, для которых наша система умеет извлекать ключевые слова, и множеству языков, поддерживаемых многоязычной лексической базой данных, которая будет использоваться при реализации системы.

Выходные данные - набор запросов для поисковой системы Google. Языки поисковых запросов будут принадлежать множеству языков, поддерживаемых многоязычной лексической базой данных, которая будет использоваться при реализации системы.

2.1 Описание алгоритма

Осуществлять поиск релевантных документов в многоязычной информационной среде будем согласно следующему алгоритму:

1. Для исходного документа определить язык;
2. Выполнить предварительную обработку документа: токенизация, стемминг, фильтрация стоп слов;
3. Определить для токенов важность их в контексте соответствующего документа;
4. Исходя из рассчитанных весов важности определить наборы ключевых слов;
5. Для полученных на предыдущем шаге ключевых слов выделить контексты их употребления;
6. Для всех слов каждого из контекстов извлечь из многоязычного тезауруса соответствующие им толкования, синонимы, а также примеры их употребления;
7. Выполнить стемминг всех слов, извлечённых из тезауруса, алгоритм стемминга зависит от определённого на шаге 1 языка;
8. С помощью алгоритма Леска разрешить многозначность при переводе ключевых слов на другие языки;
9. По составленным наборам ключевых слов сгенерировать поисковые запросы для Google;
10. Используя поисковые запросы, найти релевантные данному документы в сети интернет.

2.2 Сравнение сервисов и инструментов для извлечения ключевой информации из текста

Рассмотрим существующие сервисы/инструменты популярные в IT сообществе, позволяющие извлекать ключевую информацию из текста.

OpenCalais представляет собой web-сервис, разработанный компанией Thomson Reuters, который позволяет извлекать из текстов на естественном языке семантические метаданные. Он является бесплатным и также доступен для коммерческого использования. Семантическими метаданными являются именованные сущности вместе с относящимися к ними фактами. В основе OpenCalais лежат методы обработки естественного языка, а также заранее подготовленные онтологии для различных предметных областей и машинное обучение. Первоначально над входным текстом выполняется графематическая и морфологическая разметка, затем полученные в ходе разметки словосочетания проходят идентификацию посредством обученной модели классификации именованных сущностей, между которыми осуществляется поиск семантических отношений. Полученный в результате граф сущностей и отношений между ними конвертируется в набор RDF-троек. Поддерживаемые языки: английский, французский и испанский. Ограничения на передаваемый размер файла 100кб, 50000 запросов в сутки, до 4 запросов в секунду по одному ключу.

IBM's Watson Natural Language Understanding Service предоставляет возможность анализа текста на естественном языке для извлечения семантических метаданных. Входными данными для данного сервиса может являться как обычный текст, html, так и url-адрес некоторого веб-сайта. Сервис предварительно очищает HTML перед анализом, удаляя большинство рекламных объявлений и другой нежелательный контент. Поддерживает языки: английский, французский, немецкий, итальянский, португальский, русский, испанский. Ограничения для бесплатной версии: 1000 баллов в сутки, 1 балл – 1 запрос размером до 10000 символов и до 50 кб. Запросы превышающие по размеру ограничения стоят большее количество баллов.

Yahoo Content Analysis (ранее Yahoo! Term Extraction Web Service) — сервис, задействованный в работе поисковой системы Yahoo! Search. Имеет возможность обнаруживать ключевые фразы из текста на естественном языке. Подход к извлечению терминов в документации не описан. Обмен данными с пользователем осуществляется в форматах XML и JSON. Ограничение - 5000 запросов в сутки, сервис также не доступен для коммерческого использования.

Extractor набор инструментов разработчика для автоматического извлечения терминов. Предназначен для обработки естественного языка. В

основе системы Extractor, согласно документации, лежит машинное обучение, генетические алгоритмы, а также статистические методами обработки естественного языка. Перед использованием систему нужно обучить на корпусе текстов, который предварительно был размечен.

Mining Cloud (ранее Text Analytics) — сервис, предназначенный для поиска информации и анализа содержания текстов, в основе которого лежат методы обработки естественного языка, а также методы машинного обучения. Mining Cloud позволяет пользователям встраивать текстовую аналитику и семантическую обработку в любое приложение или систему достаточно простым способом благодаря облачной инфраструктуре, с которой легко интегрироваться. Mining Cloud предоставляет следующую функциональность: извлечение темы, посредством распознавания именованных сущностей тексте; классификация текстов через присваивание им одной или нескольких категорий в предопределенной таксономии (сервис включает несколько стандартных таксономий классификации из коробки); определение эмоциональной окраски (положительная, отрицательная, нейтральная) документа или его отдельных частей. Сервис также предлагает расширенные API-интерфейсы, такие как дополнительные тезаурусы, таксономии и т.п., оптимизированные для разных отраслей и сценариев приложений. Большинство данных API доступны на следующих языках: английском, испанском, французском, итальянском, португальском.

Stanford's Core NLP Suite предоставляет набор инструментов анализа естественного языка. Система поддерживает английский, китайский, французский, немецкий и испанский языки и включает в себя инструменты для разметки текста (разбиение текста на слова), определение базовой формы слова, части речи, извлечение именных сущностей, ключевых слов и т.д. Stanford CoreNLP предназначен для того, чтобы очень легко применить большое число инструментов лингвистического анализа к фрагменту текста, написав несколько строк кода, CoreNLP является достаточно гибким и расширяемым. Stanford CoreNLP объединяет многие инструменты Stanford’s NLP, включая частеречную разметку, распознавание именованных сущностей, синтаксический анализатор, определение эмоциональной окраски фрагмента текста и т.д.

Natural Language Toolkit - пакет библиотек и программ, предназначенный для анализа естественного языка в приложениях, разработанных на языке Python. Он предоставляет возможность выполнять следующие операции над исходным текстом: классификации, токенизации, стемминг, тэгирование и т.д. Существует подробная документация по данному пакету, в том числе объясняющая основные концепции, встречающиеся в задачах обработки естественного языка, которые можно решить с помощью данного пакета.

Apache OpenNLP - интегрированный пакет инструментов, предназначенных для обработки текста на естественном языке и работающих на основе машинного обучения. Пакет работает на платформе Java и поддерживает наиболее распространенные задачи обработки естественного языка, такие как токенизация, сегментирование предложений, частеречная разметка, извлечение именованных сущностей и т.д. Эти задачи часто встречаются при реализации систем обработки текста. Работать с данным пакетом можно посредством прикладного программного интерфейса или через командную строку. Apache OpenNLP можно использовать на условиях лицензии Apache License. Исходный код данного пакета присутствует на официальном сайте проекта.

