УДК 004.738.52

О.С. ВОЛКОВСКИЙ, Е.Р. КОВЫЛИН
Днепровский национальный университет им. Олеся Гончара

КОМПЬЮТЕРНАЯ СИСТЕМА ИНТЕЛЛЕКТУАЛЬНОГО СЕМАНТИЧЕСКОГО ПОИСКА С ИСПОЛЬЗОВАНИЕМ ГЕНЕРАЦИИ ТЕКСТОВ

Рассмотрена структура интеллектуальной запросно-ответной системы для получения семантически связных ответов, генерируемых на основе научно-технических корпусов документов. Проведен анализ существующих подходов к построению программных моделей текста. Создан алгоритм автоматического построения семантической сети текста на естественном языке, разработанный в рамках задачи реализации интеллектуальной запросно-ответной системы. Описан процесс составления опорных меток для текстов, характеризующих семантические связи для текста. На основе полученного алгоритма была создана прикладная программная система построения графа смысловых связей научно-технического текста для языков славянской группы. Реализованная программная система построения программных семантических моделей текста использована в качестве основы для построения интеллектуальной запросно-ответной системы поиска в текстовом корпусе документов. Проведено тестирование созданной системы на основе составленного тематического текстового корпуса документов. Результатом работы созданной системы является текстовый ответ, семантически связанный с поступившим в систему запросом. Несмотря на отсутствие предварительной семантической разметки текстов в корпусе, полученный тестовый ответ от системы содержит в себе предложения исходного текста, имеющих именно семантическую, а не частотную связь с исходным запросом. Это говорит о адекватности описанного алгоритма и возможностях использования созданной программной модели текста в качестве основы для решения прикладных интеллектуальных задач компьютерной лингвистики и задач семантического поиска текста. Научная ценность созданного подхода заключается в отсутствии необходимости предварительной семантической разметки документа перед его анализом и обработкой программной лингвистической системой для решения интеллектуальных задач анализу и генерирования текстов на естественном языке.

Ключевые слова: семантический поиск, семантическая сеть, генерация текста, запросно-ответная система

О.С. ВОЛКОВСЬКИЙ, Є.Р. КОВИЛІН
Дніпровський національний університет ім. Олеся Гончара

КОМП'ЮТЕРНА СИСТЕМА ІНТЕЛЕКТУАЛЬНОГО СЕМАНТИЧНОГО ПОШУКУ З ВИКОРИСТАННЯМ ГЕНЕРАЦІЇ ТЕКСТІВ

Розглянуто структуру інтелектуальної системи запит-відповідь для отримання семантично зв'язних відповідей, генерованих на основі науково-технічних корпусів документів. Проведено аналіз існуючих підходів до побудови програмних моделей тексту. Створено алгоритм автоматичної побудови семантичної мережі тексту на природній мові, розроблений в рамках завдання реалізації інтелектуальної системи запит-відповідь. Описано процес складання опорних міток для текстів, які характеризують семантичні зв'язки для тексту. На основі отриманого алгоритму була створена прикладна програмна система побудови графа смислових зв'язків науково-технічного тексту для мов слов'янської групи. Реалізована програмна система побудови програмних семантичних моделей тексту використана як основа для побудови інтелектуальної запитання-відповідь системи пошуку в текстовому корпусі документів. Проведено тестування створеної системи на основі складеного тематичного текстового корпусу документів. Результатом роботи створеної системи є текстова відповідь, семантично пов'язана з вхідним в систему запитом. Незважаючи на відсутність попередньої семантичної розмітки текстів в корпусі, отримана тестова відповідь від системи містить у собі речення вхідного тексту, що мають саме семантичний, а не частотний зв'язок з вхідним запитом. Це говорить про адекватність описаного алгоритму і можливості використання створеної програмної моделі тексту в якості основи задля вирішення прикладних інтелектуальних завдань комп'ютерної лінгвістики і завдань семантичного пошуку тексту. Наукова цінність створеного підходу полягає у відсутності необхідності попередньої семантичної розмітки документа перед його аналізом і обробкою програмною лінгвістичною системою для вирішення інтелектуальних завдань аналізу і генерування текстів на природній мові.

Ключові слова: семантичний пошук, семантична мережа, генерація тексту, система запит-відповідь.

O.S. VOLKOVSKY, E.R. KOVYLIN
Oles Honchar Dnipro National University

COMPUTER SYSTEM OF INTELLECTUAL SEMANTIC SEARCH WITH THE TEXT GENERATION USING

The structure of the question-answer intellectual system for obtaining semantically connected answers which generated on the basis of scientific and technical corpus of documents is considered. The analysis of existing approaches to the construction of program models of the text is carried out. An algorithm for the automatic construction of a semantic net of text in a natural language was developed for the task of implementing an intelligent request-response system. The process of compiling reference marks for texts characterizing semantic links for a text is described. On the basis of the obtained algorithm was created an applied software system for constructing a graph of the semantic links of the scientific and technical text for the languages of the Slavic group. The implemented software system for constructing software semantic text models was used as a basis for constructing an intelligent query-response search system in a text-based document body. The testing of the created system based on the compiled thematic text corpus of documents is carried out. The result of the work of the created system is a text response, that semantically related to the request entered into the system. Despite the absence of preliminary semantic markup of the texts in the case, the received test response from the system contains the source code proposals that do not have a frequency connection with the original query. This indicates the adequacy of the described algorithm and the possibilities of using the created software text model as a basis for solving applied intellectual problems of computer linguistics and the tasks of semantic search for text. The scientific value of the created approach is the absence of the need for preliminary semantic markup of the document before its analysis and processing by the software linguistic system for the solution of intellectual problems of analysis and the generation of texts in the natural language.

