

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

<i>Факультет</i>	<i>Кафедра</i>
вычислительной техники	МОиПЭВМ
Направление	09.04.04 «Программная инженерия»
подготовки	
Магистерская	Программное обеспечение вычислительной
программа	техники и автоматизированных систем

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему

**Модель информационного пространства на
основе концепции семантического веба**

Студент	_____	Кузнецова Анна Олеговна
	(подпись, дата)	(ФИО полностью)
Руководитель	_____	Афонин А. Ю.
	(подпись, дата)	(фамилия, инициалы)
Нормоконтролёр	_____	Такташкин Д.В.
	(подпись, дата)	(фамилия, инициалы)
Рецензент	_____	Васин Л. А.
(должность, место работы)	(подпись, дата)	(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____ Макарычев П.П.

(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ Попова Н.А.

(подпись) (фамилия, инициалы)

Пенза, 2020

Содержание

Введение	3
1 Принципы организации знаний.....	7
1.1 Понятие искусственного интеллекта. Направления и подходы искусственного интеллекта	7
1.2 Интеллектуальный анализ данных.....	12
1.3 Технологические подходы к обработке знаний	14
1.4 Концепция Semantic Web, уровни представления Semantic Web.....	16
2 Математические модели формализации знаний	23
2.1 Методы представления и извлечений знаний	23
2.2 Методы представления и хранения семантической информации	29
2.3 Веб-технологии поиска и обработки семантической информации.....	39
3 Разработка программной реализации	51
3.1 Выбор языка и технологии программирования	51
3.2 Описание основных компонентов приложения	55
3.3 Порядок запуска и работы приложения.....	57
Заключение	63
Список использованных источников	64
Приложение А	67

Введение

Развитие технологии World Wide Web является одним из ключевых направлений компьютерных и сетевых технологий. Можно говорить о существенно определяющем характере влияния технологии Web на состояние информационного общества.

Развитие информационной инфраструктуры связано с ростом масштаба компьютерных сетей и объема передаваемых в них данных. В этих условиях механическое увеличение мощности инфраструктуры сообразно решаемым задачам может дать эффект не во всех случаях. В этой связи на первый план выходят вопросы, связанные как с поиском новых типов сетевых решений, так и с возможностью изменения качественной структуры распространяемого в них контента. Так, введение интеллектуальных механизмов распространения, хранения и обмена данными во многом повысит эффективность применения уже существующих решений.

Так, среди них можно выделить нейросетевые алгоритмы маршрутизации.

Целью работы является анализ существующих методов формализации информации, применяемых в условиях сети Интернет. Для ее реализации необходимо решить следующие задачи:

- получить представление о интеллектуальном анализе данных;
- проанализировать методы представления данных и знаний;
- проанализировать методику представления семантической информации и запросов к ней в условиях сети Интернет;
- выполнить программную реализацию семантического анализа информации с возможностью ввода данных и просмотра результатов.

Актуальность темы. Для современного общества характерно увеличение темпов развития технологий и масштабная компьютеризация, и, как следствие, стремительно возрастающая роль информации. Данное явление, ученые описывают термином «информационное общество»,

характеризуя его как новый социальный порядок, для которого характерно преобладание во многих сферах общественной жизни процессов, связанных с производством, распространением и переработкой информации. Неотъемлемой частью информационного общества является развитие телекоммуникационной инфраструктуры, облегчающей обмен информацией, а также создание обширного информационного пространства.

В настоящее время подобным пространством является Интернет, или, как его еще называют, Всемирная сеть. Согласно докладу Организации объединенных наций «The State of Broadband 2017: Broadband Catalyzing Sustainable Development», по состоянию на 2017 г. 52% населения Земли не имеет возможности взаимодействовать с Интернетом. Однако, исходя из приведенных данных следует, что примерно половина населения земли (а это миллиарды пользователей) объединена единым информационным пространством.

В настоящее время, Интернет «проник» во все сферы общественной жизни – политическую, экономическую, духовную, социальную. Интернет оказывает влияние на здравоохранение, образование, искусство. Пользователи сети осуществляют поиск необходимой информации для учебы, работы, досуга, обмениваются новостями, размещают объявления, совершают сделки и многое другое. Другими словами, непрерывно осуществляют производство, распространение и переработку информации.

И одна из самых актуальных проблем при современных объемах информации, содержащейся в Интернете – это проблема поиска. При взаимодействии со всемирной сетью, большинство людей пользуются поисковыми системами, предлагающими пользователю перечень сайтов, содержащих информацию согласно запросу. Однако при современном уровне развития и доступности технологий, количество сайтов (в зависимости от запроса) может измеряться тысячами, сотнями тысяч и миллионами. Именно эту проблему – соответствия полученной информации потребностям

пользователя, призван решить Semantic Web.

Целью данной работы является исследование процессов обработки данных при помощи методов Semantic Web, такие как семантический анализ текста и выделение ключевых слов.

В соответствии с целью исследования решены **задачи**:

- анализ работы искусственного интеллекта, всемирной паутины в целом и Semantic Web в частности;
- исследование механизмов представления данных как их формализация в виде теоретико-графовых структур;
- разработка программной реализации приложения, обеспечивающего семантический анализ сайтов на базе библиотеки PyQt.

Объектом исследования является Semantic Web, увеличение релевантности запросов в интернете.

Предметом исследования являются процессы сбора и обработки во всемирной сети интернет, а также способы «понимания» машиной человеческого текста.

Методы исследования основаны на теории информационных систем, применяется метод семантического анализа текстовой информации, метод математической логики, метод интеллектуального анализа данных.

Научная новизна работы заключается в следующем: в данной работе был разработан программный код на языке Python, который выполняет следующие задачи:

- задание ссылки на подлежащий анализу ресурс;
- поддержка семантического анализа текстовой информации;
- поддержка лемматизации текстовой информации, а также удаление стоп-слов;
- визуализация текстовой информации в виде семантического облака;
- выделение ключевых слов в тексте на базе методов скрытого

семантического анализа;

- вывод результатов.

Практическая значимость исследований. Полученные в данной работе теоретические и практические результаты позволяют проработать теги для своего сайта, что повысит релевантность сайта.

Реализация и внедрение результатов работы. Результаты работы данной программы могут быть использованы при продвижении сайтов в дальнейшем.

Достоверность полученных результатов основана на использовании современных математических методов и технологий анализа данных и компьютерного моделирования.

Результаты, выносимые на защиту:

- различные принципы и подходы организации знаний;
- модели формализации знаний, методы их представления и извлечения, методы хранения и обработки семантической информации;
- разработка программной реализации.

Диссертация состоит из введения, трех глав, заключения, списка литературы из 32 наименований и 1 приложения. Объем работы: 127 страниц основного текста, включающего 24 рисунка, 3 таблицы и 12 страниц приложений.

1 Принципы организации знаний

1.1 Понятие искусственного интеллекта. Направления и подходы искусственного интеллекта

Сам термин «искусственный интеллект» (англ. artificial intelligence) был предложен в 1956 году на семинаре в Стэнфордском университете с таким же названием, который был посвящен решению логических задач.

Искусственный интеллект можно определить как научную дисциплину, в задачи которой входит исследование разумного поведения. Это определение, однако, не конкретизирует понятие разумности. Формализация последнего представляет собой довольно сложную задачу. Она в существенной степени определяется исходными данными и спецификой предметной области. В этой связи можно говорить о моделировании паттернов принятия решений, свойственных человеку.

Искусственный интеллект (ИИ) – это совокупность научных дисциплин, использующих ЭВМ для изучения методов решения задач интеллектуального (творческого) характера.

Искусственный интеллект – одно из направлений информатики, его целью является разработка аппаратно-программных средств, которые могли бы позволить пользователю-непрограммисту определять и решать свои традиционно интеллектуальные задачи и общаться с компьютерами на ограниченном подмножестве естественного языка.

Конструктивность искусственного интеллекта состоит в потенциальной возможности исследования проблем для теорий интеллекта. Модели, заимствованные из этих теорий, можно математически формализовать, после чего оценить их возможности с применением ЭВМ.

Можно выделить следующие направления исследований искусственного интеллекта (рисунок 1).

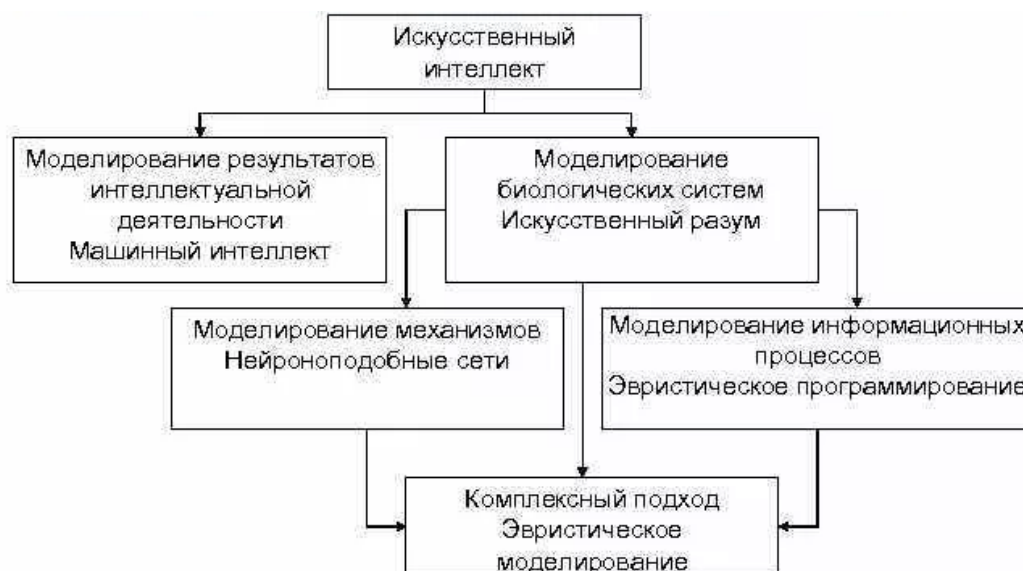


Рисунок 1 – Направления исследования искусственного интеллекта

К сфере, связанных с моделированием результатов интеллектуальной деятельности можно отнести широкий круг задач. Так, к ним можно отнести задачи, связанные с поиском оптимального решения, распознаванием образов, логические задачи и т. п. При этом найденные алгоритмы решения задач должны работать на существующей аппаратной базе ЭВМ. Данное направление можно отнести к программно-прагматическому [1, 2].

Моделирование биологических систем можно рассматривать как бионическое направление. Оно специализируется на исследовании структур и процессов, которые характерны для поведенческой деятельности биологических систем. Так, одним из его составляющих элементов является моделирование нейронных сетей и систем. На его базе наука нейроинформатика, играющая важную роль в развитии нейрокомпьютерных систем [4, 5].

На сегодняшний день традиционным (классическим) принято считать программное прагматическое направление, при котором не ставится вопрос об адекватности используемых структур и методов тем, которыми пользуется в аналогичных случаях человек, рассматривается только конечный результат решения задачи. В рамках этого направления сначала велись поиски модели и алгоритм человеческого мышления. Но ни одна из существующих наук

(философия, психология, лингвистика) не могла обеспечить такой алгоритм. Тогда специалисты в области искусственного интеллекта предложили свой вариант: модель лабиринтного поиска. Этот подход представляет задачу в виде графа, отражающего пространство состояний, и на этом графе ищется оптимальный путь от входных данных к результату.

Эвристика – это правило, которое теоретически необоснованно, но с помощью которого можно уменьшить количество переборов в области поиска; использование методов математической логики. В 1973 году был создан язык Пролог на основе метода резолюций, что позволило автоматически доказывать теоремы при наличии начальных аксиом.

Можно выделить следующие методы, применяемые в моделях искусственного интеллекта [2, 3]:

- символичный подход;
- логический подход;
- агентно-ориентированный подход;
- гибридный подход.

Символьный подход к ИИ представляет собой круг методов интеллектуальной обработки информации, которые допускают ее формализацию в классе символических представлений. Последние, в свою очередь, описывают предметную и семантическую составляющую исходной информации. Это, в свою очередь, позволяет установить между отдельными символами связи логического характера. Таким образом, символические вычисления дают возможность получения результата на основе манипулирования правилами формализованной предметной области. Это, в свою очередь, может приводить к созданию новых правил преобразования данных, основанных на композиции старых.

В качестве примера применения символического подхода можно привести решение задач, связанных с применением методов символической математики. Одним из первых языков, поддерживающих такие возможности, был язык

Lisp.

К недостаткам символьного подхода можно отнести достаточно ограниченные возможности интерпретации результатов его применения. Это, в свою очередь, обусловлено сложностью формализации предметной области.

В основе логического подхода лежит попытка формализации мышления на уровне сознания и рассуждений. В качестве теоретического аппарата в этом случае используются методы математической логики и теории предикатов.

Логический подход может быть в полной мере реализован в языке логического программирования Пролог. Написанные на нем программы можно рассматривать как наборы фактов и предикатов. В последних задаются правила и порядок выполнения логического вывода. При этом конкретный алгоритм вывода не задается.

Достоинствами рассмотренного подхода являются относительная прозрачность процесса логического вывода и легкость его естественно-языковой интерпретации. Также само поведение процесса логического вывода однозначно определяется конфигурацией исходных данных.

К недостаткам логического подхода следует отнести сложность формализации предметной области в условиях неопределенности. Наличие последней может быть связано с нечеткостью и недостаточностью знаний, отсутствием конечной цели, интервальностью или вероятностным характером данных и т.п. Также процесс принятия решения в условиях применения логических систем достаточно сложно распараллеливать.

Агентно-ориентированный подход является одним из наиболее перспективных подходов, применяемых в настоящее время. Он опирается на использовании интеллектуальных (рациональных) агентов. Концепция была предложена Шохемом (англ. Yoav Shoham) в 1990 г.

В основе концепции агентно-ориентированного подхода лежат концепции понятия агента и его поведения, которое может носить ментальный характер. Структура взаимодействия агента и среды приведена на рисунке 2.

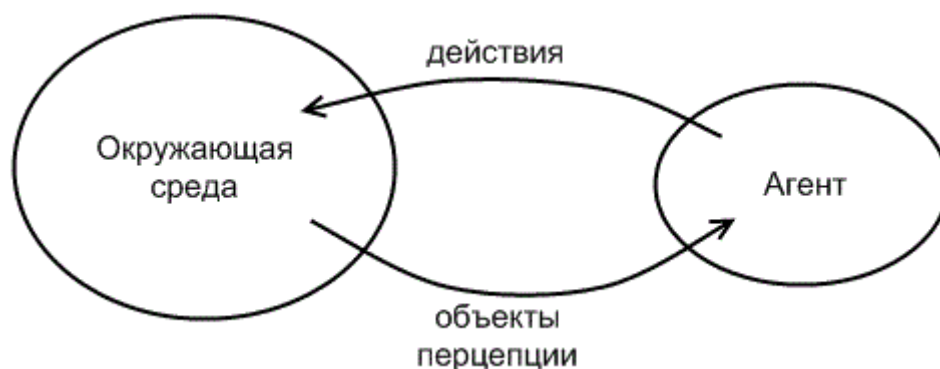


Рисунок 2 – Модель взаимодействия агента с окружающей средой

Динамика агента определяется свойствами и влиянием среды, в которой он находится. Конечным итогом функционирования агента является решение некоторой задачи или достижение цели в условиях среды, в которой он находится.

Интеллект в контексте агента можно рассматривать как элемент планирования его поведенческой стратегии. Связь между агентом и окружающей средой можно реализовать через систему датчиков, считывающих реакцию среды, и с помощью исполнительных механизмов, которые могут влиять на ее состояние.

В рамках агентно-ориентированной методики приоритет отдается изучению методов и алгоритмов, которые обеспечивают агенту минимизировать риски при выполнении его задачи в условиях внешней среды.

Гибридный подход предполагает интеграцию разнородных моделей искусственного интеллекта. В нем утверждается, что объединение возможностей подходов на основе различных парадигм может дать синергетический эффект, которых приведет к максимизации когнитивных и вычислительных возможностей системы.