В качестве сервиса/инструмента для извлечения ключевой информации из текста был выбран IBM's Watson Natural Language Understanding Service. При выборе учитывались следующие параметры:

- Простота интеграции и отсутствие необходимости в поднятия собственного сервера;
- В качестве входных данных можно передавать, как обычный текст, так и url-адрес веб сайта, в последнем случае сервис на этапе предварительной обработки очистит веб страницу от рекламы и другого нежелательного контента;
- Поддерживает языки: английский, русский, французский, немецкий, итальянский, португальский, испанский;
- Наличие бесплатной версии API.

2.3 Разрешение лексической многозначности слов при переводе

Под неоднозначностью/многозначностью языкового выражения понимают наличие у него одновременно нескольких различных смыслов. Многозначность подразделяется на следующие типы: лексическую, синтаксическую и речевую, однако в рамках данной работы мы будем рассматривать разрешение именно лексической(WSD). Например, слово “ключ” может употребляться в одном из следующих значений: ключ как инструмент для открывания и ключ как источник воды.

Процесс разрешения требует нескольких вещей: системы словарных знаний для определения множества значений слов и корпус текстов для разрешения. Знания являются одними из ключевых моментов разрешения многозначности: они предоставляют данные, на которые опирается сам процесс разрешения. Эти данные могут быть как корпусы текстов, так и словари, тезаурусы, глоссарии, онтологии и т. д. Для определения качества разрешения

многозначности обычно используются два параметра: точность и полнота разрешения.

Среди основных методов разрешения лексической многозначности выделяют: методы, использующие внешние источники информации, и методы, базирующиеся на машинном обучении, работающие на размеченных корпусах текстов. Также применяются комбинации этих методов. По другой классификации, методы разрешения лексической многозначности различают по типу используемых внешних источников информации: структурированные источники данных (машиночитаемые словари, тезаурусы, онтологии), неструктурированные источники данных в виде корпусов.

Далее будут представлены примеры методов и алгоритмов разрешения лексической многозначности, разбитые на группы:

- методы, основанные на использовании тезаурусов, словарей;
- методы, использующие нейронные сети;
- бустинг;
- лексические цепочки – построение последовательности семантически связанных слов;
- метод ансамбля байесовских классификаторов и сочетаемостные ограничения на основе байесовских сетей;
- контекстная кластеризация – кластеризация контекстных векторов, где разные кластеры соответствуют разным значениям слова.

2.3.1 Методы, основанные на использовании тезаурусных знаний

Одним из способов использования тезаурусных знаний является расчёт семантической близости между контекстом вхождения многозначного слова и всеми синсетами, каждый из которых соответствует одному из значений. Данный способ можно реализовать на основе сравнения близости путей между синсетами слов контекста и синсетами слова, значение которого для данного контекста хотим определить.

В качестве примера одного из данных методов рассмотрим метод Леска, который основан на поиске значения слова в списке словарных определений с учетом контекста, в котором используется данное слово. Основным критерием при выборе значения является следующее правило: заложенный в этом определении смысл должен был частично совпадать со смыслом значений соседних слов в контексте.

Метод леска можно разбить на следующие шаги:

1. Для исходного слова выделяется контекст, размер которого не более 10 ближайших по расположению слов;

2. Для исходного слова осуществляется поиск всех определений в словаре;
3. Сопоставление слов из контекста с каждым найденным определением. В случае если какое-либо из контекста слово присутствует в определении, то этому определению дается балл;
4. Наиболее вероятным значением является то, определение которого набрало наибольшее количество баллов.

При определении значения слова актуального данному контексту в конструкции более длинной, чем несколько слов, так же можно использовать упрощенный алгоритм Леска. В котором пересечение осуществляется между описаниями значений слов и контекстами данных слов в тексте. Кроме толкований словаря для улучшения точности можно дополнительно использоваться размеченные корпуса, а также примеры использования различных значений данного слова.

В качестве одного из примеров разрешения многозначности для английского языка на основе тезауруса WordNet с использованием метода Леска можно привести следующий пример: в словосочетании “pine cones” нужно определить значения для каждого из слов. Для каждого из слов имеем следующие таблицы (см. таблицу 2.1 – 2.2), в столбцах которых указаны соответственно: слово, номер значения, количество пересечений со значениями других слов из контекста, часть определения.

Слово	№	Пересечений	Определение
pine	1	3	kinds of evergreen tree with needle-shaped leaves
cone	2	0	waste away through sorrow or illness

Таблица 2.1 — Таблица значений “pine”

Слово	№	Пересечений	Определение
cone	1	0	solid body which narrows to a point
cone	2	1	something of this shape whether solid or hollow
cone	3	2	fruit of certain evergreen trees

Таблица 2.2 — Таблица значений “cone”

Максимальное пересечение достигаются между первым определением слова “pine” и третьем определением слова “cone”, следовательно, эти значения являются наиболее подходящими согласно методу Леска.

Недостатком алгоритма Леска является, то что при разрешении многозначности очередного слова не учитываются уже найденные значения

других слов из контекста, таким образом алгоритм выполняется для каждого слова отдельно.

2.3.2 Методы на основе нейронных сетей, построенных по данным машиночитаемых словарей

В типичной нейронной сети на вход подается слово, значение которого требуется установить, т. е. целевое слово, а также контекст, его содержащий. Узлы выхода соответствуют различным значениям слова. В процессе обучения, когда значение тренировочного целевого слова известно, веса связующих узлы соединений настраиваются таким образом, чтобы по окончании обучения выходной узел, соответствующий истинному значению целевого слова, имел наибольшую активность. Веса соединений могут быть положительными или отрицательными и настраиваются посредством рекуррентных алгоритмов (алгоритм обратного распространения ошибки, рекуррентный метод наименьших квадратов и т. д.). Сеть может содержать скрытые слои, состоящие из узлов, соединенных как прямыми, так и обратными связями.

Целевое слово представлено узлом, соединенным активирующими связями со смысловыми узлами, представляющими все возможные значения слова, имеющиеся в словарных статьях. Каждый смысловой узел, в свою очередь, соединен активирующими связями с узлами, представляющими слова в словарной статье, соответствующей толкованию данного значения. Процесс соединения повторяется многократно, создавая сверхбольшую сеть взаимосвязанных узлов. В идеале сеть может содержать весь словарь.