Keywords: semantic search, semantic network, text generation, request-response system.

Постановка проблемы

Проблема оптимального пути поиска информации является одной из ключевых в области компьютерной науки. Процесс разработки большинства программных продуктов рано или поздно приводит к реализации механизмов добавления, сохранения и получения информации для ее последующей обработки. Главным решением этой задачи становятся разнообразные системы баз данных, прекрасно справляющиеся с этими задачами на программном уровне. Однако, если в функционале системы присутствует необходимость работать непосредственно с пользовательским запросом, зачастую состоящего из нескольких критериев, то обработка полученных результатов, полностью ложится на плечи пользователя. Речь идет о подходе, использованном во многих популярных web-поисковых системах: ответом на полученный запрос является множество ранжированных гипертекстовых документов (web-страниц), предполагающее последующий самостоятельный анализ пользователя каждого документа. Главным минусом такого похода является отсутствие глубокого семантического анализа содержимого документа, из-за чего в полученном массиве документов содержится большое количество несвязанной с запросом пользователя информации, а также множество повторений одинаковой информации, поданной в разных интерпретациях. Решением этих проблем являются семантические поисковые системы, ориентирующиеся во время своей работы на смысловые связи в документах. Однако, создание таких систем подразумевает решение многих научных проблем искусственного интеллекта и компьютерной лингвистики, главной из которых является необходимость ручного описания семантических онтологий между всеми элементами языка и между всеми документами в системе.

Текущий подход поиска информации в множестве документов предлагает решение задачи ранжирования относительно пользовательского запроса. Именно этот принцип (минусы которого описаны выше) положен в основу многих популярных web-поисковых систем. Альтернативным подходом решения поисковой задачи является получение семантически значимого текста относительно пользовательского запроса. Главной проблемой рассмотренных подходов к созданию семантических систем является частичное решение первоначальной поисковой задачи. Говоря о системах, использующих классификацию текстовых объектов и связей между ними, стоит отметить что включение лексико-семантического словаря соответствий требует его предварительного ручного заполнения для рабочей предметной области текстов. Это не только ограничивает использование системы, но и является весьма сложной массовой задачей. Системы без семантического компонента избегают этих проблем, при этом, как видно из функционала [2], решая менее сложные интеллектуальные задачи. В нашем случае, поисковая задача сводится к созданию запросно-ответной системы генерации текстов на основе базы знаний корпуса текста, поскольку в ответ на поступивший запрос пользователя будет создан новый текст – сам ответ. Такой класс задач относится к сложным, поскольку разработка систем семантической генерации текстов требует глубокого семантического анализа структуры документа. На прикладном уровне это подразумевает создание

некоторой программной модели семантической структуры текста. Анализ, проведенный в [3], позволяет утверждать о целесообразности использования семантических сетей в связке с интеллектуальным обработчиком данных, что, однако, влечет за собой ряд фундаментальных для автоматической обработки текстов проблем. Речь идет как о об отсутствии четких алгоритмов к автоматизированному созданию семантической сетей – большинство подходов опираются на массивные базы знаний, создание и обработка которых требует больших ресурсов, так и о сложностях программной обработки естественного языка – системы, обладающей непостоянностью и неоднородностью собственных правил, из-за чего усложняется математическое и алгоритмическое описания его компонент. Особенно это касается группы славянских языков, обладающих свойством высокой флексии, что существенно осложняет их программное структурирование. Поэтому, при построении программной модели системы, в первую очередь, важно учитывать степень ее адаптивности, выраженную в нашем случае как независимость от языковых правил и конструкций.

В этой работе описана и реализована, в виде прикладного программного продукта, система семантического поиска, позволяющая автоматически формировать программную семантическую модель как отдельного документа, так и всего корпуса документов в целом.

Анализ последних исследований и публикаций

Алгоритм создания сематических поисковых систем не имеет четко сформированного подхода к своей реализации. Это повлекло за собой возникновение достаточно большого количества теоретических разработок, касающихся вопроса семантического поиска. Однако, когда речь заходит о создании программных продуктов, многие из подобных разработок наталкиваются на ограничения, накладываемые современными возможностями ЭВМ. Поэтому, для наших целей в первую очередь наиболее интересны алгоритмы, положенные в основы реализованных прикладных программных разработок. Проанализировав множество существующих прикладных программных продуктов возможно выделить два класса систем: работающих на основе лексико-семантического словаря отношений и работающих на основе статистического анализа текстов.