Гибридный подход можно проиллюстрировать на следующем примере. Например, в некоторой системе правила умозаключений на экспертном уровне могут генерироваться нейронными сетями. Порождающие правила, в свою очередь, можно синтезировать на основе механизма статистического вывода. Таким образом, можно ожидать, что интеллектуальные системы на

основе гибридного подхода могут обладать большими когнитивными возможностями по сравнению с традиционно применяемыми решениями.

В данной главе были рассмотрены основные методы, применяемые в моделях с искусственным интеллектом. Каждый из данных методов активно применяется в самых разнообразных задачах.

1.2 Интеллектуальный анализ данных

Под интеллектуальным анализом данных (ИАД) следует понимать анализ предварительно обработанных данных. Он может носить достаточно многообразный характер. При этом могут использоваться различные математические методы, относящиеся машинному обучению. К ним можно отнести методы, основанные на статистике, генетические алгоритмы, деревья решений и т. д. Другую группу методов образуют нейросетевые модели, различные подходы на базе искусственного интеллекта и классификации данных. Целью применения анализа такого уровня сложности является извлечение новой информации из данных, отражающей глубинные связи между их элементами [7, 8].

Методы ИАД позволяют решить многие задачи, с которыми сталкивается аналитик.

Процесс ИАД в общем случае состоит из трёх стадий [6, 7]:

- 1) выявление закономерностей (свободный поиск);
- 2) использование выявленных закономерностей для предсказания неизвестных значений (прогностическое моделирование);
- 3) анализ исключений, предназначенный для выявления и толкования аномалий в найденных закономерностях.

В некоторых случаях явно выделяется промежуточный уровень, в котором проверяется достоверность закономерностей найденных шаблонов между их нахождением и использованием (стадия валидации).

Все методы ИАД также делятся на две большие группы в соответствии с принципом их работы с начальными данными обучения. В первой группе исходные данные могут быть сохранены в явно детализированной форме и использованы непосредственно для прогнозного моделирования и/или анализа исключений. Этот метод работы с данными является так называемым методом рассуждений, который основан на анализе ситуации. Основная проблема с этим набором методов заключается в их сложном использовании для больших объемов данных, хотя методы ИАД наиболее полезны именно для анализа больших хранилищ данных.

Во второй группе данные сначала должны быть взяты из исходной информации, а затем преобразована в формальные структуры (форма которых зависит от используемого метода).

Согласно предыдущей классификации, этот этап выполняется в фазе свободного поиска, которая в принципе отсутствует в методах первой группы. В результате итоги этого этапа, которые намного более компактны, чем сами массивы исходных данных, используются для прогнозного моделирования и анализа исключений. Кроме того, результирующие конструкции могут быть «прозрачными» (интерпретируемыми) или «черными ящиками» (нетрактуемыми). Перед использованием технологии Data Mining нужно тщательно проанализировать ее проблемы, ограничения и связанные с ней критические вопросы, а также понять, чего эта технология не способна сделать.

Интеллектуальный анализ данных (ИАД) – это междисциплинарная область науки, которая развивается на основе достижений прикладной статистики, распознавания образов, искусственного интеллекта, теории баз данных и других областей современной науки. Схема междисциплинарных связей в DataMining представлена на рисунке 3 [8].

В итоге можно сказать, что ИАД представляет собой спектр технологий и методик, выявляющих статистические взаимосвязи, закономерности и

правила в наборах данных. Технология ИАД в настоящее время включает в себя множество различных подходов к распознаванию знаний. У каждого из них есть свои преимущества и недостатки. Можно также сказать, что выбор конкретного подхода определяется спецификой предметной области и организацией данных.



Рисунок 3 – Междисциплинарные связи Data Mining

В данной главе был рассмотрен интеллектуальный анализ данных. Данная технология обычно применяется для поиска новых связей между элементами и их глубинных значений.

1.3 Технологические подходы к обработке знаний

Важной стороной технологического аспекта развития искусственного интеллекта является развитие интеллектуальных систем. Интеллектуальные системы настроены на решение специализированных задач, принадлежащих определенной предметной области.

Для ИС характерны следующие признаки [1 – 3]:

- развитые диалоговые и коммуникационные возможности;
- умение решать слабо формализованные задачи;
- возможность самоорганизации и самообучения;
- адаптивность и гибкость.

Каждому из приведенных выше признаков условно соответствует свой класс ИС. Разные системы могут обладать несколькими или одним признаком интеллектуальности с разнообразной степенью проявления.

Характерной чертой ИС является наличие знаний, необходимых для решения задач определенной предметной области. Конечно, возникает вопрос о том, что такое знание и чем оно отличается от обычных данных, обрабатываемых компьютерами [9 – 11].

Одним из типичных представителей систем искусственного интеллекта являются экспертные системы. Экспертная система (ЭС) от других автоматизированных систем на этапе ее использования характеризуется высокой интеллектуальностью, специализацией и ориентацией, которые направлены на решение задач в определенной области.

В ЭС выполняется процесс решения определенного класса задач, которые носят интеллектуальный характер. Качество результатов при этом должно быть сопоставимо с результатами, которые мог бы получить квалифицированный эксперт.

В составе ЭС можно выделить следующие компоненты:

- база знаний для хранения информации, описывающей предметную область, и правил, описывающих соответствующие преобразования данных в этой области;
- интерпретатор, который формирует ряд правил, обеспечивающих решение проблемы;
- компонента извлечения знаний, которая отвечает за формализацию процесса наполнения ЭС знаниями, полученными от эксперта.

Предназначением пользовательского интерфейса является обеспечение взаимодействия между прикладной системой и конечным пользователем или внешними подсистемами. Интеллектуальный интерфейс (интеллектуальные средства связи) – это интерфейс, который связан с процессом распознавания образов, голосового ввода и вывода, а также с техническим зрением. Такой

интерфейс основан на искусственном интеллекте программного и аппаратного обеспечения, который обеспечивает взаимодействие с компьютерами на естественных языках.

В состав интеллектуального интерфейса должны входить следующие компоненты [10, 12]:

- семантическая модель предметной области;
- программно-аппаратные средства логического вывода, используемые для решения задач пользователя и для накопления знаний;
- система общения пользователя и ЭВМ на языке, близком к естественному.

Каждая прикладная система должна быть построена на основе определенных знаний специалиста области, соответствующей поставленным задачам. Множество этих знаний называется моделью знаний (M). В дополнение к модели M система всегда имеет определенный механизм принятия решений в виде ряда определенных процедур.

В данной главе были рассмотрены различные прикладные системы для определенных задач. Важно, чтобы такие системы были удобны для пользователя.

1.4 Концепция Semantic Web, уровни представления Semantic Web

Технология World Wide Web является одной из наиболее динамично развивающихся в настоящее время. Можно сказать, что от динамики изменения технологии Web зависит облик современных информационных технологий.

Можно выделить следующие концептуальные особенности технологии Web [13]:

- работа с ресурсами;

- стандартизированная адресация;
- масштабируемость в условиях больших сетей;
- открытость, полнота и согласованность.

Ресурс является основной единицей обработки данных в технологии Web. Им может являться материальный пакет данных, ссылки, документы или страницы. Адресация ресурса основана на использовании специальной ссылки URI. При этом адресация может носить вложенный характер за счет того, что URI могут ссылаться друг на друга.

Понятие ресурса является довольно гибким. Одной из его важных характеристик является представление, которое зависит от применяемого протокола. Так, для случая протокола FTP под представлением может пониматься копия файла. В то же время такое толкование представления, например, для HTTP-протокола, вообще говоря, необязательно.

Стандартизированная адресация понимается как унификация механизма ссылок на ресурсы. Как отмечалось выше, каждому ресурсу можно сопоставить URI, который, в частности, может быть и URL-ссылкой. При этом формат представления URI является единообразным. Обращение и модификация URI происходит с применением команд HTTP-протокола.

Масштабируемость означает возможность работы Web в условиях роста Web-сайтов и оставаться эффективной при увеличении размера сети. Утвердительный ответ о том, что Web обладает такими свойствами, обусловлен двумя архитектурными свойствами Web. Первый аспект этих свойств состоит в децентрализации Web как сети. Это позволяет гибко вводить серверы данных и менять их роли и параметры. Добавление и обмен ресурсами между серверами, как правило, не требует регистрации этих операций где-либо еще [14, 15].

Другим свойством Web является то, что запросы и ответы на них, как правило, содержат все необходимые для запроса данные. Как правило, промежуточные данные не хранятся на сервере между запросами. Однако в

случае необходимости хранения промежуточных данных возможно кэширование ответа. Это повышает производительность запросов и уменьшает сетевой трафик. Данные, которые поддерживают некоторую хронологию транзакций, иногда называют состоянием системы. Транзакции Web не имеют состояний. Если есть потребность в сохранении информации между несколькими транзакциями, то сервер должен обеспечить для этого специальные средства.

Открытость, полнота и согласованность означает то, что узлы и ресурсы Web можно добавлять свободно. При этом доменные имена и их назначение не ограничивают возможность информационного обмена между Web-серверами.

В то же время гибкость Web в некотором смысле позволяет говорить о ее неполноте. Так, не гарантируется доступность и работоспособность отдельных связей и информационных ресурсов. Также можно говорить о несогласованном характере информации в Web. Это означает, что контент различных веб-страниц может противоречить друг другу. При этом изменение структуры сайтов или сетевых связей носит динамичный характер.

Традиционный Web имел целью обработку содержащейся в нем информации и был ориентирован на присутствие человека в этом процессе.

Можно выделить основные особенности организации информации, присущие традиционному Web [15, 16]:

- теги в разметки страницы не несут или имеют слабое семантическое значение, эта проблема отчасти решена в стандарте HTML 5, но остается актуальной;
- основная роль в структурировании данных отводится разметке. Таким образом, это также снижает процент содержательной информации;
- требования технического характера (кроссбраузерность, масштабируемость под разные устройства), могут увеличивать сложность разметки, и потенциально создавать перспективу возникновения возможных

ошибок в ее составе.

Ориентированный на восприятие человеком контент определяет преимущественно статистический характер поисковых алгоритмов. Так, важность или релевантность текста, или отдельно взятого слова определяется, в частности, введением некоторых весовых коэффициентов. Они на эмпирическом уровне определяют степень значимости слова в тексте. Другим вариантом оценивания является распределение частоты встречаемости слов в тексте. Подобные подходы можно применить и к ссылкам на отдельно взятые документы. Семантичность текста можно оценить, оценивая степень важности тега, в котором он приводится.

Из-за этого возникают следующие проблемы:

- отсутствие семантичности анализа информации на уровне поисковых систем и машин;
- возможная неудовлетворительность или отсутствие релевантности результатов поиска информации;
- возможность некорректной манипуляции структурой разметки страниц при SEO-оптимизации.

Термин «семантика» связан с изучением и изменением значения. В случае с Semantic Web это означает так же и обработку данных в Web компьютерами, а не только людьми [17 – 19].

Web и Semantic Web различаются присутствием во втором алгоритмически структурированных обработки и интерпретации данных. Во внимание так же должна приниматься и семантическая интероперабельность информационных ресурсов.

Семантика (от греч. *σημαντικός* – обозначающий) – это наука, в которой изучается понимание отдельных знаков, последовательных символов, а также других условных обозначений. Семантика определяет смысловое значение совокупности единиц языка. Этот язык может быть естественным или искусственным. Semantic Web можно рассматривать как этап эволюции сети

World Wide Web. В рамках новой концепции акцентируется внимание на автоматизацию обработки информации на стороне ЭВМ с учетом ее контекстного и смыслового характера. Внедрение этой технологии в существенной степени должно привести к расширению поисковых и аналитических возможностей браузеров и специализированных программных агентов. Также принципиально изменится и характер распределения и комбинирования информации. В Semantic Web разнородная информация рассматривается как единое целое. Каждому элементу информационного ресурса будет соответствовать машинный код. Он представляет из себя некоторый семантический тэг.

Основной математической моделью Semantic Web является теоретико-графовое представление информационных элементов [18]. Роль вершин в этом случае отводится понятиям предметной области. Описание понятий, в свою очередь, находится в определенной базе знаний. Дуги графа носят характер связей и отношений между понятиями. Такой подход к структурированию информации отражает контекстно-семантический характер предметной области. Таким образом, можно говорить о онтологической структуре представления знаний и информации в контексте Semantic Web.

В глобальном аспекте Semantic Web можно рассматривать как семантическую сеть, ориентированную на управление оптимизацией машинной обработки данных. Таким образом, данная сеть не выполняет функции логического вывода. Однако на ее стороне будет выполняться контроль валидности выполняемых операций и характера связей между ними.

Архитектура Semantic Web приведена на рисунке 4.

Из рисунка 4 видно, что, в отличие от традиционного Web, в Semantic Web существуют уровни онтологического представления информации. Так, онтологии поддерживаются в технологиях RIF, OWL, RDF. Формат XML работает преимущественно с древовидными структурами тегово-ориентированных записей. Ресурсы идентифицируются посредством URI, а

конечное представление ресурсов получается посредством серии проверок моделей данных на верхних уровнях архитектуры.

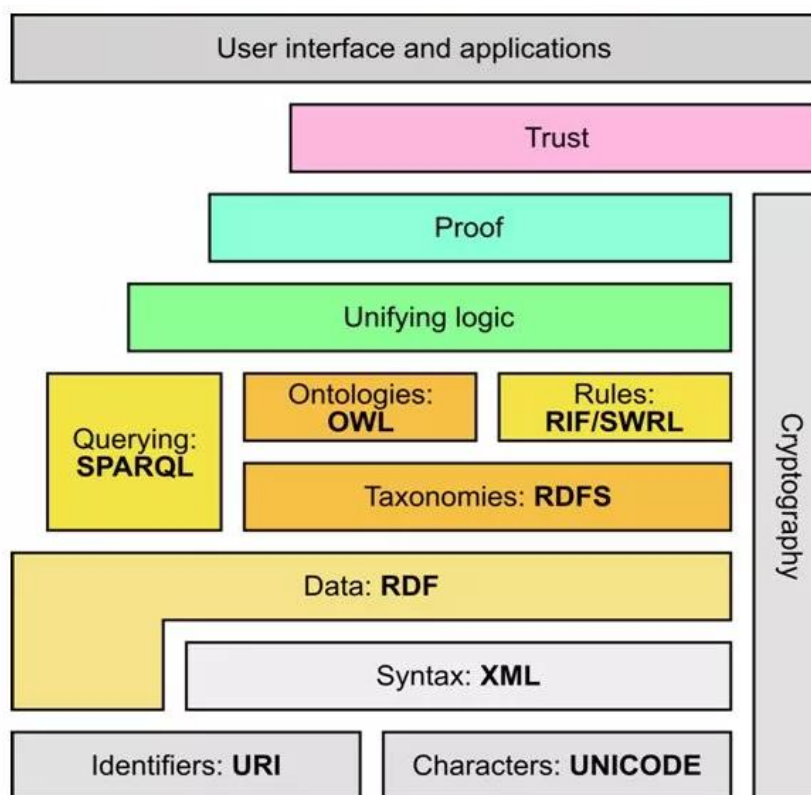


Рисунок 4 – Архитектура Semantic Web

Semantic Web носит кроссплатформенный и языково-независимый характер. При этом информация в сети данного типа обладает четко выраженной структурированностью [18, 19].

Согласно концепции Semantic Web, программы будут обладать функциями самоорганизации. Они будут обладать возможностями глубинного автоматизированного поиска нужных данных. Их основной особенностью является работа в рамках смыслового контекста анализируемой информации, а также выполнение адаптивного индексирования данных. Наряду с этим как промежуточный результат может быть выявление логических связей между элементами, а также выполнение промежуточных выводов. Конечным итогом представляется принятие решения на их базе.