При запуске сети первыми активируются узлы входного слова (согласно принятой кодировке). Затем каждый входной узел посылает активирующий сигнал своим смысловым узлам, с которыми он соединен. В результате сигналы распространяются по всей сети в течение определенного числа циклов. В каждом цикле узлы слова и его значений получают обратные сигналы от узлов, соединённых с ними. Узлы конкурирующих значений посылают взаимно подавляющие сигналы. Взаимодействие сигналов обратной связи и подавления, в соответствии со стратегией “победитель получает все”, позволяет увеличить активацию узлов-слов и соответствующих им правильных узлов-значений, одновременно уменьшая активацию узлов, соответствующих неправильным значениям. После нескольких десятков циклов сеть стабилизируется в состоянии, в котором активированы только узлы-значения с наиболее активированными связями с узлами-словами. При обучении сети используется метод обратного распространения (back propagation).

2.3.3 Бустинг

Бустинг – это общий и доказуемо эффективный метод получения очень точного правила предсказания путем комбинирования грубых и умеренно неточных эмпирических правил.

Рассмотрим бустинг на примере алгоритма AdaBoost. AdaBoost является адаптивным алгоритмом, поскольку он может адаптироваться к уровням ошибок отдельных слабых гипотез. На вход алгоритма поступает обучающая выборка, где каждый элемент x_i принадлежит некоторому домену или признаковому пространству X и каждая метка y_i принадлежит некоторому набору меток Y. Для каждого обучающего примера i вес распределения для целых t обозначается $D_t(i)$, где t – это шаг алгоритма. За начальное распределение весов принимается $D_1(i) = \frac{1}{m}$. Пусть метки принимают значения из множества $Y = \{-1, 1\}$. Далее на каждом шаге t, где $t = 1 \ldots T$, выполняется обучение с использованием текущего распределения D_t, после чего строится слабая гипотеза:

$$h_t: X \to \{-1; 1\}, \varepsilon_t = \sum_{i:\, h_t(x_i) \neq y_i} D_t(i) \tag{2.1}$$

где ε_t - ошибка первого рода, по которой выбирается уровень значимости:

$$\alpha_t = \frac{1}{2} ln\left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right) \tag{2.2}$$

и строится новое распределение для следующего шага:

$$D_{t+1} = \frac{D_t(i)}{Z_t} * \begin{cases} e^{-\alpha t}, & \text{если } h_t(x_i) = y_i \\ e^{\alpha t}, & \text{если } h_t(x_i) \neq y_i \end{cases} \tag{2.3}$$

Конечная гипотеза $H(x)$ – это среднее из большинства решений T слабых гипотез, где α_t – вес, присвоенный гипотезе h_t:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{2.4}$$

Идея алгоритма заключается в определении набора весов для обучающей выборки. Первоначально все веса примеров устанавливаются равными, но в каждом цикле веса неправильно классифицированных по гипотезе h_t примеров увеличиваются. Таким образом получаются веса, которые относятся к сложным примерам. Основное свойство AdaBoost – это способность алгоритма уменьшать ошибку обучения. AdaBoost также относительно быстро и просто запрограммировать. Он не имеет никаких параметров для настройки, за

исключением количества циклов и не требует никаких предварительных знаний о слабом обучаемом и поэтому может быть скомбинирован с любым методом для нахождения слабых гипотез. Недостатки метода заключаются в следующем: фактическая производительность бустинга на конкретной задаче явно зависит от данных и слабо обучаемого алгоритма. Теоретически бустинг может выполниться плохо, если данных недостаточно, слабые гипотезы слишком сложные или, наоборот, слишком слабые. Также бустинг особенно восприимчив к шуму.

2.3.4 Использование лексических цепочек для разрешения многозначности

Рассмотрим пример разрешения многозначности с использованием лексических цепочек на основе тезауруса WordNet. Метод построения лексических цепочек включает шаги:

1. Выбирается набор слов-кандидатов на включение в цепочки (существительные и составные существительные).
2. По словарю строится список всех значений для каждого слова-кандидата.
3. Для каждого значения каждого слова-кандидата находится расстояние до каждого слова во всех уже построенных цепочках (слово в цепочке имеет строго определенное значение, задаваемое другими словами в той же цепочке). Между двумя словами есть отношение, если мало расстояние между этими словами в тексте или между значениями этих слов существует путь в тезаурусе WordNet. Выделяют три вида отношений:
    - Extra-strong отношение существует для слов, повторяющихся в тексте. Повтор может быть на любом расстоянии от первого употребления слова.
    - Strong отношение определено между словами, связанными отношением в WordNet. Два таких слова должны находиться в окне не более семи предложений.
    - Medium-strong отношение указывается для слов, синсеты которых находятся на расстоянии больше одного в WordNet (но есть еще и дополнительные ограничения на путь между синсетами). Слова в тексте должны находиться в пределах трех предложений.
4. Слово-кандидат добавляется в цепочки, со словами которых найдена связь. Смысловая неоднозначность устраняется, в цепочку добавляется не просто слово, а его конкретное значение (благодаря выбору значения в словаре на шаге 2).

Для выбора приоритетной цепочки (для вставки слова-кандидата) отношения упорядочены так: extra-strong, strong, medium-strong. Цепочки можно выбирать жадным алгоритмом, при этом слово-кандидат попадает ровно в одну цепочку и после этого выбор уже не может быть изменен, даже если последующий текст покажет ошибочность первоначального решения. Так же приоритетную цепочку можно выбирать по схеме, требующей рассмотрения всех возможных цепочек. Таким образом, будут сформированы цепочки с учетом всех возможных значений слов с последующим выбором наилучшей цепочки.

2.3.5 Разрешение лексической многозначности методом ансамбля байесовских классификаторов

Наивный байесовский классификатор – это простой вероятностный классификатор на основе применения теоремы Байеса. Для различения значений учитывается совместная встречаемость слов в окне заданного размера в текстах корпуса. При разрешении лексической многозначности, представленном в виде задачи обучения с учителем, применяют статистические методы и методы машинного обучения к размеченному корпусу. В таких методах словам корпуса, для которых указано значение, соответствует набор языковых свойств.

Подход основан на объединении ряда простых классификаторов в ансамбль, который разрешает многозначность с помощью голосования простым большинством голосов. В проблеме разрешения лексической многозначности существует понятие контекста, в котором встречается многозначное слово. Этот контекст представляется в виде функции переменных ($F_1, F_2, \dots, F_n$), а значение многозначного слова представлено в виде классификационной переменной S. Все переменные бинарные. Переменная, соответствующая слову из контекста, принимает значение “ИСТИНА”, если это слово находится на расстоянии определенного количества слов слева или справа от целевого слова. Совместная вероятность наблюдения определенной комбинации переменных контекста с конкретным значением слова выражается следующим образом:

$$p(F_1, F_2, .., F_n, S) = p(S) \prod_{i=1}^{n} p(F_i \mid S) \qquad (2.5)$$

Для оценки параметров достаточно знать частоты событий, описываемых взаимозависимыми переменными ($F_i,\ S$). Эти значения соответствуют числу предложений, где слово, представляемое F_i, встречается в некотором контексте многозначного слова, упомянутого в значении S. Если возникают нулевые значения параметров, то они сглаживаются путем присвоения им по умолчанию

очень маленького значения. После оценки всех параметров модель считается обученной и может быть использована в качестве классификатора.

Контекст представлен в виде bag-of-words (модель “мешка слов”). В этой модели выполняется следующая предобработка текста: удаляются знаки препинания, все слова переводятся в нижний регистр, все слова приводятся к их начальной форме (лемматизация). Контексты делятся на два окна: левое и правое. В первое попадают слова, встречающиеся слева от неоднозначного слова, и, соответственно, во второе – встречающиеся справа. Окна контекстов могут принимать 9 различных размеров: 0, 1, 2, 3, 4, 5, 10, 25 и 50 слов. Первым шагом в ансамблевом подходе является обучение отдельных наивных байесовских классификаторов для каждого из 81 возможных сочетаний левого и правого размеров окон. Наивный байесовский классификатор (l, r) включает в себя l слов слева от неоднозначного слова и r слов справа. Исключением является классификатор (0, 0), который не включает в себя слов ни слева, ни справа. В случае нулевого контекста классификатору присваивается априорная вероятность многозначного слова (равная вероятности встретить наиболее употребляемое значение). Следующий шаг при построении ансамбля – это выбор классификаторов, которые станут членами ансамбля. 81 классификатор группируется в три общие категории, по размеру окна контекста. Используются три таких диапазона: узкий (окна шириной в 0, 1 и 2 слова), средний (3, 4, 5 слов), широкий (10, 25, 50 слов). Всего есть 9 возможных комбинаций, поскольку левое и правое окна отделены друг от друга. Например, наивный байесовский классификатор (1, 3) относится к диапазону категории (узкий, средний), поскольку он основан на окне из одного слова слева и окне из трех слов справа. Наиболее точный классификатор в каждой из 9 категорий диапазонов выбирается для включения в ансамбль. Затем каждый из 9 членов классификаторов голосует за наиболее вероятное значение слова с учетом контекста. После этого ансамбль разрешает многозначность путем присвоения целевому слову значения, получившего наибольшее число голосов.

Для разрешения многозначности можно так же воспользоваться построением сочетаемостных ограничений на основе байесовских сетей. Сочетаемостные ограничения – это закономерности использования глагола относительно семантического класса его параметров (субъект, объект (прямое дополнение) и косвенное дополнение. Модели автоматического построения сочетаемостных ограничений важны сами по себе и имеют приложения в обработке естественного языка. Сочетаемостные ограничения глагола могут применяться для получения возможных значений неизвестного параметра при известных глаголах. При построении предложения сочетаемостные ограничения позволяют отранжировать варианты и выбрать лучший среди них. Исследование

сочетаемостных ограничений могло бы помочь в понимании структуры ментального лексикона. Системы обучения сочетаемостных ограничений без учителя обычно комбинируют статистические подходы и подходы, основанные на знаниях. Компонент базы знаний – это обычно база данных, в которой слова сгруппированы в классы.

Статистический компонент состоит из пар предикат-аргумент, извлеченных из неразмеченного корпуса. В тривиальном алгоритме можно было бы получить список слов (прямых дополнений глагола), и для тех слов, которые есть в WordNet, вывести их семантические классы. Семантическим классом называется синсет тезауруса WordNet, т.е. класс соответствует одному из значений слова. Таким образом, в тривиальном алгоритме на основе данных WordNet можно выбрать классы (значения слов), с которыми употребляются (встречаются в корпусе) глаголы.

Байесовские сети, или байесовские сети доверия (БСД), состоят из множества переменных (вершин) и множества ориентированных ребер, соединяющих эти переменные.

Такой сети соответствует ориентированный ациклический граф. Каждая переменная может принимать одно из конечного числа взаимоисключающих состояний. Пусть все переменные будут бинарного типа, т. е. принимают одно из двух значений: истина или ложь. Любой переменной A с родителями $B1$, ..., Bn соответствует таблица условных вероятностей. Иерархия существительных в WordNet представлена в виде ориентированного ациклического графа. Синсет узла принимает значение “истина”, если глагол “выбирает” существительное из набора синонимов. Априорные вероятности задаются на основе двух предположений: во-первых, маловероятно, что глагол будет употребляться только со словами какого-то конкретного синсета, и вовторых, если глагол действительно употребляется только со словами из данного синсета(например, синсет ЕДА), тогда должно быть правомерным употребление этого глагола с гипонимами этого синсета (например, ФРУКТ).

Те же предположения, что для синсетов, верны и для употреблений слов с глаголами:

1. слово, вероятно, является аргументом глагола в том случае, если глагол употребляется с каким-либо из значений этого слова;
2. отсутствие связки глагол-синсет говорит о малой вероятности того, что слова этого синсета употребляются с глаголом.

Словам “вероятно” и “маловероятно” должны быть приписаны такие числа, сумма которых равна единице.

2.3.6 Контекстная кластеризация

Каждому вхождению анализируемого слова в корпус соответствует контекстный вектор. Выполняется кластеризация векторов, где разные кластеры соответствуют разным значениям слова. Алгоритмы кластеризации полагаются на дистрибутивную гипотезу, в соответствии с которой слова, употребляемые в схожих контекстах, считаются близкими по смыслу.

При решении задачи различения значений используются контекстные вектора: если целевое слово встречается в тестовых данных, то контекст этого слова представляется в виде вектора контекста. Вектор контекста - это средний вектор по векторам свойств каждого из слов контекста. Вектор свойств содержит информацию о совместной встречаемости данного слова с другими словами, этот вектор строится по данным корпуса текстов на этапе обучения.

Первоначально строится матрица совместной встречаемости слов по данным обучающего корпуса. Вектор свойств (строка матрицы) содержит информацию о совместной встречаемости данного слова с другими. После создания матрицы выполняется разделение тестовых данных, т. е. группировка примеров употреблений (фраз) с целевым словом. Каждому слову в примере употребления в тестовых данных соответствует вектор свойств из матрицы встречаемости. Таким образом, набор тестовых данных, включающих употребление исследуемого слова, преобразуется в набор контекстных векторов, каждый из которых соответствует одному из употреблений целевого слова.