Semantic Web должен обеспечивать поиск ресурсов Web по их

содержанию, а не только по ключевым словам. Кроме того, Semantic Web предназначена для предоставления доступа не только к статическим документам, которые содержат полезную информацию, но также и к службам, которые обеспечивают полезное поведение. Semantic Web включает в себя расширение веб-служб, позволяя программным агентам автоматизировать процедуры, выполняемые в настоящее время вручную, и вводить новые приложения, не внедренные сегодня.

В данной главе был рассмотрен Semantic Web. Данная технология имеет большие возможности для улучшения релевантности запросов.

Таким образом комплексное применение методов обработки знаний сможет упростить жизнь пользователям. В зависимости от уровня сложности системы можно использовать один из описанных в данной главе подходов.

2 Математические модели формализации знаний

2.1 Методы представления и извлечений знаний

Традиционно выделяют следующие модели представления знаний:

- продукционные системы;
- семантические сети;
- фреймы;
- нейронные сети.

Методы представления знаний с использованием модельной логики основаны на следующих принципах. Согласно положениям теории категорий для разрешимости метода должны быть установлены все возможные связи (морфизмы) между объектами. При жестких временных рамках, установленных для принятия решений, и неопределенности исходной информации это невозможно. Данные условия способны привести к формированию неправильной базы знаний экспертной системе, что будет приводить к существованию тупиковых, висящих и изолированных объектов, что в свою очередь приведет к неопределенности процесса управления обучением [20 – 24].

Представление знаний на основе аппарата семантических сетей заключается в рассмотрении предметной области как набора объектов (сущностей и концепций), а также связей (отношений) между ними.

Семантическая сеть – это модель для представления знаний, которая состоит из узлов и соединяющих их дуг, где узлы представляют собой объекты или понятия, в дуги – отношения между узлами. Сетевые модели представления знаний отличаются между собой типом используемых в них связей (отношений). Классифицирующие и функциональные сети различаются между собой в зависимости используемых в модели типов соединений [21].

Как модель для предоставления знаний семантическая сеть представляет собой иерархическую сеть, вершины которой являются информационными

единицами с присвоенными им индивидуальными именами. Дуги этой сети – это различные отношения между информационными единицами. Помимо этого иерархические отношения определяются как отношения структуризации, а неиерархические – отношениями любых других типов. Научной основой, благодаря которой можно предоставить данные семантической сети являются графо-теоретические методы.

Семантическая сеть содержит наиболее часто встречающиеся в тексте слова, которые несут основную смысловую нагрузку. Для каждой концепции формируется серия ассоциативных (семантических) отношений, т.е. список других понятий, в сочетании с которыми он был найден в предложениях текста. Считается, что чем чаще два понятия появляются вместе в предложениях в тексте, тем больше вероятность того, что они будут связаны по смыслу.

Одним из преимуществ семантических сетей по сравнению с другими схемами представления знаний является то, что связанные понятия имеют тенденцию объединяться в единый сетевой пакет. Это позволяет фильтровать информацию, не имеющую отношения к предметной области сети. Приведенное свойство позволяет говорить о аналогии с гипертекстами. Однако в рассматриваемом случае необходимо построить такие представления, которые могут быть автоматически интерпретированы.

Семантическая сеть как средство структурирования контента рассматриваемой предметной области может быть формально определена и описана с использованием теории графов. С помощью теории графов возможно предоставить простой, доступный и мощный инструмент для построения моделей и решения проблем упорядочивания объектов. Эта теория позволяет нам описывать операции, используемые для создания семантической сети на относительно высоком уровне формализации. Рассмотрим некоторые из них.

Непосредственное создание семантической сети, которая включает

определение концептов (узлов семантической сети) предметной области, которые формируют описанную модель поля и отношений (связей) между ними.

Модификация семантической сети может выполняться как на уровне локальных дополнений, так и путем внесения изменений в структуру семантической сети. Она может быть описана в рамках теоретико-графовых операций по добавлению, редактированию, удалению вершин и дуг сети и т.п.

Операции над семантическими сетями как отдельными объектами могут быть также описаны в терминах теоретико-графовой и множественной парадигм. Однако при формировании новой сети, основанной на объединении или пересечении существующих, необходимо добиваться непротиворечивости используемых в ней концептов и связей между ними.

Вместе с тем, помимо приведенных выше действий с семантическими сетями можно выполнять действия по декомпозиции и агрегированию. В этом случае на базе сети строятся структуры, которые могут играть роль информационных паттернов. Их замкнутое множество которых может повысить качество описания предметной области. Также к семантическим сетям можно применять различные критерии сравнения и оценивания.

Преимущества семантических сетей как средства представления знаний заключаются в том, что в них заключены большие выразительные возможности, такая система знаний естественна и наглядна, она представлена графически, структура фраз в семантической сети близка к структуре в естественном языке. Помимо этого научная база семантических сетей опирается на уже проверенную теорию графов.

Недостатком является отсутствие строгих правил для манипуляции знаниями, а значит для каждой конкретной задачи потребуются разработка собственных эвристик.

Фреймы в ряде работ рассматриваются как развитие и обобщение семантических сетей. Фреймы были первоначально предложены как своего

рода концепция для представления знаний человека, который находится в трудной ситуации. Под фреймом понимается структура данных, которая должна представлять конкретную ситуацию.

Фрейм состоит из слотов, каждому из которых присваивается значение определенного типа (включая ссылку на другой фрейм), указатель наследования для свойств родительского фрейма, значение по умолчанию, демоны и присоединенная процедура. Демон – это процедура, которая запускается автоматически при выполнении определенного условия, в зависимости от состояния слота или действий над ним (например, если его значение не определено при доступе к слоту или изменяется значение слота). С помощью фреймов мы можем не только описывать отдельные факты, но и сложные ситуации, а также представлять операционные знания и контролировать метазнания с помощью демонов и присоединенных процедур.

Различают фрейм-прототип и фрейм-экземпляр. Первый – это описание класса таких ситуаций, а второй – спецификация первого при определенных условиях. Система фреймов – это иерархия фреймов и общие механизмы их обработки, например, к ним относятся порождение, удаление и сопоставление структур фреймов, демоны и отправка сообщения в прикрепленную процедуру. Фреймовый подход для представления знаний о программном обеспечении имеет ряд преимуществ и недостатков. Преимущества фреймового формализма заключаются в следующем [21 – 25]:

- фреймовый подход позволяет ввести модульность в представление знаний, чтобы обеспечить механизм наследования и ожидания, т.е. описать не только статические объекты, но и временные изменения объектов; включение продуктов в их слоты; обеспечение концепцию «умолчания», если элементарные концепции наследуют свойства более общих концепций в сети фреймов;
- фреймы объединяют преимущества декларативного и процедурного представления знаний (комбинация серии описаний и

связанных наборов интегрированных методов обработки);

– аналогия между терминалами фрейма и формальными параметрами языков программирования позволяет формально связать модульный организационный принцип прикладного программного обеспечения с информационными структурами, которые были созданы с использованием фреймовой организации знаний.

К недостаткам фреймового подхода относятся следующие:

– фрейм может представлять собой любую сложную структуру данных, но методов для выделения фреймов для программных задач не было разработано;

– каждый объект в ПО может обладать очень обширными связями с другими объектами, а значит для описания типа этого объекта может потребоваться слишком громоздкий фрейм;

– в фреймовых системах интерпретируются только иерархические отношения, и нет возможности произвольных отношений с действиями;

– фреймовый формализм хорошо решает проблему управления целью и плохо – проблему управления данными;

– фреймы имеют отличные выразительные возможности, гибкость, модульность и удобное представление знаний, но механизмы манипулирования и вывода для фреймового представления развиты слабо.

Нейронные сети являются адаптивными системами для обработки и анализа данных. Это математическая структура, которая имитирует некоторые аспекты работы человеческого мозга и демонстрирует его возможности, такие как способность к неформальному обучению, способность обобщать и кластеризовать неклассифицированную информацию, а также способность прогнозировать на основе ранее представленных временных рядов.

Основа каждой нейронной сети – относительно простые, в большинстве случаев того же типа, элементы (клетки), которые имитируют функцию нейронов мозга [4, 26]. Далее нейрон будет пониматься как искусственный

нейрон, то есть как НС-ячейка. На вход искусственного нейрона поступает определенный набор сигналов, каждый из которых является выходом другого нейрона. Каждый вход умножается на соответствующий вес, аналогично синаптической производительности, и все продукты суммируются для определения степени активации нейронов. На рисунке 5 представлена модель, реализующая эту идею.

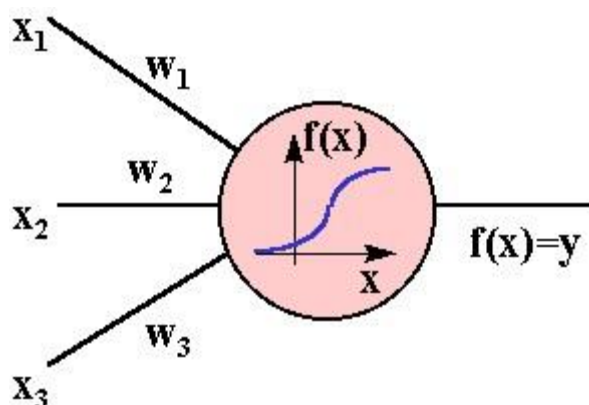


Рисунок 5 – Искусственный нейрон

Множество входных сигналов, обозначенных $x_1, x_2, x_3 \dots x_n$, поступает на искусственный нейрон. Эти входные сигналы, в совокупности обозначаемые вектором X , соответствуют сигналам, приходящим в синапсы биологического нейрона. Каждый сигнал умножается на соответствующий вес $w_1, w_2, w_3 \dots w_n$, и поступает на суммирующий блок, обозначенный буквой S . Каждый вес соответствует «силе» синаптической связи [27].

Сформулированные особенности организации нейрона позволяют предположить широкий диапазон изменения свойств нейросети. Так, в зависимости от конкретной организации нейронов и методов настройки их весов, нейросеть может обладать различными ассоциативными и синергетическими свойствами. Это предоставляет качественно иные возможности по работе с слабо формализованными данными или в условиях, когда полной информации о исследуемом объекте не существует.

В данной главе рассмотрены различные модели представления и извлечения знаний. Чаще всего знания представляются в виде иерархических

систем со строго заданными связями.

2.2 Методы представления и хранения семантической информации

Это может быть объяснено тем, что подобные модели основываются на формальной (дескриптивной) логике, а так же тем, что для работы с ними в рамках концепции Semantic Web был разработан ряд стандартных технологий для создания, описания и использования онтологических моделей. Неформально онтологией называют некоторые описания представлений о внешнем мире, которые можно применить к некой предметной области. Так же она содержит описание и интерпретацию терминов в границах данной предметной области.

На формальном уровне онтология является системой, которая состоит из наборов понятий и наборов понятий об этих утверждениях. На их основе можно описывать классы, функции, отношения и индивиды. Одно из самых известных определений для онтологии дал Т. R. Gruber: «Онтология – явная спецификация концептуализации». Немного изменил это понятие R. Studer: «Онтология – это формальное точное описание (спецификация) согласованной концептуализации».

Каждое из свойств онтологии можно определить следующим образом [19]:

- формальная: спецификация (подробное описание) представлена на формальном машиночитаемом (обрабатываемым программой) языке, в концепции Semantic Web это означает использование стандартного языка RDF;
- явная: это свойство подразумевает возможность повторного использования (уменьшение зависимости от конкретной задачи, компонента программного обеспечения или другой онтологии) и практичность, которая означает, что разработанная онтология должна быть понятной для

разработчика или некоторого сообщества и соответствовать задачам и приложениям, в которых она используется;

- спецификация: онтология – это описание артефакта, которое в целом не зависит от описанной сущности;
- согласованная: онтология имеет смысл, только в том случае, если члены некоторого сообщества, в котором она будет использоваться, согласны с ее содержанием, цель и преимущества онтологий в контексте Semantic Web заключаются в том, что они поддерживают функциональную совместимость создателей электронных ресурсов и их пользователей;
- концептуализация: это структура реальности, которая рассматривается независимо от словаря предметной области и конкретной ситуации, следовательно, онтология должна быть согласованной и полной в определенной области.

Основными компонентами онтологии являются:

- 1) классы понятий описываемой предметной области;
- 2) отношения, которые существуют между этими понятиями.

Онтологии способны предоставить более подробные данные по описанию предметной области, чем иные модели, такие как, например, тазаурус, словари или таксономия. Онтологии, в отличие от тезаурусов и таксономий, обладают более гибкой структурой, а также способны определить любые отношения. Тогда как таксономии определяют исключительно иерархические отношения между понятиями, тезаурус же определяет и такие отношения, как гипонимия, гиперонимия и синонимия.

У самого понятия «онтология» имеется несколько значений. Данное понятие пришло к нам из философии. Она классифицировала онтологию как дисциплину, которая изучает фундаментальные принципы бытия. В рамках онтологии рассматривалось учение о сущем, о бытии. В ней изучались фундаментальные принципы и формы, в ней давались наиболее общие определения и категории бытия.

В рамках искусственного интеллекта и информатики онтология – это точная спецификация концептуализации [21, 22]. Концептуализация – это некая модель предметной области, которая часто является абстрактной. Система понятий предметной области описывается именно через нее.

Многообразная структура предметной области содержит в себе понятия, объекты и прочие сущности, предположительно являющиеся известными элементами. Помимо объектов в ее состав входят определенные между ними отношения. Концептуализация представляет собой упрощенную, абстрактную точку зрения на мир, она предоставлена для некоторых целей. «Спецификация» подразумевает под собой описание системы понятий в явном виде.

Онтологии используются для формальной спецификации отношений и понятий, коротыми характеризуется некая область знаний. Преимущество онтологий как средства предоставления знаний заключается в том, что они обладают формальной структурой [22], благодаря чему упрощена их компьютерная обработка. Онтологии можно использовать для решения разнообразных задач, обычно в случае работы над одной предметной областью, для того, чтобы появилась возможность накопления знаний для повторного их использования, а также чтобы создавать модели. Благодаря этому специалисты одной предметной области смогут лучше манипулировать знаниями, обмениваться опытом или обнаруживать новые закономерности.

Онтологии часто используются в тех случаях, когда требуется обработка данных с учетом их семантики, например, если требуется повысить релевантность (эффективность поиска) в сети интернет [23]. В последнее время широкое распространение получили интеллектуальные системы на основе онтологий.

Для создания онтологии предполагается определить класс объектов и описать их отношения с помощью одного из формальных языков. Обычно процесс построения онтологии начинается с создания базовой онтологической

структуры, таксономического ядра онтологии, которое представляет собой основные понятия предметной области и родовидовые связи между ними. Очень важно, чтобы успешно построить искомую онтологию, чтобы ею занимался специалист, хорошо разбирающийся в этой предметной области. Традиционно именно хорошо разбирающиеся в своей предметной области эксперты и привлекаются для создания онтологий. Поскольку ручная работа отнимает много сил и времени у экспертов, актуальной проблемой является автоматизация построения онтологии.

Одним из наиболее эффективных методов, используемых для автоматизации построения онтологий, является использование методов автоматического извлечения данных из текстов, а это не вынуждает прибегать к услугам дорогостоящих специалистов. Многие типы источников содержат информацию о терминах и отношениях между ними, к таким источникам относятся: тексты с разметкой, базы данных, словари, неструктурированные тексты. Самым перспективным подходом представляется тот, который основан на извлечении терминов и отношений между ними напрямую из корпуса неструктурированных текстов, потому что именно такие тексты встречаются чаще всего. Большая часть информации хранится в подобном виде. Большое количество исследований за недавнее время было посвящено онтологиям. Несмотря на большое внимание, уделяемое данной проблеме, удовлетворительного решения для автоматического построения онтологий пока не существует.

Онтологии сильно различаются по ряду параметров, и исследователи определяют различные основания их классификации.

Можно выделить классификацию онтологий по цели создания. В рамках этой классификации выделяют следующие объекты (рисунок 6).

Цель создания онтологии представления состоит в описании области представления знаний. Благодаря ему можно создать язык для спецификации соподчиненных онтологий более низких уровней иерархии.

Онтологии верхнего уровня структурируют знания, являющиеся общими для всех предметных областей. Данные онтологии могут применяться многократно.

Онтологии предметных областей имеют схожее назначение с онтологиями верхнего уровня. Однако их описание ограничено предметной областью. Роль предметной области может быть отведена какой-либо дисциплине (авиация, медицина, культура).

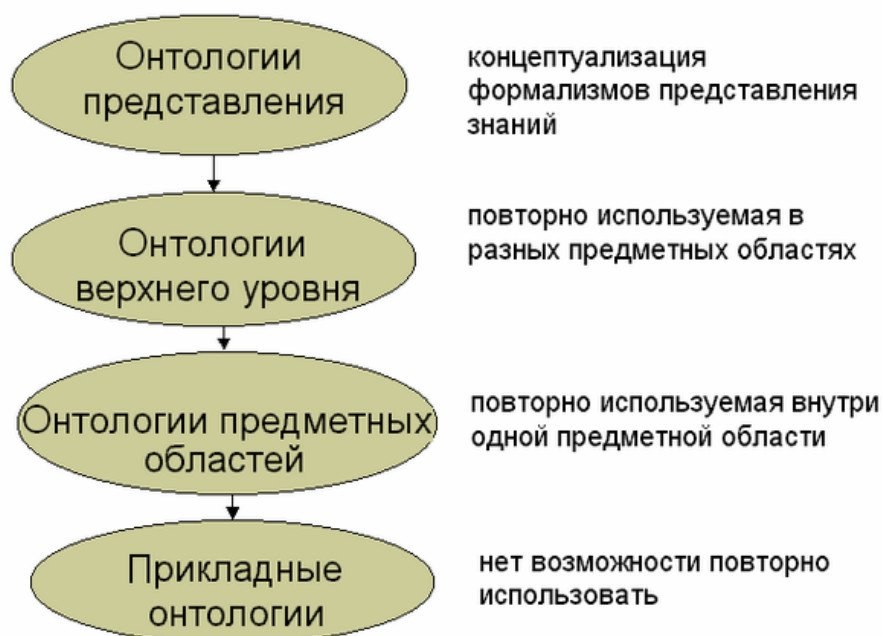


Рисунок 6 – Классификация онтологий по цели создания

Назначение прикладных онтологий состоит в описании концептуальной модели конкретной задачи или приложения. Они содержат наиболее детальную информацию, отражающую специфику предметной области.

Другой вариант классификации онтологий состоит в их ранжировании по содержанию. Структура классификации для этого случая приведена на рисунке 7.

Общие онтологии описывают концепты, обладающие наиболее обобщенными свойствами. Благодаря своему уровню абстракции они не зависят от конкретной проблемы или области. В эту категорию попадают и

онтологии представления, и онтологии верхнего уровня.

Онтология, ориентированная на задачу – это онтология, ориентированная на конкретную задачу. Задачи, которым может быть посвящена онтология, могут иметь разнообразную направленность. При этом онтология задачи использует специализацию терминов, представленных в онтологиях верхнего уровня (смотри рисунок 6).

Предметная онтология (или онтология предметов) описывает реальные предметы, используемые в практических действиях.

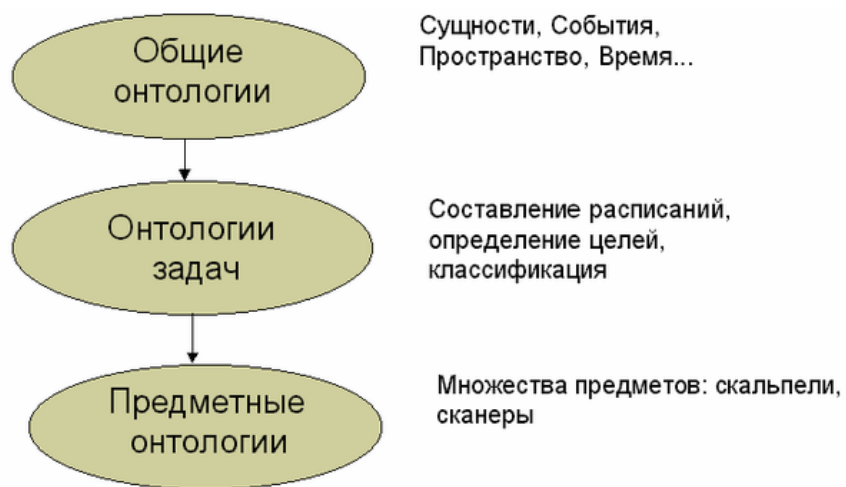


Рисунок 7 – Классификация онтологий по содержанию

Связь между онтологиями верхнего уровня и нижних уровней приведена на рисунке 8.



Рисунок 8 – Связь между онтологиями верхнего уровня и нижних уровней

Вместе с описанным подразделением все онтологии могут быть разделены на глубинные и поверхностные. Поверхностные онтологии основаны на семантике поверхности, они определяют понятия по значению слов. Однако это ставит проблему о том, сколько смыслов должно быть присвоено каждому слову. Глубинные онтологии используют глубокую семантику.

Онтология – это иерархически структурированный набор терминов, описывающих предметную область, которая может использоваться в качестве исходной структуры для базы знаний. Онтология основана на свойствах, классах, объектах и ограничениях, которые реализуют идею объектов как группы объектов, которые характеризуются определенной группой свойств. Эти сущности имеют определенные отношения друг с другом и сгруппированы в классы в соответствии с определенными признаками (свойства и ограничения). В результате полного описания объектов и их свойств предметная область представляется в виде сложной иерархической базы знаний, с помощью которой можно выполнять «интеллектуальные» операции, такие как семантический поиск, а также целостность и достоверность данных.

Extensible Markup Language (XML) – это язык разметки, который определяет набор правил для кодирования документов в формате, удобном для чтения компьютером. Это простой и гибкий формат. XML был разработан как инструмент для реализации задач издательского дела и становится все более популярным, когда данные размещаются в Интернете и организуется межпрограммный обмен данными. Последняя задача как раз отвечает цели его проектирования.

XML-файл – может рассматриваться как набор тегов для определения объектов и их атрибутов. В этом отношении состав данных очень похож на язык разметки HTML. Тем не менее, есть фундаментальная разница между

XML и HTML. XML использует теги, которые были прописаны пользователями. Система тегов в HTML была изначально типизирован [4 – 8].

Структура XML-документа представляет собой дерево элементов.

Некоторые из этих элементы имеют атрибуты и содержимое.

Например, `<tag name="name1">value</tag>`. Здесь `<tag>`, `</tag>` – открывающий и закрывающий теги элемента, `name` – атрибут, `name1` – значение атрибута, `value` – содержимое.

Пример структуры XML-документа приведена на рисунке 8.

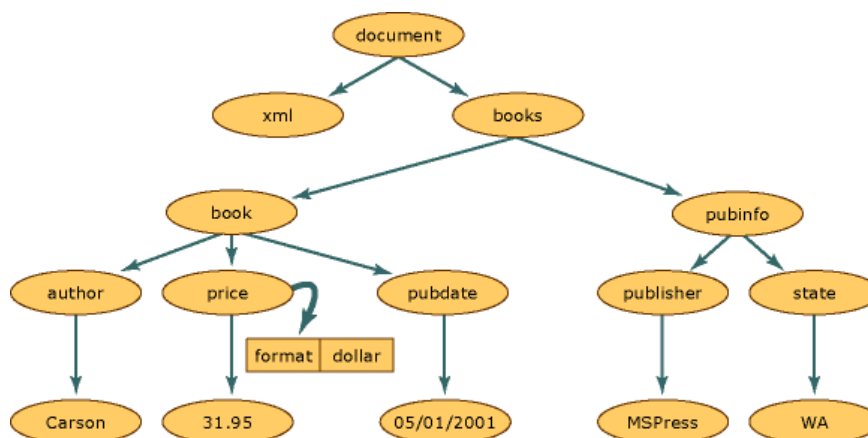


Рисунок 8 – Структура XML-документа

Файлы XML и файлы других расширений на основе XML получили очень широкое распространение. XML-файлы могут хранить данные разной природы и состава. Они могут включать в себя описание настроек приложения, описание графических объектов и базы данных знаний. Поскольку XML-файлы часто являются текстовыми файлами, их можно легко обрабатывать в текстовом редакторе любого уровня. В то же время можно выбрать формат Unicode в качестве текстового формата с многоязычной поддержкой.

XML также основан на принципе простоты и поддержки данных всех видов. XML – это больше, чем просто формат файла. XML задает необходимый фундамент для обработки самоописываемых документов с иерархической структурой. Они могут как храниться на жестком диске, так и находиться в памяти.

Resource Description Framework – это стандартная модель для обмена данными по сети. RDF предоставляет общую структуру для описания интернет-ресурсов, например, веб-сайта и его содержания. Он расширяет веб-структуру с помощью URI для именования отношений между сущностями. Обычно утверждение принимает форму «субъект-предикат-объект». Такая структура называется трипplerом.

Эта структура соединения образует направленный помеченный граф, ребра которого представляют собой ссылку, которая называется предикатом и расположена между двумя ресурсами, узлами графа.

Описание RDF (такие описания часто называют метаданными или «данными о данных») может включать авторов ресурса, дату создания или обновления, организацию страниц на веб-сайте (карту сайта), информацию, связанную с содержанием целевой аудитории или рейтинга, содержат ключевые слова для сбора поисковых данных, тематические категории и так далее.

Пример реализации приведен на рисунке 9.

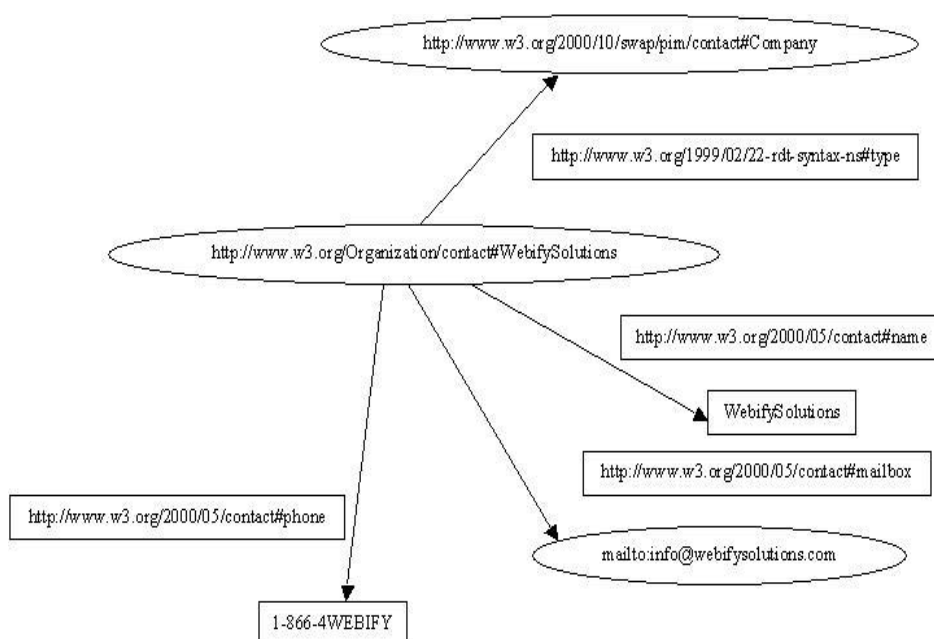


Рисунок 9 – Пример RDF

RDF является универсальным методом декомпозиции знания на отдельные части. Само разбиение на составляющие происходит в

соответствии с некоторыми правилами, в которых выражена семантика (смысл) этих частей. Метод, решающий такие задачи, должен обладать достаточной простотой и гибкостью. Последнее необходимо для того, чтобы с его помощью можно было представить любой факт. С другой стороны, он должен быть достаточно выразительным и обладать высоким уровнем структурированности. В таком случае факт имеет представление, пригодное для конструктивной обработки программами на ЭВМ со знаний, выраженных в формате RDF.

Из приведенных выше соображений можно сделать вывод о том, что с использованием естественных языков можно по-разному реализовать некоторые структуры данных. В этом смысле можно говорить о некоторой связи между RDF и XML.

XML также основан на принципе простоты и поддержки данных всех видов. XML – это больше, чем просто формат файла. XML задает необходимый фундамент для обработки самоописываемых документов с иерархической структурой. Они могут как храниться на жестком диске, так и находиться в памяти. Отличает RDF от XML то, что RDF разработан для представления знаний с учетом распределённого характера фактов. Из того, что RDF в большей мере ориентирован на хранение знаний, а не данных, означает, что RDF отчасти отражает смысловую структуру контента. Всё, что упомянуто средствами RDF, содержит некоторое значение.

Совокупность фактов в достаточном количестве может быть формой представления знаний. Основанные на RDF стандарты, включая RDFS и OWL, дополняют семантику RDF. Благодаря им потенциально возможно построение логических выводов на основе совокупности данных.

Исходя из приведенной особенности RDF, можно сказать, что семантическая составляющая в XML выражена слабо. Это обуславливается достаточно общей структурой XML-документа. XML-узлы в общем случае не обязательно могут связываться с какими-либо обособленными понятиями.

Стандарт XML не указывает на способ извлечения фактов из документа.

Таким образом, в XML-стандарте могут быть действительно успешно обработаны, потому что RDF определяет некоторый начальный уровень смысла.

Другой особенностью RDF является то, что данная технология эффективно работает с распределённой информацией. Это значит, что RDF-приложения могут объединять данные из RDF-файлов, опубликованных в Интернете. Это дает возможность обобщения данных и вывода новых фактов, ранее не существовавших ни в одном отдельно взятом документе.

Такой уровень синтеза информации может быть достигнут двумя способами. Во-первых, при связывании документов можно использовать в них общие словари. Во-вторых, в любом документе можно разрешить применять любой словарь. Все вместе взятое приводит к существенному повышению гибкости при поиске и представлении фактов о широком диапазоне вещей. При этом сама информация будет браться из широкого круга источников.

В данной главе были рассмотрены модели представления и хранения семантической информации. Разница хранения обычной и семантической информации заключается в более детальном описании объектов и их значений. Придается больший вес их значениям.

2.3 Веб-технологии поиска и обработки семантической информации

Общей особенностью существующих систем поиска и анализа информации является отсутствие семантической структуры, которая описывает содержание и уровни понятий. Это включает, в частности, концептуальную систему предметной области, в которой осуществляется поиск.

Наиболее выраженной тенденцией в современных исследованиях является обработка неструктурированных массивов данных в компьютерных

сетях. Конечная цель – обновить данные и преобразовать их в знания. Это требует создания подходящих инструментов и сред для работы с ними. Эта проблема приводит к разному интересу к методам и технологиям искусственного интеллекта.

Одной из особенностей деятельности, связанной с информационными технологиями, является использование специалистами из разных областей удаленных информационных ресурсов. Эта особенность позволила выделить группу пользователей, которые, являясь экспертами в своей области, используют мультитемный поиск информации с использованием одного или нескольких поисковых критериев. Это позволяет им осмысленно определять границы своей деятельности, что дает возможность составить концептуальную модель предметной.

Приведенные особенности создают необходимость наличия среди инструментов поиска информации такого программного обеспечения, которая позволит онтологическое моделирование домена и его обработки. Также было бы важно иметь средства для создания и обработки запросов к системе, а также средства для сбора результатов поиска и обработки результатов. Ввиду вышесказанного можно выделить следующие основные поисковые задачи:

- поиск документов в соответствии с запросами на естественном языке;
- ранжирование документов по степени близости к запросу, при этом возможна оценка его релевантности документа;
- просмотр пользователем информации о документе;
- автоматизация создания описания документов;
- возможность развития системы в зависимости от полученных результатов поиска.