Различение значений происходит путем кластеризации контекстных векторов с помощью разделяющего или иерархического “сверху вниз” алгоритма кластеризации. Получающиеся кластеры составлены из употреблений близких по значению фраз, и каждый кластер соответствует отдельному значению целевого слова. Векторы свойств, полученные по небольшому корпусу текстов, имеют очень малую размерность (несколько сотен), что не позволяет полностью описать закономерности совместной встречаемости слов. Для решения этой проблемы векторы свойств слов расширяются содержательными словами, извлеченными из словарных толкований разных значений данного слова.

Метод кластеризации может быть полезен при различении значений слов без учителя при небольшом количестве обучающих данных.

В данной работе для разрешения лексической многозначности при переводе будет использован метод, основанный на использовании тезаурусных знаний, который позволит избежать самостоятельного обучения системы на большом корпусе размеченных документов. Основные этапы применения метода следующие: для исходного ключевого слова будет выделяться контекст, для всех слов контекста из многоязычного тезауруса будут извлекаться соответствующие

им толкования, синонимы, а также примеры их употребления. Затем будет осуществляться пересечение между описанием значений и соответствующими синонимами ключевого слова, а также контекстом его употребления и информацией, извлечённой из тезауруса. Значение ключевого слова, которое будет иметь наибольшее количество пересечений и будет искомым. В качестве многоязычного тезауруса воспользуемся BabelNet.

BabelNet - это многоязычный энциклопедический словарь материалы которого доступны на 284 языке с лексикографическим и энциклопедическим охватом терминов, а также семантическая сеть, которая связывает понятия и именованные сущности большой сетью семантических отношений, состоящей из около 15 миллионов синсетов. Каждый синсет представляет собой определенное значение и содержит все синонимы, которые выражают это значение на разных языках.

На данынй момент BabelNet получается из автоматической интеграции:

- WordNet (версия 3.0);
- Open Multilingual WordNet (январь 2017);
- OmegaWiki - большой многоязычный словарь (январь 2017);
- Wikipedia - крупнейшая многоязычная веб-энциклопедия (январь 2017);
- Wiktionary (февраль 2017);
- Wikidata (январь 2017);
- Wikiquote - многоязычный сборник цитат и творческих работ (март 2015);
- VerbNet (версия 3.2);
- Microsoft Terminology (июль 2015);
- GeoNames - свободная географическая база данных, содержащая более восьми миллионов названий городов (апрель 2015);
- WoNeF - французский перевод WordNet (февраль 2017);
- ItalWordNet - лексико-семантическая база данных для итальянского языка (февраль 2017);
- ImageNet - база данных изображений, организованная в соответствии с иерархией WordNet (2011);
- FrameNet (версия 1.6);
- WN-Map - сопоставления между версиями WordNet (2007);
- Korean WordNet (январь 2017);
- GAWN WordNet - база данных, состоящая из ирландских слов и семантических отношений между ними (январе 2017).

Использовать данный тезаурус можно бесплатно под лицензией CC BY-NC-ND 4.0. Ограничение на число запросов - 50000 в сутки.

2.4 Выводы

В ходе анализа существующих сервисов/инструментов для реализации CLIR пришли к следующим заключениям:

1. Существует множество открытых API для извлечения ключевой информации из документа. В данной работе будет использоваться IBM's Watson Natural Language Understanding Service, благодаря относительно высокой точности извлечения, поддержке английского, французского, немецкого, итальянского, португальского, русского и испанского языков, а также возможности приёма в качестве входных данных url веб-страницы, что особенно актуально для мобильных клиентов.
2. Среди основных методов разрешения лексической многозначности выделяют: методы, использующие внешние источники информации, такие как тезаурусы, и методы, базирующиеся на машинном обучении. В данной работе для разрешения многозначности при переводе ключевых слов будет использоваться многоязычный энциклопедический словарь BabelNet, включающий в себя данные из WordNet, Wikipedia, а также других семантических ресурсов и покрывающий 271 язык.

ГЛАВА 3
МОБИЛЬНЫЙ КЛИЕНТ ДЛЯ ПОИСКА РЕЛЕВАНТНЫХ ДОКУМЕНТОВ

Учитывая все выше перечисленное в дипломной работе, была поставлена задача по разработке мобильного приложения под операционную систему Android, которое будет простым в использовании, обладать удобным и приятным интерфейсом, минималистическим, поддерживать весь спектр существующих устройств с версией API 14+ (Android 4.0), тем самым, согласно официальной статистике от google, будет поддерживать 99.1% активных девайсов. Интерфейс для работы с приложение будет реализован в соответствии с концепцией material design.

При реализации данной системы для извлечения ключевой информации из поступающего на вход текста/url-адреса веб сайта воспользуемся IBM’s Watson Natural Language Understanding Service, для разрешения лексической многозначности используем многоязычный многоязычный энциклопедический словарь BabelNet.

Разработанное приложение будет поддерживать следующие языки для исходного текста: английский, французский, немецкий, итальянский, португальский, русский, испанский. Язык же обнаруженных релевантных документов может быть один из 271 предоставленных здесь http://babelnet.org/stats#LanguagesandCoverage.

3.1 Разработка архитектуры системы

При разработке приложения был использован объектно-ориентированный подход. Архитектура приложения будет построена по принципу Clean Architecture.

Clean Achitecture — принцип разработки приложений, предложенный Uncle Bob'ом. Код, спроектированный с учётом этой архитектуры, легче тестировать и переиспользовать.

Преимуществами данной архитектуры являются:

- Простота написания тестов;
- Независимость от фреймворков;
- Независимость от UI;
- Независимость от Баз Данных;
- Независимость от внешних сервисов, с которыми взаимодействует приложение.

Суть Clean Architecture заключается в разделении логики приложения на несколько составляющих слоёв: слой бизнес-логики, слой представления и слой данных.

При этом чтобы обеспечить максимальную независимость этих слоев, на каждом из них используется своя модель данных, которая конвертируется при взаимодействии между слоями. Так же выделяются отдельные интерфейсы для взаимодействия между слоями.

Схема данных слоев выглядит следующим образом:

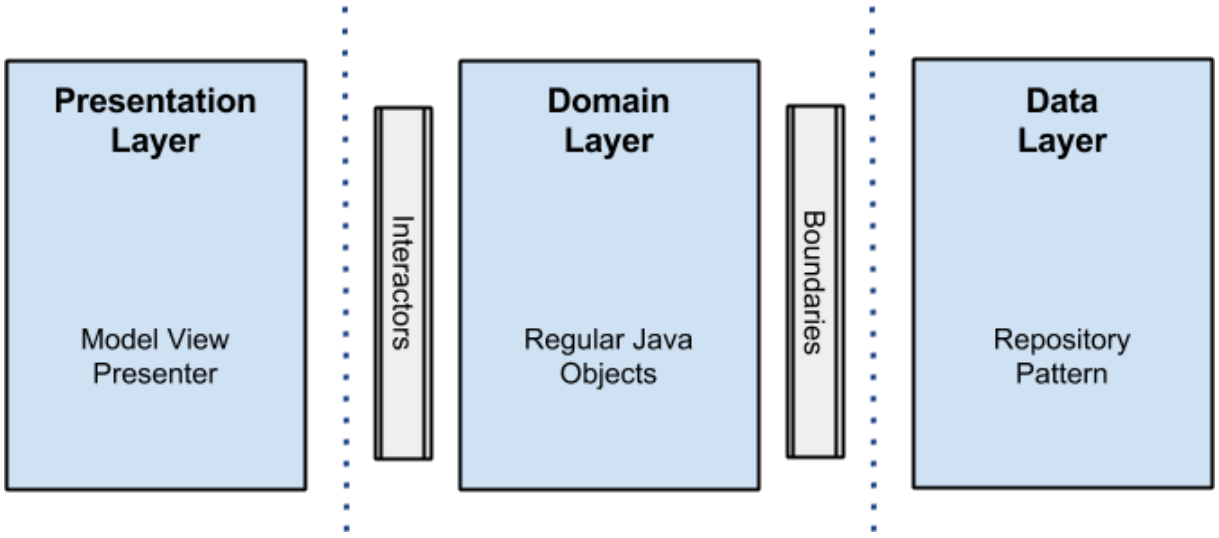

Рисунок 3.1 – схема Clean Architecture

Слой представления предназначен в первую очередь для взаимодействия с пользователем, так же он отвечает за логику отображения данных на экране и за другие процессы, связанные с UI. Этот слой не должен содержать логику приложения, не связанную с UI. Именно слой представления привязывается к экранам и помогает организовать взаимодействие со слоем бизнес-логики и работу с данными. Этот слой может быть реализован с использованием любого предпочитаемого паттерна, к примеру, MVC, MVP, MVVM и других.

При реализации данного приложения слой представления организуем согласно паттерну MVP. Он позволит нам разделить экран на UI-часть (View), на логику работы с UI (Presenter) и объекты для взаимодействия с UI (Model).

В MVP Presenter управляет только одной View и взаимодействует с ней через специальный интерфейс. View управляется только с помощью Presenter и не отслеживает изменения Model. Presenter получает все данные из слоя данных, обрабатывает их в соответствии с требуемой логикой и управляет View.

Слое бизнес-логики содержит всю бизнес-логика приложения. Этот слой является неким объединением слоев сценариев взаимодействия и бизнес-логики.

Именно к этому слою обращается слои представления для выполнения запросов и получения данных. В данном приложении слой бизнес-логики будет реализован в виде Java-модуля, который не содержит никаких зависимостей от Android-классов. Преимуществом данного подхода является то, что для реализации бизнес-логики нам нужны только классы моделей и стандартные средства языка Java. Более того, такой подход позволит легко тестировать этот слой с помощью обычных тестов на JUnit, что очень удобно. В таком случае иногда не будет возможности выполнить какой-либо метод или использовать некоторые классы из других слоев. Поэтому для взаимодействия с этим слоем используются интерфейсы.

Слой данных отвечает в первую очередь за получение данных из различных источников и их кэширование. Он реализуется за счет паттерна Repository, и его общую схему можно представить следующим образом:

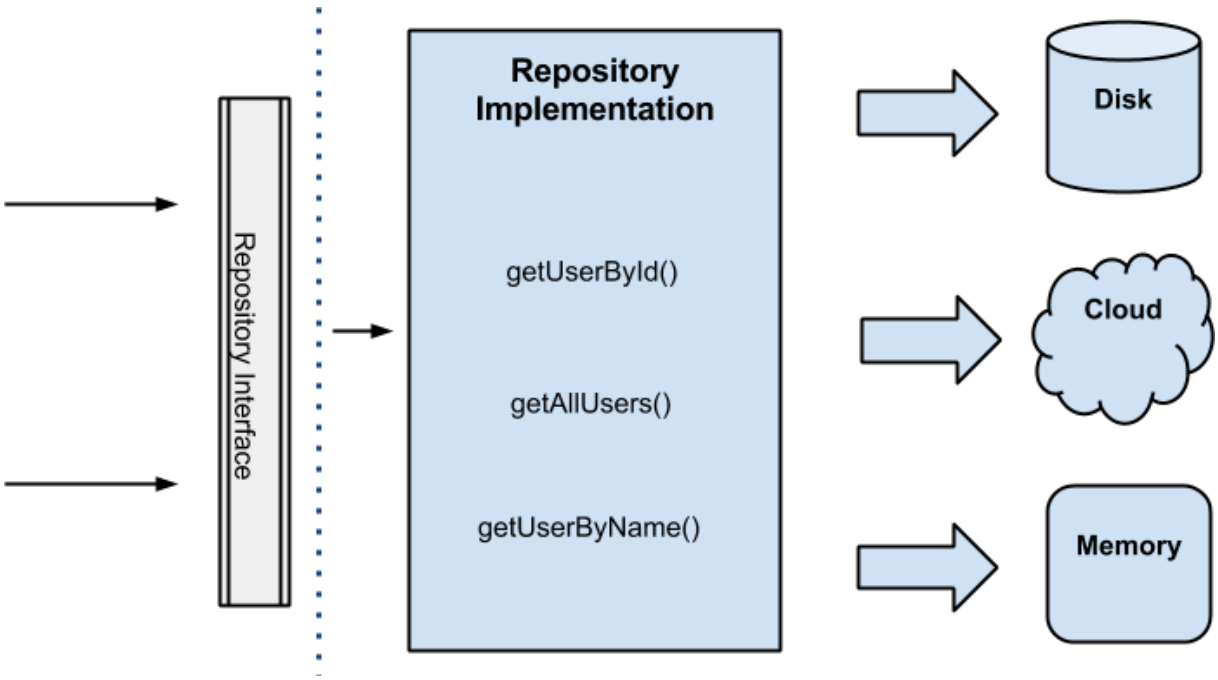

Рисунок 3.2 – Слой данных

Существует несколько плюсов от использования такого подхода. Во-первых, другие слои, которые запрашивают данные, не знают о том, откуда эти данные приходят. Более того, им не нужно этого знать, так как это усложняет логику работы и модуль берет на себя лишнюю ответственность. Во-вторых, слой данных в таком случае выступает единственным источником информации.