Можно выделить несколько методов выполнения информации в сети Web. Первый вариант предполагает наиболее простой вариант доступа. В этом случае можно непосредственно выходить на Web-страницу, заранее зная

ее местонахождение. Во втором случае, помимо текущего контента, используются гипертекстовые ссылки, исходящие из Web-страницы. Они создают ассоциированные связи текущей страницы с другими страницами. Они, в свою очередь, имеют необходимую дополнительную информацию по смежным темам. Третий вариант используется для передачи по кабельному телевидению конечному пользователю страниц, непосредственно связанных с интересующими его темами. Четвертый способ использования заключается в том, что поисковые системы позволяют пользователю точно определить тип информации, которую необходимо найти. Она также соответствует описанию, содержащемуся в запросе.

Ниже приведены некоторые из основных современных проблем с получением информации.

- при получении пользователем большого объема информации в результате автоматизированного поиска ее просмотр и анализ является достаточно трудоемким;
- выбор информации, осуществляемый человеком, может быть неточным и проводится в разных вариантах, это, в свою очередь существенно повышает сложность ее поиска;
- пользователь при поиске информации, как правило, неформально определяет цель поиска, и это приводит к тому, что ключевые искомые понятия могут быть сформулированы нечетко.

Под моделью документа принято понимать ряд определенных функций, которые учитываются поисковой системой во время обработки. Признаки условно делятся на два типа: атрибуты, которые присваиваются тексту документа, и атрибуты источника информации, которые не присваиваются тексту. Функции, связанные с текстом документа, включают:

- наличие или отсутствие терминов запроса в тексте, под термином в данном случае подразумевается отдельное слово (минимальная синтаксически значимая единица текста) естественного языка.

- взаимное расположение терминов в тексте, слова, расположенные рядом относительно друг друга, составляют предложения, фразы, абзацы, а информация о их взаимном расположении, очевидно, имеют ценность для информационного поиска;
- форматирование текста/разметка документа, отдельные слова, предложения или абзацы в тексте могут различаться по шрифту и стилю (документы – языком разметки (HTML, XML)), эта информация учитывается при поиске основных моментов важных частей текста;
- логическая структура текста в документе, текст разделяется на абзацы, параграфы, главы, разделы, данное деление учитывает смысл источника информации, что является важной составляющей при его обработке – гиперссылочная структура документов.

Источники информации обычно содержат ссылки на другие источники информации в виде гиперссылок или в библиографии. Эта информация характеризует отношения между документами и служит основой для утверждения, например, в отношении тематического сходства документов. В информационно-поисковых системах в веб-пространстве такие функции называются «мета-атрибутами». Сюда входят данные об URL-адресе документа в веб-пространстве, информация о том, когда документ был создан или изменен, и другие. Эти особенности документа классифицируются следующим образом:

- информация об источнике, в поисковых системах в веб-пространстве эта информация извлекается через адрес-URL (Universal Resource Locator) источника информации, и других системах, таких как информационные системы или системы документооборота, свойство, учитывающее информацию об источнике, обычно имеет более сложную структуру, чем поисковые системы в веб-пространстве, информация о моменте создания и модификации источника информации;
- тематическая рубрикация источника информации, современные

поисковые системы в веб-пространстве используют информацию из каталогов, которые заполняются экспертами вручную, либо путем добровольного голосования пользователей, либо с помощью инструментов автоматического рубрикации;

– поведение пользователя по отношению к источнику информации, такая информация характеризуется частотой, с которой пользователи просматривают этот документ, а также указывает на популярность и полезность документа.

Большое количество различных функций документа означает, что ни одна современная поисковая система в веб-пространстве не использует только один атрибут, но и не использует их все. При разработке информационно-поисковых систем необходимо определить модель документа, использование которой, по запросу пользователя, обеспечивает наибольшую эффективность поиска при сборе документов и ее влияние на качество поиска информации. Использование сложных моделей документов обычно обеспечивает более высокое качество результатов поиска, но в то же время требует огромных ресурсов (время обработки, объем памяти).

Поисковый процесс представляется четырьмя стадиями (рисунок 10).

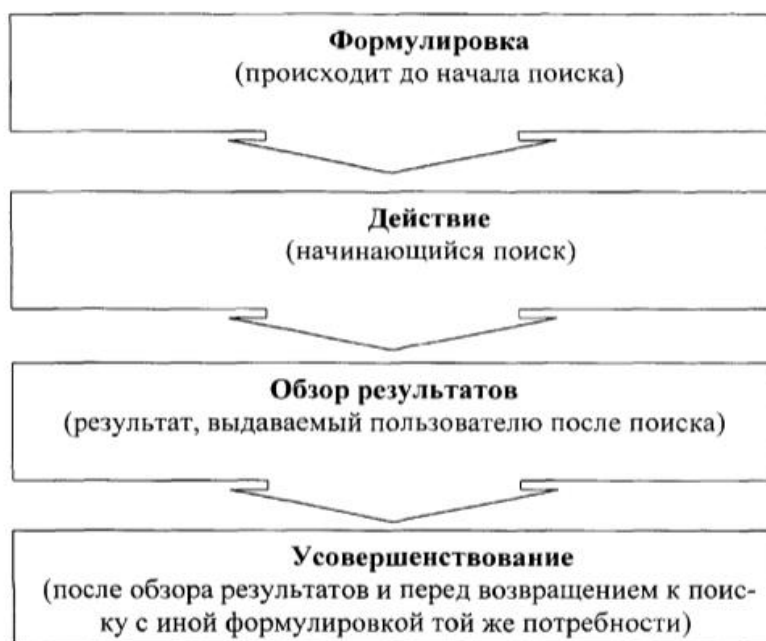


Рисунок 10 – Стадии процесса поиска информации

Более подробное описание поиска информации состоит из следующих этапов.

- естественно-языковая формулировка поискового запроса;
- выбор системы поиска информации и представление записи запроса на конкретных информационно-поисковых языках;
- выполнение команд пользователя на поиск информации;
- анализ и предварительный просмотр полученных списков ссылок на документы;
- переход по выбранным адресам с целью получения искомого контента;
- предварительный просмотр системой содержимого найденных документов;
- выполнение сохранения релевантных документов для последующего изучения;
- анализ всего корпуса полученных и сохраненных документов;
- в случае отсутствия релевантности результатов целям запроса выполняется повторение поиска.

Рассмотрим возможности наиболее популярных поисковых систем.

Одной из наиболее популярных поисковых систем является Google (рисунок 11).

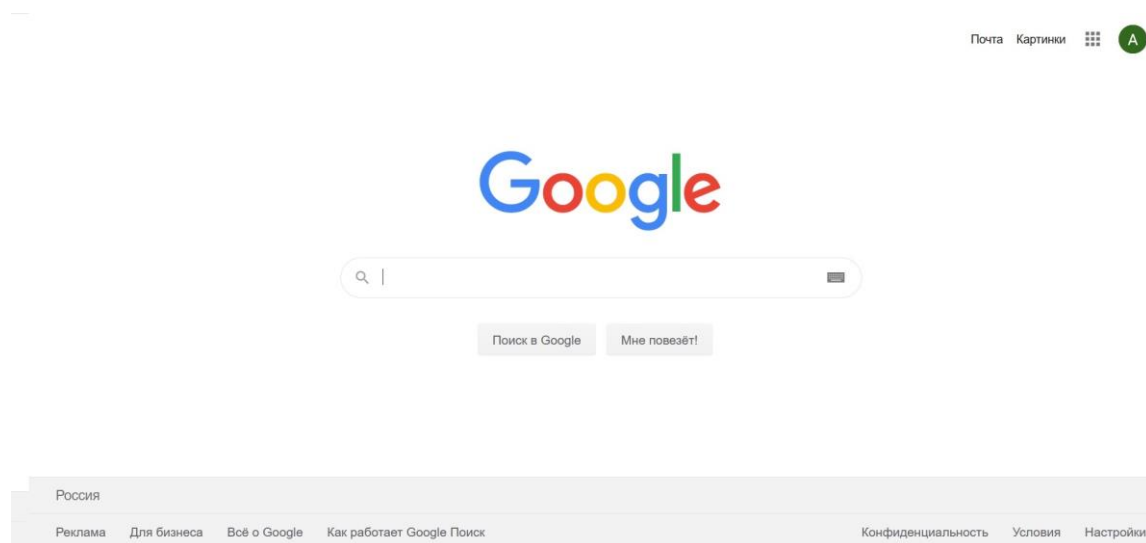


Рисунок 11 – Вид поисковой системы Google

Помимо прочего Google позволяет искать последние новости по заданной теме. При этом ведется просмотр в журналах и газетах с распределением выдаваемой информации по актуальности. Google считается одной из лучших поисковых систем, однако у нее имеются недостатки, в частности на широкие по смысловой нагрузке запросы Google выдает очень много лишней, не относящейся к теме запроса информации.

Среди всех зарубежных систем, Google дает самую высокую релевантность результатов для простого поиска. Такой результат обусловлен поддержкой семантического поиска информации, который определяет смысл в запросах пользователя, а не только по словам. Это дает возможность получения качественного результата, при том, что в запрашиваемых пользователем страницах могут не содержаться введенные ключевые слова. Google доступен для использования на многих языках мира.

В Google на высоком семантическом уровне реализована адресная реклама. Во многих случаях содержание рекламного баннера на странице содержит в себе элементы темы предыдущих поисковых запросов. Так же в Google имеется возможность расширенного поиска.

Одной из наиболее известных отечественных поисковых систем является Mail.ru (рисунок 12).

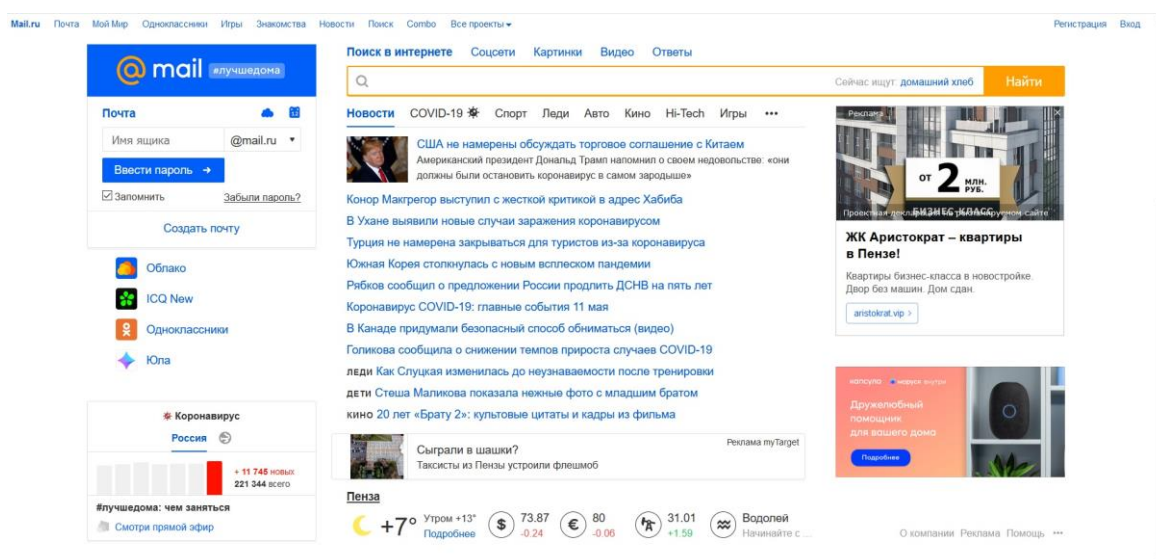


Рисунок 12 – Вид поисковой системы Mail.ru

ПС Mail.ru сочетал большой объем базы данных, высокую релевантность результатов поиска, удобную форму вывода запросов и много дополнительного сервиса. Данная система, как и многие другие, имеет возможность исправления опечаток в запросе. Падежи и числа употребляемых лексических единиц в данном случае не имеют значения ввиду того, что в системе существует учет особенностей русского языка. Это в значительной степени защищает от неточностей, возникающих при случайном сочетании ключевых слов, например, в начале и в конце документа.

С начала 1998 года и по настоящее время Mail.ru значительно вырос, помимо возможностей поисковых систем он предоставляет пользователям множество разных услуг, таких как, например, доступ к игровому portalу, новостному сервису, календарю, киносервису, ТВ-программе или сервису «Ответы» (рисунок 13). Помимо стандартных проектов, у Mail.ru есть несколько сторонних, таких как Юла, Mars.me или Delivery club. Помимо прямого доступа к этим сервисам через браузер существуют так же мобильные приложения, например, для прямого доступа к электронной почте, а также облаку, поиску или новостям.

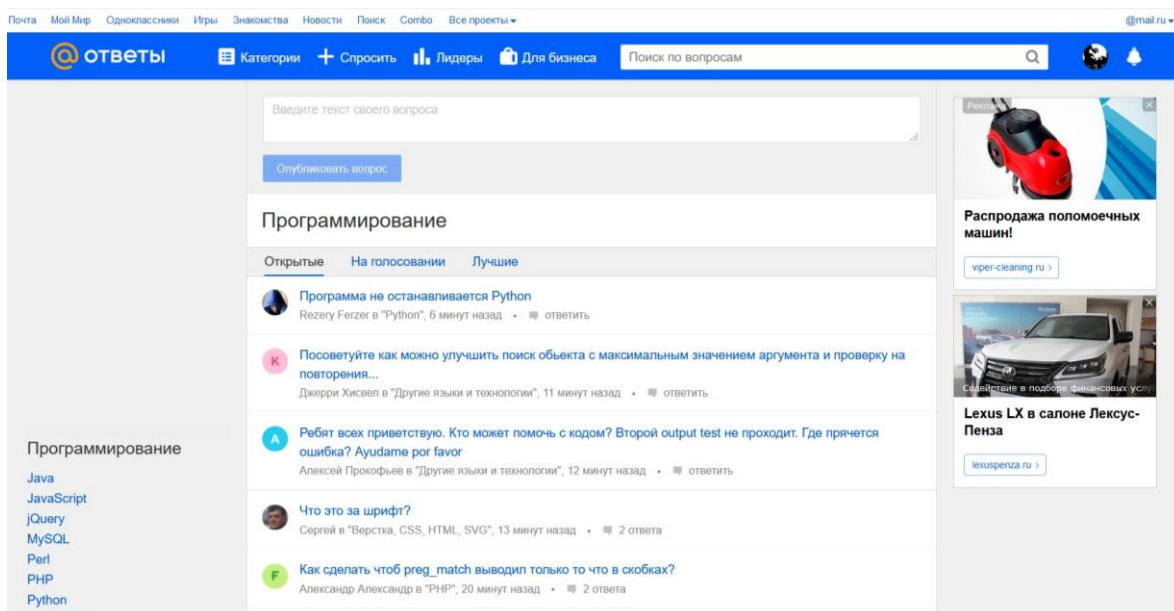


Рисунок 13 – Вид сервиса «Ответы» на mail.ru

ПС Яндекс – русскоязычная лингвистическая оболочка для поисковых целей, в которой реализован механизм анализа морфологии русского языка, в

результате чего создана система, снимающая проблему многовариантности русского языка. Однако результаты стандартного поиска на Яндекс не слишком велики. Фактическая релевантность первых ссылок может быть и высокой и крайне низкой, поэтому целесообразно воспользоваться средствами расширенного поиска и более точно сформулировать критерии отбора данных.

Яндекс является крупнейшим российским поисковым мультипорталом. Яндекс специализируется на поиске широкого спектра информации. Яндекс поддерживается большое число бесплатные веб-сервисов. К ним можно отнести такие, как электронная почта, платежная система, блоги, сервис поиска товаров и услуг, информация о погоде, пробках, и т.п. Так же, как и у Mail.ru, у Яндекса существуют мобильные приложения для доступа к их сервисам. Так же существуют сервисы, разработанные для ведения бизнеса, такие как Яндекс.Директ, Яндекс.Касса или Яндекс.Вебмастер. Яндекс.Вебмастер позволяет следить за статистикой запросов, по которым сайт оказывается в поиске, проверить, как отображается сайт на мобильных устройствах, а также провести техническую диагностику.

Вид ПС Яндекс приведен на рисунке 14.