3.2 Методика применения разработанного приложения

Разработанная программа обладает интуитивно понятным интерфейсом.

Для поиска в сети интернет документов релевантных данному тексту/url веб-страницы нужно выполнить действия, описанные ниже. Сначала необходимо запустить приложение. После заставки отобразится главный экран (см. Рисунок 3.3).

Рисунок 3.3 – Главный экран приложения

На главном экране в появившемся поле ввода необходимо ввести текст, для которого будет осуществляться поиск релевантной информации в интернете. Так же в качестве исходных данных можно использовать url веб-страницы. Для этого нужно перейти на экран “Поиск по url”, доступный через навигационное меню. Для отображения навигационного меню можно выполнить свайп слево-направо по экрану, либо нажать на кнопку “Меню”.

По нажатию кнопки “Продолжить” появится экран выбора языков, на которых будет извлекаться актуальная информация из сети интернет(см. Рисунок 3.4).

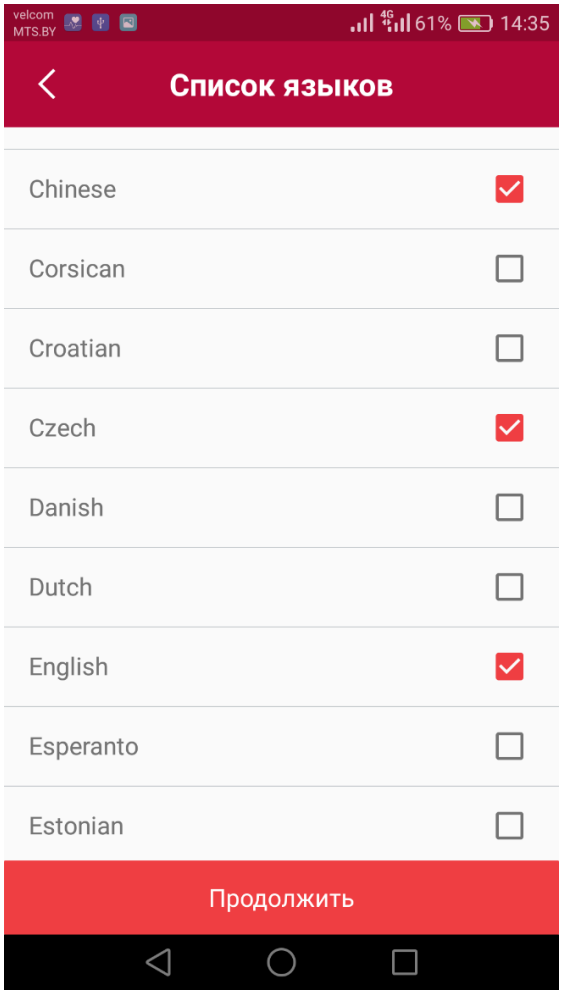

Рисунок 3.4 – Экран выбора актуальных пользователю языков

На данном экране выбора отображены все поддерживаемые приложением для поиска информации языки.

По нажатию кнопки “Продолжить” после обработки входных данных будет выдан список сформированных приложением запросов для поиска релевантных данному документов на выбранных пользователем языках. В дальнейшем список языков можно будет поменять не вводя текст заново.

Экран сформарованных запросов для поиска релевантных данному документов на выбранных языках и экран результатов поиска в поисковой системе Google для запроса на английском языке приведены ниже(см. Рисунок 3.5).

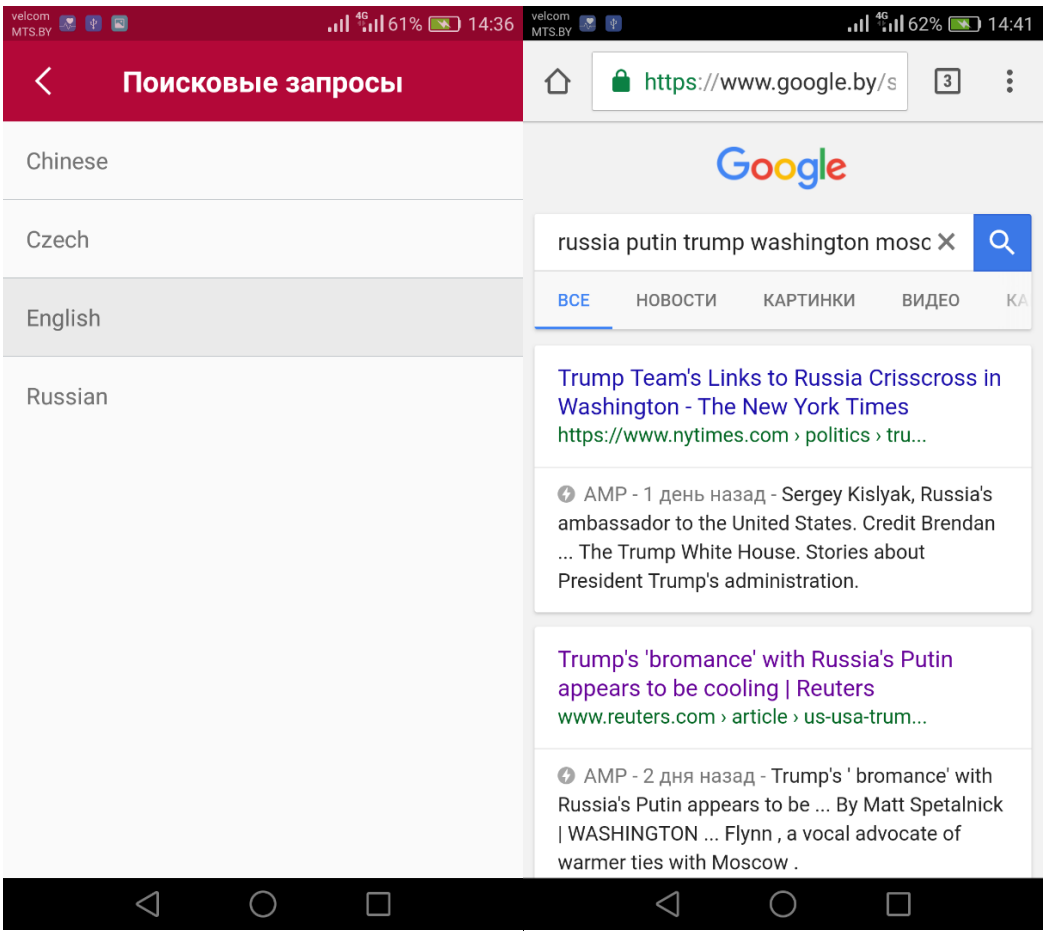

Рисунок 3.5 – Результаты поиска релевантных документов

Так же через навигационное меню(см. Рисунок 3.6) можно просмотреть историю поиска и перейти на экран настроек приложения. На экране настроек можно поменять цветовую гамму и предустановить языки, которые будут выбраны при поиске информации по умолчанию. В истории вместе с входными данными так же хранятся уже сформированные запросы для Google. При необходимости можно очистить историю поиска.

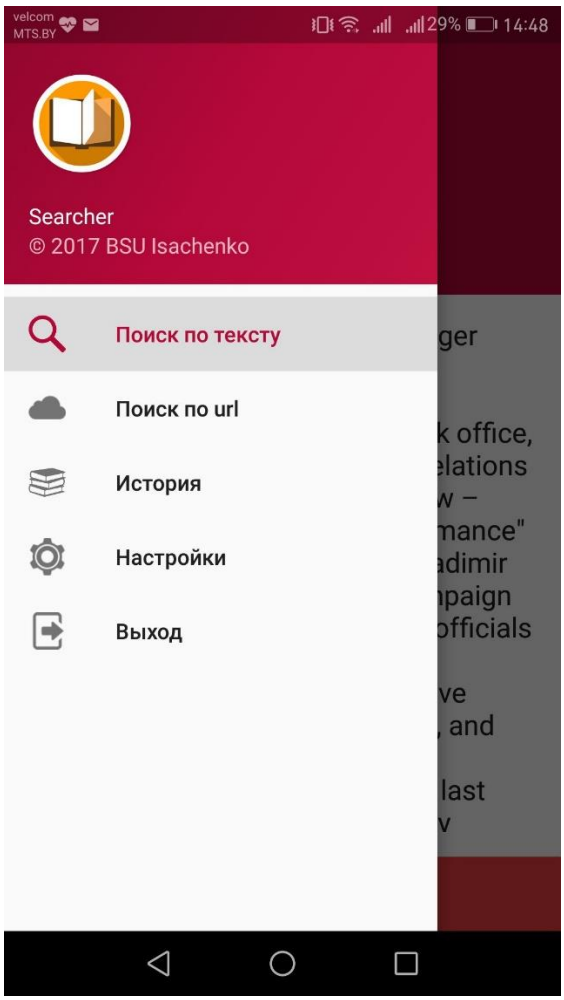

Рисунок 3.6 – Навигационное меню приложения

По клику на пукт “История” откроется экран истории поиска. По клику на экран “Настройки” откроется экран настроек. Для выхода из приложения можно использовать кнопку “back” или пункт “Выйти” в навигационном меню.

ЗАКЛЮЧЕНИЕ

В рамках дипломной работы «*Cross-language функциональность автоматического поиска в сети Internet релевантных документов*» получены следующие результаты.

В первой главе выполнен анализ подходов к реализации многоязычного поиска. Сформулированы цели и задачи для поставленной проблемы. Рассмотрены методы извлечения ключевой информации из текста, основанные на семантических данных о словах и численных характерестиках встречаемости слов в тексте. Показана роль перевода при CLIR, а также рассмотрены способы и преимущества каждого из подходов его реализации.

Во второй главе исследованы все самые известные на текущий момент алгоритмы, которые применяются для разрешения лексической многозначности при переводе слов, описаны их преимущества и недостатки. Данные алгоритмы можно подразделить на 3 класса: алгоритмы, использующие внешние источники информации, алгоритмы, базирующиеся на машинном обучении, работающие на размеченных корпусах текстов, а также алгоритмы предоставляющие собой комбинацию 1-ых и 2-ых. Так же выполнен анализ сервисов, предоставляющих возможность извлечения ключевой информации из текста.

В третьей главе описана разработка мобильного клиента на языке *Java* для ОС Android, предоставляющего пользователю возможность для входного документа/url веб-страницы сформировать поисковые запросы для поисковой системы Google, на актуальных для пользователя языках. Также приведена методика применения разработанного приложения.

CLIR является очень актальной задачей, однако точность многоязычного поиска на данный момент невысока, одной из причин является сложность разрешения лексической многозначности слов при переводе. Возможно уже в ближайшем будущем, данный тип поиска позволит устранить лингвистическое несоответствие между предоставляемыми запросами и документами, которые извлекаются из информационной сети, тем самым удалив языковой барьер.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Manning, C. D. Introduction to Information Retrieval. / C. D. Manning, P. Raghavan, H. Schütze. - Cambridge University Press, 2008. – 581 с.
2. Автоматическая обработка текстов на естественном языке и компьютерная лингвистика: учеб. пособие /Е.И. Большакова, Э.С. Клышински, Д.В. Ландэ [и др.]. - М.: МИЭМ, 2011. - 272 с.
3. Matsuo, Y. Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information. / Y. Matsuo – Tokyo, 2003. – 13 с.
4. Turney, P.D. Learning algorithms for keyphrase extraction. Information Retrieval / P.D. Turney. - Ottawa, Ontario, Canada, 2000. – 477 с.
5. Porter, M.F. An algorithm for suffix stripping. / M.F. Porter – Cambridge, 1997. – 6 с.
6. ANDERKA, M., LIPKA, N., AND STEIN, B. 2009. Evaluating cross-language explicit semantic analysis and cross-querying. In Proceedings of the 10th Cross-language Evaluation Forum Conference on Multilingual Information Access Evaluation: Text Retrieval Experiments (CLEF’09). Springer, 50–57 с.
7. Kraiij, W., Nie, J-Y., Simard, M.: Emebdding Web-based Statistical Translation Models in Cross-Language Information Retrieval. Computational Linguistics (2003) – 39 с.
8. The Cross-Language Evaluation Forum (CLEF). http://clef-campaign.org
9. Virga, P., Khudanpur, S.: Transliteration of proper names in cross-lingual information retrieval. In: ACL Workshop on Multilingual and Mixed Language Named Entity Recogni-tion (2003) – 8 с.

Обновить