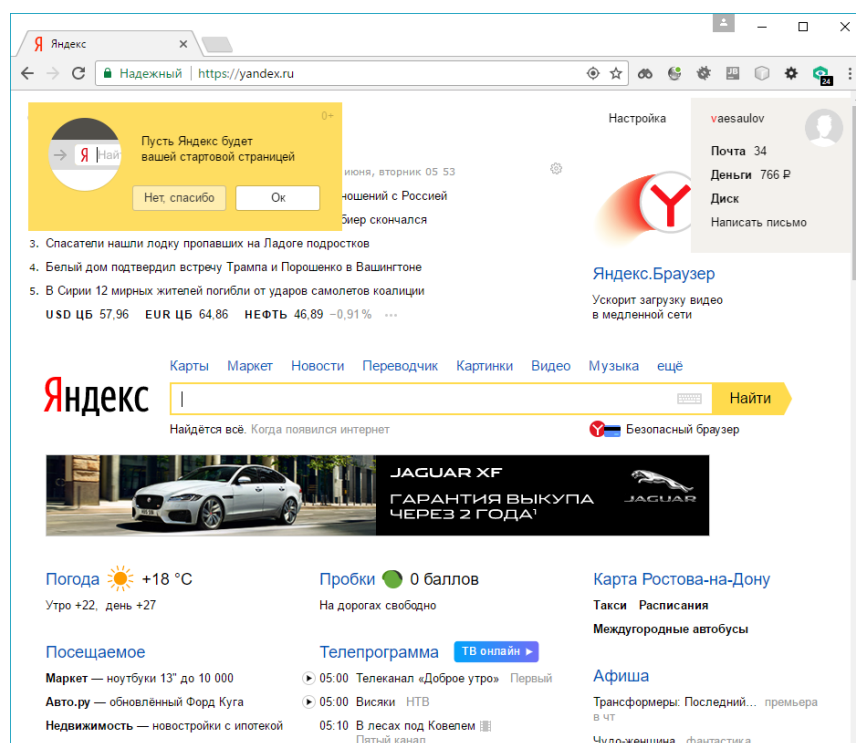


Рисунок 14 – Вид поисковой системы Яндекс

ПС Яндекс с самого имеет следующие особенности:

- проверка контента на уникальность, при этом клонированные тексты отвергаются или ранжируются крайне низко;
- в поиске учитывается морфология с дальнейшим исправлением ошибок;
- поиск ведется с учетом присутствия ключевых фраз в объеме текста;
- ведется учет релевантности страниц;
- учитывал, помимо количества запросов определенного слова; частоту его употребления (насыщенность), положение в тексте (разбросанность), расстояние между словами;
- реализовал функцию вопрос-ответ.

В структуру поисковика входят три модуля:

- crawler, краулер или робот (он же спайдер (spyder));
- база данных;
- клиентская часть.

Поисковая система Яндекса использует алгоритм ранжирования, который позволяет конечному пользователю выполнять лучший поиск по страницам, которые лучше всего соответствуют его поиску.

Алгоритм ранжирования оценивает факторы, основываясь на их совокупности и их анализе, поисковая система определяет полезность страницы и ее рейтинг.

Около 250 факторов в настоящее время участвуют в рейтинге, в том числе следующие факторы:

- присутствие слов из запроса в контексте страницы;
- присутствие ключевых слов в заголовках и на метауровне документа;
- наличие ссылок на ранжируемый документ;
- наличие комментариев к ссылкам и изображениям;

– авторитетность ресурса.

Другой популярный ПС, которая в настоящее время используется, является Rambler. Он поддерживает все кириллические кодировки, предлагает полнотекстовый поиск на более чем 350 тысяч страниц с 3900 отечественных узлов, а количество проиндексированных страниц в день увеличивается в среднем на десять тысяч. При углубленном запросе отображается возможность получения дополнительной информации. Вы можете уточнить порядок поиска до даты последнего обновления документа и указать условия, которые следует исключить из источника (фактически это дополнительный оператор NOT). Для удобства пользователя, все манипуляции с терминами могут быть выполнены через меню. Как и у Mail.ru или Яндекса, у Rambler есть и другие проекты. Среди проектов можно выделить такие как Rambler/класс, который помогает школьникам получать ответы на их вопросы, и Rambler/лайки, который помогает отслеживать статистику переходов на нужную страницу. Вид ПС Rambler представлен на рисунке 15.

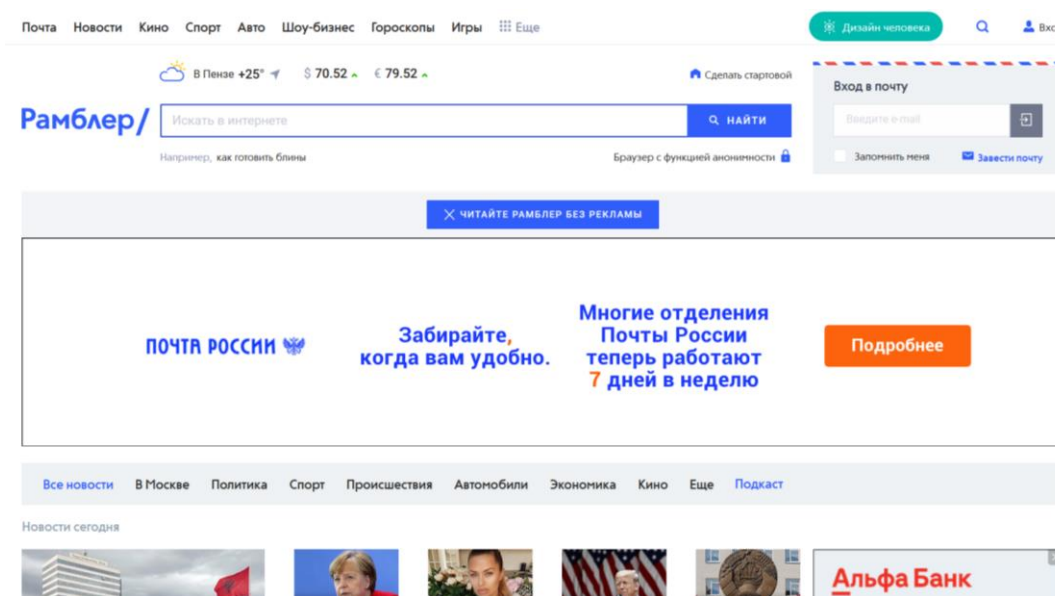


Рисунок 15 – Вид поисковой системы Rambler

Одним из главных преимуществ Рамблера является близкий к образцовому вывод результатов поиска. Помимо имени, адреса электронной почты, кода, размера и времени обновления документа, ссылка на найденный

объект также содержит впечатляющие размеры резюме, которые можно использовать, чтобы получить представление о контексте, в котором искомые термины используются (они выделены жирным шрифтом).

Недостатком этой ПС является преждевременное удаление ссылок на сайты, которые по какой-то причине были удалены с серверов. Для некоторых запросов количество ссылок, за которыми следует сообщение «Файл не найден», составляет до 10%.

В данной главе описаны веб-технологии поиска и обработка семантической информации. Данные технологии помогают точнее понять системе, что именно имел в виду пользователь, а значит система может выдать более точный результат.

В целом крайне важно правильно сформировать знания в компьютерных системах. Благодаря правильной организации данных, работа с ними станет проще и точнее, а значит конечному пользователю будет гораздо удобнее пользоваться такими системами.

3 Разработка программной реализации

3.1 Выбор языка и технологии программирования

В качестве языка разработки программного обеспечения использовалась библиотека Qt.

Библиотека Qt представляет собой кросс-платформенный фреймворк прикладного уровня. Он широко используется для разработки программного обеспечения различного назначения. Одной из сильных сторон данной библиотеки является наличие богатого набора графических компонент, поддерживающих разработку графических интерфейсов. Языком реализации Qt является язык C++. Это позволяет разработчику взаимодействовать с другими мощными расширениями этого языка. Помимо C++, библиотека Qt предоставляет для других языков программирования. В качестве примеров можно привести такие языки, как Python (PyQt), Ruby (Korundum/QtRuby) и Perl (PerlQt) [28].

Модульная архитектура библиотеки Qt приведена на рисунке 16.

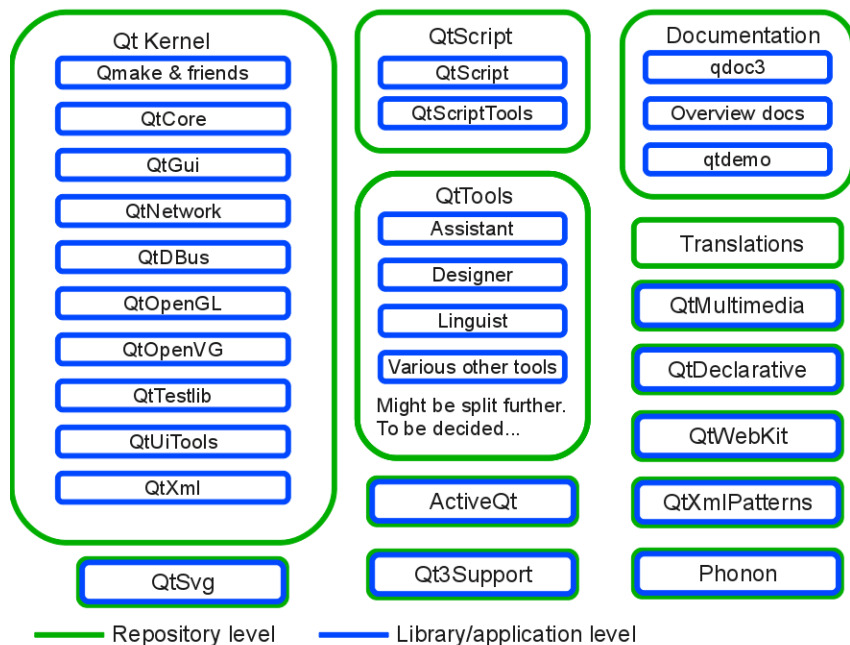


Рисунок 16 – Модульная архитектура библиотеки Qt

Рассмотрим некоторые компоненты архитектуры Qt. QtCore представляет собой функционал ядра библиотеки. Его классы могут

использоваться другими модулями и сущностями. QtGui предоставляет функции компонент графического интерфейса. Подсистема QtNetwork поддерживает классы, необходимые для написания сетевых программ. Так, поддерживаются протоколы HTTP, FTP, TCP/IP и UDP. Подсистема QtSvg позволяет работать с данными графики Scalable Vector Graphics (SVG). Модуль QtXml дает возможность работы с XML [29].

Обобщенная иерархия классов библиотеки Qt приведена на рисунке 17.

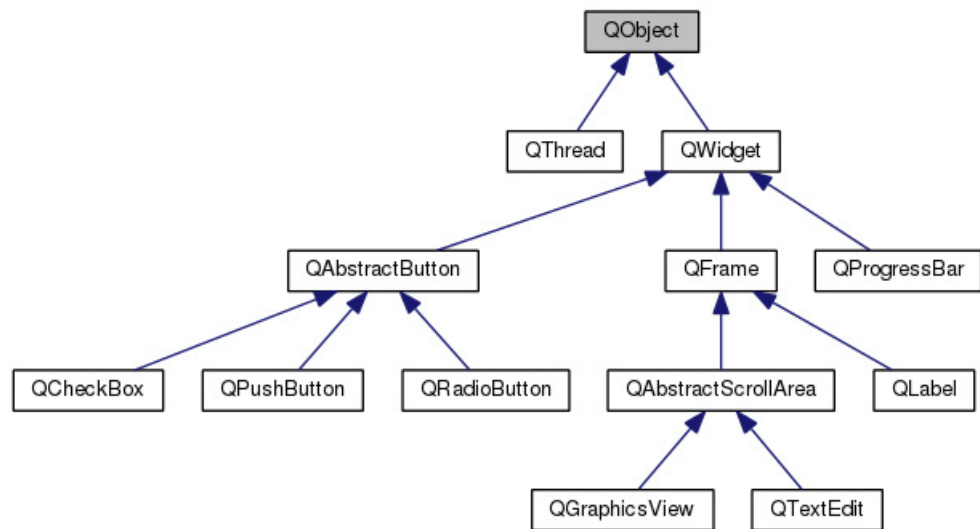


Рисунок 17 – Обобщенная иерархия классов библиотеки Qt

Все объекты Qt представляют собой гибкие индивидуальные сущности. Наследование в иерархии начинается от объекта QObject. При работе с объектами используются ссылки. Поэтому понятие объекта и указателя на объект, наследуемого от QObject, синонимичны.

При реализации графических пользовательских интерфейсов взаимодействие объектов основано на механизме событий. При этом часто используются т.н. обратные вызовы. Это означает, что передача кода для последующего выполнения происходит с использованием таких объектов, как указатели на функции, функторы. В условиях событийной реализации широко поддерживается концепция обработчиков. Ее суть состоит в том, что обработчик может перехватить события определенного объекта.

Центральной концепцией Qt, описывающей взаимодействие объектов, является концепция сигналов и слотов. Архитектура их взаимодействия

приведена на рисунке 18.

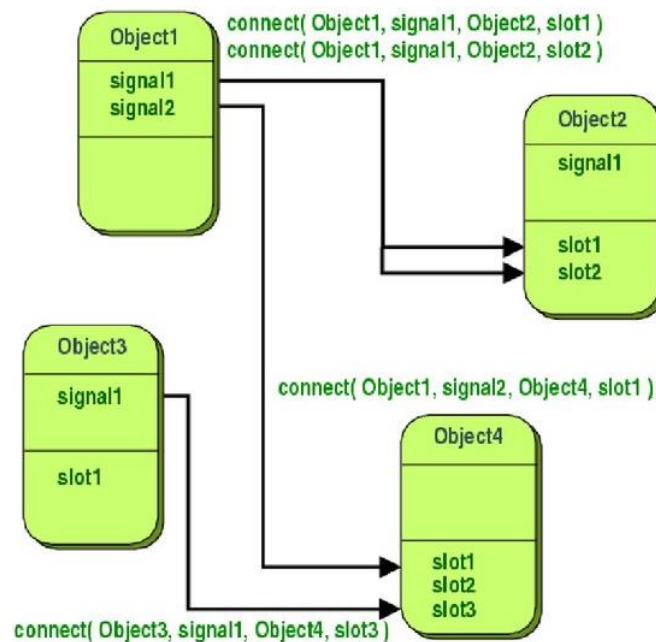


Рисунок 18 – Архитектура взаимодействия объектов на основе слотов и сигналов

Сигнал возникает в случае, когда вызван отвечающий ему метод. Слот является методом, исполняемым при получении сигнала. Слоты могут иметь разные уровни инкапсуляции. При этом уровень защиты ограничивает использование слотов в качестве обычных методов. Однако это не влияет на возможность подключения сигналов к слотам.

Основное отличие слот-сигнальной модели от событийной с использованием обработчиков состоит в следующем. В отличие от событийной модели, число соединений слота с сигналами не ограничено. Аналогично, сигнал может подключаться к произвольному количеству слотов. В случае отправки сигнала выполняется вызов всех отвечающих ему слотов.

В качестве языка программирования, использующего библиотеку Qt, был выбран язык Python. Его отличают высокая простота в применении, переносимость и открытость. Он ориентирован на высокую производительность разработчика и максимальную читаемость кода. Не предлагая революционных особенностей и нововведений, язык Python комбинирует основные лучшие качества многих различных языков

программирования [30, 31].

Основным направлением Python является нацеленность на быструю разработку приложений. Создание программ достигается использованием встроенных высокоуровневых структур данных, динамической типизацией и простым синтаксисом. Все это существенно снижает временные сроки программирования. Программирование на языке Python значительно увеличивает производительность труда разработчика по сравнению с традиционными языками программирования. К таким языкам можно отнести C, C++ и Java.

Стандартная библиотека Python предоставляет обширный функционал, который используется в прикладных программах. Она включает модули для работы с текстом, мультимедийными форматами, архивами и т.п. Кроме этого, на ее уровне реализована богатая функциональность для работы по сети, обработки данных в Интернет. Также обеспечивается поддержка юнит-тестирования с целью повышения качества кода.

Функциональные возможности разрабатываемых программных продуктов расширяются за счет интеграции отдельных компонентов. Ядро Python имеет широкие возможности встраивания в разнообразные приложения и системы. С другой стороны, есть возможность писать модули для Python на других языках (extending). Стандартная библиотека позволяет программам Python напрямую обращаться к динамическим библиотекам (DLL, Dynamic-link library), написанным на C. Существуют модули, позволяющие встраивать код на C/C++ прямо в исходные файлы Python, программный интерфейс для написания собственных модулей на других языках. Такие модули расширения позволяют объединить эффективность кода на C/C++ с удобством и гибкостью интерпретатора Python. На этом пути нивелируются недостатки языка Python, которые связаны, прежде всего, со скоростью выполнения программ, которая не всегда может быть такой же высокой, как у программ, написанных на языках программирования низкого

уровня, таких как C или C++.

Интересной особенностью Python является то, что все данные в языке Python являются объектами. Над ними можно выполнять те или иные действия. С отдельным объектом связывается область памяти со значениями и ассоциированными с ними наборами операций.

В Python существует множество прикладных библиотек для решения самых разнообразных задач (Web-приложения, базы данных, графические библиотеки и т. д.). Среди них для нас наибольший интерес вызывают библиотеки численных методов. Их функциональность в последних версиях значительно расширилась, что дает возможность говорить о конкуренции с системами уровня MATLAB [32].

В данной главе для разработки программы был выбран язык программирования Python с модулем Qt.

3.2 Описание основных компонентов приложения

Для реализации задачи семантического анализа Web-данных с использованием языка Python и библиотеки PyQt была реализована программа, выполняющая следующие функции:

- задание ссылки на подлежащий анализу ресурс;
- поддержка семантического анализа текстовой информации;
- поддержка лемматизации текстовой информации, а также удаление стоп-слов;
- визуализация текстовой информации в виде семантического облака;
- выделение ключевых слов в тексте на базе методов скрытого семантического анализа;
- вывод результатов.

Для реализации лемматизации текста и контроля стоп-слов

использовалась библиотека инструментов обработке естественно-языковой информации `ntlk` в составе языка Python. Она также широко используется при векторно-числовом представлении текстовых данных и содержит большое число предобученных словарей, применяя которые, можно говорить о семантическом сходстве текстовой информации.

Для реализации выполнения обработки текстовых данных применялась библиотека `sklearn` языка Python, которая обладает большим числом алгоритмов машинного обучения. В основе модели выделения ключевых слов в тексте лежали алгоритмы LDA и NMF, основанные на выполнении сингулярного разложения матрицы связей вида «текст - слово» или «предложение-текст».

Список основных модулей программы приведен в таблице 1.

Таблица 1 – Список модулей программы

Имя модуля	Описание
<code>main_app.py</code>	Модуль головной программы
<code>worldcloud_form.py</code>	Модуль формы настроек построения семантического облака
<code>textmodel_form.py</code>	Модуль формы настроек выделения ключевых слов из текста
<code>CommonAlgorithms</code>	Модуль алгоритмов обработки текста
<code>TextFuncs.py</code>	Модуль сервисных функций для работы с текстом

Список основных функций модуля `TextFuncs` приведен в таблице 2.

Таблица 2 – Список основных функций модуля `TextFuncs`

Имя функции	Описание
<code>ReadStopWords()</code>	Формирование стоп-слов
<code>NormalizeTokens(tokens)</code>	Нормализация токенов
<code>DeleteStopWords(tokens, stopwords=None, min_length=4)</code>	Удаление стоп-слов

AnalyseText(tokens,options)	Извлечение слов при анализе текста
GetWords(list_res,opts)	Извлечение слов из списка строк
GetText(link,r_)	Извлечение текста из веб-ссылки

Список основных классов приложения приведен в таблице 3.

Таблица 3 – Список основных классов приложения

Имя класса	Описание
TextModel	Класс алгоритма выделения ключевых слов
CloudOfWords	Класс построения семантического облака
Options	Класс опций настройки алгоритма выделения слов при построении семантического облака
WordCloudForm	Форма настроек построения семантического облака
TextModelForm	Форма настроек выделения ключевых слов

Из описания функциональности таблиц 1-3 видно, что функции отвечают решаемой задаче.

Программный код всех модулей приведен в приложении А.

В данной главе были описаны основные компоненты приложения и способы их реализации.

3.3 Порядок запуска и работы приложения

Разработанная программа имеет следующий вид, показанный на рисунке 19.

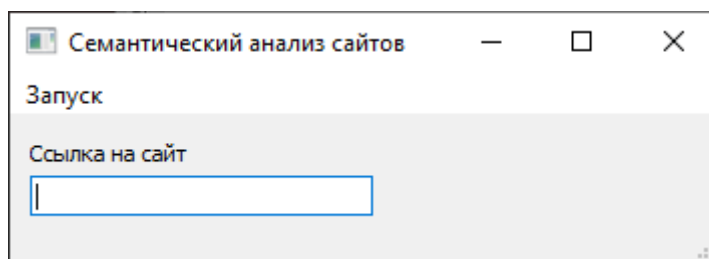


Рисунок 19 – Общий вид программы

В поле «Ссылка на сайт» вводится веб-адрес ресурса.

Для выполнения построения семантического облака необходимо выбрать пункт меню Запуск > Семантическое облако. Появится форма настройки параметров семантического облака (рисунок 20).

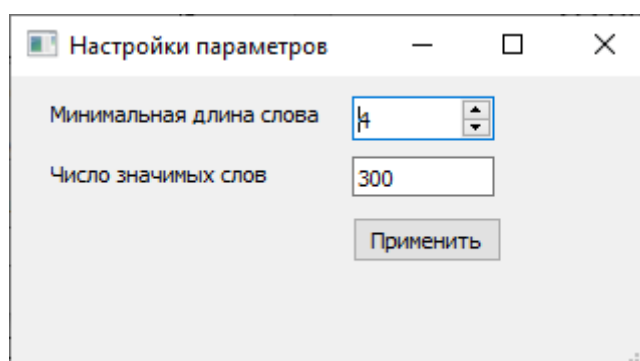


Рисунок 20 – Настройки параметров семантического облака

В поле «Минимальная длина слова» задается длина слова, пригодного для дальнейшей обработки. В поле «Число значимых слов» задается число значимых для построения семантического облака слов. По нажатию на кнопку «Применить» выполняется расчет формирования семантического облака и его визуализация.

Для выделения списка ключевых слов и предложений необходимо выполнить команду. Появится форма (рисунок 21).

В качестве параметров алгоритма извлечения ключевых слов можно задать число ключевых предложений, а также число слов, которые в них входят. Это достигается посредством изменения данных в полях «Число предложений» и «Число слов».

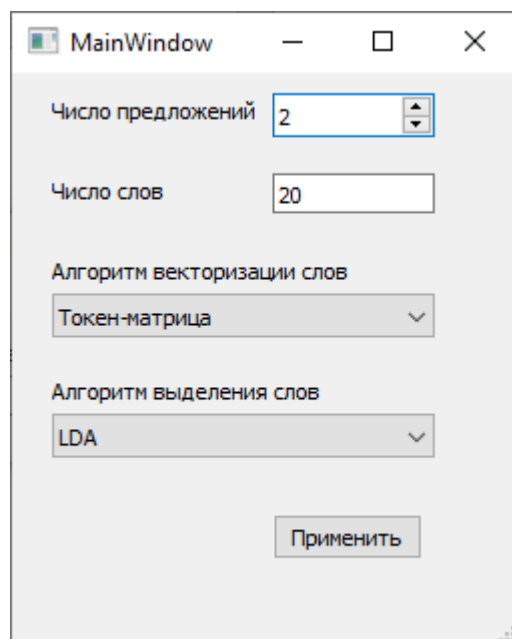


Рисунок 21 – Вид формы настройки параметров выделения ключевых слов

Векторизация слов проводится с применением алгоритмов «токен-матрица» и модели TF-IDF. В первом случае текст рассматривается как двоичная последовательность, где единица отвечает токену, встречающемуся в предложении. Такие признаки могут обладать довольно высокой разреженностью. Модель TF-IDF представляет собой статистическую меру. Она применяется для оценки важности слова в контексте документа. При этом слово может входить как в документ, так и текстовую коллекцию. Вес некоторого слова пропорционален частоте употребления этого слова в документе и обратно пропорционален частоте употребления слова во всех документах коллекции.

При нажатии кнопки «Применить» выполняется поиск ключевых слов. При этом результаты выводятся в консоль.

В качестве примера применения рассмотрим семантический анализ страницы, приведенной на рисунке 22.

При этом использовался алгоритм LDA, а для векторизации применялась модель «токен-матрица».

Листинг 1 – Извлеченные ключевые слова при числе предложений 2, числе слов 20 и алгоритме LDA с моделью «токен-матрица»

Topic #0: компьютерная техника на дону ростове ростов купить
яндекс отзывы ул компьютеров цены магазины оргтехники маркет
низкие сайте адреса читать скидки

Topic #1: unit цвет ugr электрическая мясорубка oldi черныйrnd
ru черный uek электрический 261 чайник 457 458 459 460 бежевый
бежевыйrnd ucj.

При числе предложений, равном 5, при алгоритме векторизации TF-IDF результат примет вид, показанный на листинге 2.

Листинг 2 – Извлеченные ключевые слова при числе предложений 5, числе слов 20 и алгоритме векторизации TF-IDF

Topic #0: техника вычислительная это что вычислений большая
информатика смотрите определение обработки википедия программное
обеспечение является компонентом данных важнейшим процесса
computing англ

Topic #1: средств тельная методов совокупность используемых
вычисли информации те хника приёмов задач решения автоматизации
область технич для математических механизации технических
вычислений

Topic #2: вычислительной техники история развития разбить развитие
поколения можно на век периоды следующие механические википедия до
века xvii читать вычислительные ещё

Topic #3: картинки яндекс ещё смотрите ооо директ картинкияндекс
вычислительная станциядля друга вашего меломана0 делать спасибо
помогаете лучше поколениясовременнаяфото1 относитсяпримерыпервого
поколениямеханический этапактив

Topic #4: для вычислений компонентом важнейшим является данных

обработки процесса души музыка погода эфир коллекции индекс
 бизнеса директ метрика как сателит телефония для computing используемая англ
 меломана 0 друга вашего компании вакансии блог контакты
 приспособлениями известные счётные.

Из приведенных результатов можно сделать следующий вывод. Семантическое облако можно рассматривать как достаточно наглядный способ визуализации семантических зависимостей, что наглядно иллюстрирует рисунок 22.

Состав ключевых слов и предложений на качественном уровне определяется как числом предложений как компонент текстового корпуса, так и настройками алгоритмов векторизации текста и извлечения слов. При относительно малом числе компонент для результатов характерен высокий уровень общности. Вместе с тем, увеличение числа предложений повышает информативность получаемого результата.

В данной главе был рассмотрен запуск программы и работа приложения. Разработанное приложение наглядно демонстрирует семантические зависимости посредством построения семантического облака.

Программа была успешно разработана и проверена, благодаря ее результатам можно ознакомиться с ключевыми словами страницы.

Заключение

В работе был проведен анализ структуры Semantic Web как одной из наиболее актуальных концепций развития технологии World Wide Web. Ее ключевыми особенностями является семантический характер представления информации и процессов, связанных информационным обменом. При этом для полноценного внедрения Semantic Web необходимо привлекать методы, связанные и интеллектуальным анализом данных.

В процессе выполнения работы были получены следующие результаты:

1. произведен анализ работы искусственного интеллекта, всемирной паутины в целом и Semantic Web в частности;
2. выполнено исследование механизмов представления данных как их формализация в виде теоретико-графовых структур;
3. проведена разработка программной реализации приложения, обеспечивающего семантический анализ сайтов на базе библиотеки PyQt.

Корректность и верность работы программы подтверждена продемонстрированным примером, качественно отвечающим наблюдаемым в практической деятельности процессам.

Список использованных источников

1. Джонс, М. Т. Программирование искусственного интеллекта в приложениях / М. Т. Джонс ; Пер. с англ. Осипов А. И. – М. : ДМК Пресс, 2006. – 312 с.
2. Макконнелл, Дж. Основы современных алгоритмов: Учеб. пособие / Дж. Макконнелл. – М. : Техносфера, 2004. – 366 с.
3. Рассел, С. Искусственный интеллект. Современный подход / С. Рассел. – М. : Вильямс, 2007. – 1410 с.
4. Рутковская, Д., Пилиньский, М., Рутковский, Л. Нейронные сети, генетические алгоритмы и нечёткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. – М. : Горячая линия-Телеком, 2006. – 452 с.
5. Емельянов, В. В. Теория и практика эволюционного моделирования / В. В. Емельянов, В. М. Курейчик, В. В. Курейчик. – М. : ФИЗМАТЛИТ, 2003. – 431 с.
6. Барсегян, А. А. Технологии анализа данных: Data Mining, Visual Mining, Text Mining, OLAP / А. А. Барсегян, М. С. Куприянов, В. В. Степаненко, И. И. Холод. – СПб. : БХВ-Петербург, 2007. – 384 с.
7. Барсегян, А. А. Анализ данных и процессов: учеб. пособие / А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. – СПб. : БХВ-Петербург, 2009. – 512 с.
8. Чубукова, И. А. Data Mining : учебное пособие / И. А. Чубукова. – М. : БИНОМ. Лаборатория знаний, 2008. – 382 с.
9. Гушин, А. Н., Радченко, И. А. Экспертные системы: учебное пособие / А. Н. Гушин, И. А. Радченко. – Санкт-Петербург : БГТУ, 2007. – 92 с.
10. Поспелов, Д. А. Моделирование рассуждений. Опыт анализа мыслительных актов / Д. А. Поспелов. – М.: Радио и связь, 1989. –184 с.
11. Гольдин, Д. А., Чесноков, А. М. Экспертные системы и продукционные правила в интеллектуальных комплексах распределения ресурсов автономных систем электроснабжения / Д. А. Гольдин, А. М.

Чесноков // УБС. – 2008. – №20. – С. 95–106.

12. Васильев, О. М., Ветров, Д. П., Кропотов, Д. А. Представление и обнаружение знаний в экспертных системах для задач распознавания образов / О. М. Васильев, Д. П. Ветров, Д. А. Кропотов // Журнал вычислительной математики и математической физики. – 2007. – №8. – С. 1428–1454.

13. Васильев, В.В. Практикум по Web-технологиям. Гриф УМО вузов России / В.В. Васильев. – М. : Форум, 2015. – 275 с.

14. Дронов, В. PHP, MySQL и Dreamweaver MX 2004. Разработка интерактивных Web-сайтов / В. Дронов. – М. : БХВ-Петербург, 2016. – 448 с.

15. Евсеев, Д. А. Web-дизайн в примерах и задачах / Д.А. Евсеев, В.В. Трофимов. – М. : КноРус, 2015. – 272 с.

16. Якобсон, Й. Концепция разработки Web-сайтов. Как успешно разработать Web-сайт с применением мультимедиа-технологий / Й. Якобсон. – М. : НТ Пресс, 2006. – 496 с.

17. Semantic Web Challenge // International Semantic Web Conference [Электронный ресурс]. URL: <http://challenge.semanticweb.org/>.

18. Рогушина, Ю. В., Гришанова, И. Ю. Средства интеллектуализации поиска информационных ресурсов в сети Интернет / Ю. В. Рогушина, И. Ю. Гришанова // Тез. VI Междунар. конф. «Интеллектуальный анализ информации ИАИ-2007», 2007. – С. 322–331.

19. Rogushina, J., Gladun, A. Ontological Approach to Domain Knowledge Representation for Informational Retrieval in Multiagent Systems / J. Rogushina, A. Gladun // International Journal «Information Theories & Applications». – 2006. – №4. – С. 354 – 362.

20. Искусственный интеллект. Междисциплинарный подход / под ред. Д. И. Дубровского и В. А. Лекторского. – Москва : ИИнтелЛ, 2006. – 446 с.

21. Калиниченко, Л. А., Рывкин, В. М. Машины баз данных и знаний / Л. А. Калиниченко, В. М. Рывкин. – М. : Наука, 2011. – 296 с.

22. Кейтер, Дж. Компьютеры – синтезаторы речи / Дж. Кейтер. – М. :

Мир, 2012. – 238 с.

23. Коваль, С. А. Лингвистические проблемы компьютерной морфологии / С.А. Коваль. – Санкт-Петербург : Изд-во С.-Петербургского университета, 2016. – 152 с.

24. Ковальски, Р. Логика в решении проблем / Р. Ковальски. – М. : Наука, 2013. – 280 с.

25. Крейг, Дж. Введение в робототехнику. Механика и управление / Дж. Крейг. – М. : Институт компьютерных исследований, 2013. – 564 с.

26. Курцвейл, Рэй Эволюция разума / Рэй Курцвейл. – М. : Эксмо, 2015. – 352 с.

27. Макаров, И. М., Лохин, В. М., Манько, С. В., Романов, М. П. Искусственный интеллект и интеллектуальные системы управления / И. М. Макаров, В. М. Лохин, С. В. Манько, М. П. Романов. – М. : Наука, 2012. – 336 с.

28. Бланшет, Ж. Саммерфилд, М. QT 4: программирование GUI на C++ / Ж. Бланшет, М. Саммерфилд. – М. : КУДИЦ-Пресс, 2008. – 718 с.

29. Шлее, М. Qt 5.3. Профессиональное программирование на C++ / М. Шлее. – Санкт-Петербург : БХВ-Петербург, 2015. – 928 с.

30. Мэтиз, Э. Изучаем PYTHON. Программирование игр, визуализация данных, веб-приложения / Э. Мэтиз. – СПб. : Питер, 2017. – 496 с.

31. Лутц, М. Программирование на Python, I том / М. Лутц. – СПб. : Символ-плюс, 2015. – 992 с.\

32. Саммерфилд, М. Программирование на Python 3. Подробное руководство / М. Саммерфилд. – СПб. : Символ-плюс, 2015. – 608 с.

Приложение А
КОД ПРОГРАММЫ
(рекомендуемое)

main_app.py

```

import sys
#Библиотеки Qt
from main_ui import *
from PyQt5 import QtCore, QtGui, QtWidgets
from wordcloud_form import *
from textmodel_form import *

#Главная форма
class MyWin(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        QtWidgets.QWidget.__init__(self, parent)
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        # Коннектимся по сигналам
        self.ui.WordCloudItem.triggered.connect(self.WordCloudsParams)
        self.ui.TextModelItem.triggered.connect(self.TextModelParams)

    # Обработчик щелчка пункта семантического облака
    def WordCloudsParams(self):
        form_ = WordCloudForm(self)
        form_.show()

    #Обработчик щелчка пункта ключевых слов
    def TextModelParams(self):
        form_ = TextModelForm(self)
        form_.show()

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    myapp = MyWin()
    myapp.show()
    sys.exit(app.exec_())

```

main_ui.py

```
from PyQt5 import QtCore, QtGui, QtWidgets
```

```
class Ui_MainWindow(object):
```

```
    def setupUi(self, MainWindow):
```

```
        MainWindow.setObjectName("MainWindow")
```

```
        MainWindow.resize(354, 94)
```

```
        self.centralwidget = QtWidgets.QWidget(MainWindow)
```

```
        self.centralwidget.setObjectName("centralwidget")
```

```
        self.link_site = QtWidgets.QLineEdit(self.centralwidget)
```

```
        self.link_site.setGeometry(QtCore.QRect(10, 30, 171, 20))
```

```
        self.link_site.setText("")
```

```
        self.link_site.setObjectName("link_site")
```

```
        self.label = QtWidgets.QLabel(self.centralwidget)
```

```
        self.label.setGeometry(QtCore.QRect(10, 10, 111, 16))
```

```
        self.label.setObjectName("label")
```

```
        MainWindow.setCentralWidget(self.centralwidget)
```

```
        self.menubar = QtWidgets.QMenuBar(MainWindow)
```

```
        self.menubar.setGeometry(QtCore.QRect(0, 0, 354, 21))
```

```
        self.menubar.setObjectName("menubar")
```

```
        self.menu_2 = QtWidgets.QMenu(self.menubar)
```

```
        self.menu_2.setObjectName("menu_2")
```

```
        MainWindow.setMenuBar(self.menubar)
```

```
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
```

```
        self.statusbar.setObjectName("statusbar")
```

```
        MainWindow.setStatusBar(self.statusbar)
```

```
        self.action = QtWidgets.QAction(MainWindow)
```

```
        self.action.setObjectName("action")
```

```
        self.action_2 = QtWidgets.QAction(MainWindow)
```

```
        self.action_2.setObjectName("action_2")
```

```
        self.WordCloudItem = QtWidgets.QAction(MainWindow)
```

```
        self.WordCloudItem.setObjectName("WordCloudItem")
```

```
        self.TextModelItem = QtWidgets.QAction(MainWindow)
```

```
        self.TextModelItem.setObjectName("TextModelItem")
```

```
        self.menu_2.addAction(self.WordCloudItem)
```

```

self.menu_2.addAction(self.TextModelItem)
self.menubar.addAction(self.menu_2.menuAction())

self.retranslateUi(MainWindow)
QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):
    _translate = QtCore.QCoreApplication.translate
    MainWindow.setWindowTitle(_translate("MainWindow", "Семантический анализ сайтов"))
    self.label.setText(_translate("MainWindow", "Ссылка на сайт"))
    self.menu_2.setTitle(_translate("MainWindow", "Запуск"))
    self.action.setText(_translate("MainWindow", "Параметры поиска"))
    self.action_2.setText(_translate("MainWindow", "Запустить анализ"))
    self.WordCloudItem.setText(_translate("MainWindow", "Семантическое облако"))
    self.TextModelItem.setText(_translate("MainWindow", "Ключевые слова"))

```

CommonAlgorithms.py

```

from TextFuncs import *
from wordcloud import WordCloud
import matplotlib.pyplot as plt

#Класс построения облака
class CloudOfWords:
    def __init__(self):
        self.options=[]
        #Число значимых слов
        self.words_count=300
    def Run(self,list_res):
        res=GetWords(list_res,self.options)
        text = "";
        list_words = []
        for item in res:
            text += ' ' + ' '.join(item)
        # Построение семантического облака
        wc = WordCloud(max_words=self.words_count, background_color='black').generate(text)
        fig = plt.figure()

```

```

# Построение облака
plt.imshow(wc, interpolation="bilinear")
plt.axis("off")
plt.show()

#Класс построения списка из топ-предложений с топ-словами в тексте
class TextModel:
    def __init__(self):
        #Модель векторизации текста
        self.ModelVectorizer=[]

        #Алгоритм выделения главных компонент
        self.Algorithm=[]

        #Число компонент
        self.components = 5

        #Число слов
        self.top_words = 20

    def Run(self,list_res):
        n_samples = 2000
        n_features = 1000

        #Обучение модели векторизации на словаре
        vectorizer = self.ModelVectorizer.fit_transform(list_res)
        alg_ = self.Algorithm.fit(vectorizer)

        lst = list()

        feature_names = self.ModelVectorizer.get_feature_names()
        for topic_idx, topic in enumerate(alg_.components_):
            message = "Topic #%d: " % topic_idx
            message += " ".join([feature_names[i]
                                for i in topic.argsort()[::-self.top_words - 1:-1]])

            lst.append(message)

        return lst

```

TextFuncs.py

```

import os
import requests
from operator import attrgetter
from pathlib import Path

```

```

# import pandas as pd
import nltk
from nltk import sent_tokenize, word_tokenize, regexp_tokenize
from nltk.corpus import stopwords
import pymorphy2

#Класс структуры опций для обработки текста
class Options:
    def __init__(self):
        # Пороговая длина текста, не относящегося к стоп-словам
        self.min_length=4
        # Список стоп-слов для опций
        self.stopwords=[]

#Формирование стоп-слов
def ReadStopWords():
    #Файл стоп-слов скачан по ссылке
    # https://raw.githubusercontent.com/stopwords-iso/stopwords-ru/master/stopwords-ru.txt
    path = './stopwords-ru.txt'
    #Кодировка текста
    encoding = 'utf-8'
    #Английские стоп-слова
    stopwords_en = stopwords.words('english')
    #Чтение файла с русскими стоп-словами
    with open(path, encoding=encoding) as f:
        stopwords_ru = f.read().split('\n')
    #Русские стоп-слова
    StopWords_RU=list(stopwords_ru)
    # Английские стоп-слова
    StopWords_EN=list(stopwords_en)
    #Объединение списков слов
    for item in StopWords_RU:
        #Добавление в список русских стоп-слов
        StopWords_EN.append(item)
    #Результат как расширенный список
    result=StopWords_EN
    return result

```

#Нормализация токенов

def NormalizeTokens(tokens):

 #Объект морфологического анализатора

 morph = pymorphy2.MorphAnalyzer()

 res_tokens=list()

 #Обход токенов

 for item in tokens:

 #Морфологический анализ и нормализация слова

 res_=morph.parse(item)[0].normal_form

 #Добавление в результат

 res_tokens.append(res_)

 return res_tokens

def DeleteStopWords(tokens, stopwords=None, min_length=4):

 #Если список стоп-слов не указан

 if not stopwords:

 #Возвращаем текст какой есть

 return tokens

 #Список с результатами

 res_=list()

 #Цикл по токенам

 for item in tokens:

 #Если текущее слово не стоповое и длина больше пороговой

 if item not in stopwords and len(item) >= min_length:

 #Добавляем его в итоговый список

 res_.append(item)

 return res_

#Извлечение слов

def AnalyseText(tokens,options):

 #Получение списка стоп-слов по данным из опций

 stopwords_=options.stopwords

 #Токенизация текста на слова

 tokens=word_tokenize(tokens)

```

#Нормализация слов с учетом морфологии
tokens_ = NormalizeTokens(tokens)

# Удаление стоп-слов с учетом опций
tokens_ = DeleteStopWords(tokens_, stopwords_, options.min_length)

return tokens_

#Извлечение слов
def GetWords(list_res, opts):
    res = list()
    stopwords_ = ReadStopWords()
    # opts = Options()
    # opts.min_length = count
    # opts.stopwords = stopwords_
    # Цикл по корпусу
    for item in list_res:
        res_ = AnalyseText(item, opts)
        res.append(res_)
    return res

#Функционал работы с текстом из веба
from bs4 import BeautifulSoup
import requests as req
from TextFuncs import *
import re

#Регулярные выражения для поиска слов
r_ru = re.compile("[а-яА-Я]+")
r_en = re.compile("[a-zA-Z]+")

#Извлечение текста из веб-ссылки
def GetText(link, r_):
    #Переход по ссылке
    resp = req.get(link)
    #Парсер ответа
    soup = BeautifulSoup(resp.text, 'lxml')
    list_all = list()
    #Теги
    list_tags = ['a', 'p', 'span', 'div', 'b']
    #Поиск по тегам

```

```

for tag in list_tags:
    #Список строк по тегу
    list_=soup.find_all(tag)
    #Цикл по списку
    for item in list_:
        #Добавляем текст
        list_all.append(item.text)
    #Исключение дубликатов
    list_all=set(list_all)
    #Извлечение только слов, принадлежащих языку
    list_all = [w for w in filter(r_.match, list_all)]
    #Результат
    res_list=list_all
    return res_list

```

textmodel_form.py

```

import sys
#Библиотеки Qt
from textmodel_ui import *
from PyQt5 import QtCore, QtGui, QtWidgets
#Импорты алгоритмов
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import NMF, LatentDirichletAllocation
#Импорты логики
from TextFuncs import *
from CommonAlgorithms import *

class TextModelForm(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        QtWidgets.QWidget.__init__(self, parent)
        self.ui = Ui_TextModelParams()
        self.ui.setupUi(self)
        self.owner=self.parent()
        self.ui.AlgcomboBox.addItem(["LDA","NMF"])
        self.ui.VectcomboBox.addItem(["Токен-матрица","TF-IDF"])

```

```

# Подписка на обработчик
self.ui.AcceptButton.clicked.connect(self.Accept)

def Accept(self):
    # Регулярные выражения для поиска слов
    import re
    r_ru = re.compile("[а-яА-Я]+")
    r_en = re.compile("[a-zA-Z]+")
    #Алгоритмическая часть
    n_samples = 2000 #Размер выборки
    n_features = 1000 #Число признаков для алгоритма векторизации и выделения слов
    #Число компонент-предложений
    components_=self.ui.CountBox.value()
    # Число слов
    words_count=int(self.ui.WordsCount.text())
    # Ссылка на сайт
    link_=self.owner.ui.link_site.text()
    # Подготовка текста для анализа
    list_res = GetText(link_, r_ru)
    #Выбор алгоритма выделения слов
    ind_alg=self.ui.AlgcomboBox.currentIndex()
    obj_alg=[]
    #Если выбрана опция алгоритма LDA
    if ind_alg==0:
        obj_alg=LatentDirichletAllocation(n_components=components_, max_iter=5,
                                         learning_method='online',
                                         learning_offset=50.,
                                         random_state=0)
    # Если выбрана опция алгоритма NMF
    if ind_alg==1:
        obj_alg=NMF(n_components=components_, random_state=1,
                   alpha=.1, l1_ratio=.5)

    # Выбор алгоритма векторизации

```

```

obj_vect=[]
ind_vect = self.ui.VectcomboBox.currentIndex()
# Если выбрана опция алгоритма матрица-токен
if ind_vect == 0:
    obj_vect=CountVectorizer(max_df=0.95, min_df=2,
                             max_features=n_features)
# Если выбрана опция алгоритма TF-IDF
if ind_vect == 1:
    obj_vect= TfidfVectorizer(max_df=0.95, min_df=2,
                              max_features=n_features)
obj_text = TextModel()
obj_text.ModelVectorizer = TfidfVectorizer(max_df=0.95, min_df=2,
                                             max_features=n_features)
obj_text.Algorithm = NMF(n_components=components_, random_state=1,
                          alpha=.1, l1_ratio=.5)
obj_text.top_words = 20
obj_text.components = components_
msg = obj_text.Run(list_res)
for item in msg:
    print(item)

k=1

if __name__ == "__main__":
    app = QtWidgets.QApplication(sys.argv)
    myapp = TextModelForm()
    myapp.show()
    sys.exit(app.exec_())

```

wordcloud_form.py

```

import sys
#Библиотеки Qt
from wordcloud_ui import *
from PyQt5 import QtCore, QtGui, QtWidgets
#Импорты логики

```

```

from TextFuncs import *
from CommonAlgorithms import *

class WordCloudForm(QtWidgets.QMainWindow):
    def __init__(self, parent=None):
        QtWidgets.QWidget.__init__(self, parent)
        self.ui = Ui_WordCloudSettings()
        self.ui.setupUi(self)
        self.owner=self.parent()
        #Подписка на обработчик
        self.ui.pushButton.clicked.connect(self.Accept)
    def Accept(self):
        # Регулярные выражения для поиска слов
        import re
        r_ru = re.compile("[а-яА-Я]+")
        r_en = re.compile("[a-zA-Z]+")
        #Минимальная длина слова
        word_len=self.ui.WordLenBox.value()
        #Число значимых слов в облаке
        word_count=int(self.ui.WordsCountEdit.text())
        #Построение семантического облака
        # Ссылка на сайт
        link_ = self.owner.ui.link_site.text()
        # Подготовка текста для анализа
        list_res = GetText(link_, r_ru)
        obj_ = CloudOfWords()
        obj_.words_count = word_count
        stopwords_ = ReadStopWords()
        opts = Options()
        opts.min_length = word_len
        opts.stopwords = stopwords_
        obj_.options = opts
        obj_.Run(list_res)

```

```
if __name__ == "__main__":  
    app = QtWidgets.QApplication(sys.argv)  
    myapp = WordCloudForm()  
    myapp.show()  
    sys.exit(app.exec_())
```