Типичным представителем первого класса является система [1]. Семантический анализ заключается в выявлении взаимосвязей между объектами (персоналиями, организациями, событиями) и классификации отношений между ними, а также отождествлении объектов с заранее заданными семантическими классами. Для этого семантический анализатор в тексте выявляет связи между описанными в нем событиями и предметами. Совокупность имен всех предметов и событий с их взаимосвязями образует семантическую сеть. Показателем наличия некой связи между предметами является описание их участия в одном событии (в любых ролях), а показателем наличия связи между предметом и событием – описание участия предмета в данном событии (в любой роли). Семантическим связям можно задать веса. Элементы семантической сети характеризуются следующими параметрами: SemanticType – семантический класс объекта (date, event (сделка), person и т.д.); RelationName – тип синтактико-семантической связи (аргумент, принадлежность, обстоятельство и т.д.); RelationRole – семантическая роль (субъект, объект, инструмент). Система позволяет получить главные темы текста, связи между темами, тематический реферат (информативные фрагменты текста, в которых упоминалась данная тема).

Примером системы статического анализа текстов является система [2]. В схеме ее работы отсутствует семантический компонент и поиск основан на лексико-статистическом анализе текстов. Для поступившего запроса составляется "информационный портрет" – набор упорядоченных по значимости ключевых слов и словосочетаний, характерных именно для данной выборки текстов. По набору ключевых слов пользователь может самостоятельно определить темы, которые могут быть выданы в ответ на его запрос, и тем самым уточнить нужную ему тематику. Под значимыми ключевыми словами понимается контекст запроса относительно контекста всей базы, то есть происходит отбор слов, характеризующих отличие конкретной выборки от всех остальных данных в базе. Формула вычисления значимости слов представляет собой отношение частот встречаемости данного слова (словосочетания) в конкретной выборке и во всей базе. Составленный информационный портрет может использоваться для ранжирования найденных документов (по наличию ключевых слов с большей значимостью) и для автоматической рубрикации.

Цель исследования

В работе перед нами стоит несколько задач. В первую очередь, необходима разработка адаптивного алгоритма построения семантической сети, требующего минимальный и конечный набор знаний для своей корректной работы. Следующим шагом является разработка методики снятия семантических характеристик сети как с отдельного документа, так и с корпуса текстов для последующей обработки. Последним этапом является генерация текста - формирование семантически связанного ответа относительно поступившего пользовательского запроса.

Изложение основного материала исследования

Первым этапом в процессе разработки семантической поисковой системы становится построение программной модели текста – семантической сети (рис. 1).

Рис. 1. Общая структура автоматического построение программной семантической сети документа.

Первым этапом, который зачастую реализуется в большинстве систем автоматической обработки текстов, является синтаксический анализ. На этом этапе происходит выделение предложений и слов анализируемого текста. Кроме того, происходит сокращение рабочего множества слов благодаря стеммингу и очистки текста от служебных частей речи. Для этого над каждой парой слов происходит отсечение окончаний по алгоритму Портера, после чего рассчитывается расстояние Левенштейна для полученных результатов. Если его значение больше или равно длины наибольшей общей части рассматриваемых слов, то считается, что стемма была найдена и каждое слово заменяется на найденную общую часть. Следующим шагом синтаксического анализа становится определение частей речи стемм для извлечения неинформативных слов (такие как служебные части речи) в рамках упрощения процесса семантического анализа. Для этого в систему была включена размеченная выборка размером в пятнадцать тысяч слов и соответствующим им частям речи, которая служит учебным корпусом для наивного Байесовского классификатора, где классами являются части речи, а соответствующими для класса значениями - две и три последние буквы исходного слова и окончания, полученное по алгоритму Портера. Каждое слово из анализируемого текста проходит классификацию на обученной модели. Если прогноз указывает на неинформативную часть языка - слово удаляется. Заключительным этапом синтаксического анализа становится взвешивания стемм, в результате чего каждой стемме соответствует количество ее повторений в тексте и взвешивание предложений, где под весом предложения подразумевается суммарный вес всех стемм в нем.

Размеченный таким образом текст должен пройти этап частотного анализа, в результате которого текстовым данным будут соответствовать некоторые числовые характеристики. Для этого предлагается составить матрицу, строки которой соответствуют предложением, столбцы - стемам, а значения формируются как число вхождений стемы в предложение. Над полученной таким образом матрицей выполняется операция сингулярного разложения. Поскольку сингулярное разложение является устойчивым, становится возможно убрать значения левой и правой матрицы, соответствующие низким сингулярным значениям, оставив только два наибольших, после чего использовать их как координаты для отображения на двухмерную плоскость. Полученные результаты представлены на рис.2. и на рис.3.

После получения частотной числовой картины текста, необходимо привести полученные данные к некоторой семантической картине, на основе которой возможна генерация семантической сети. Для этого, над полученными данными проводится операция кластеризации по алгоритму *k-means,* где количество точек-кластеров определяется по формуле (1), в которой *count(W)* – общее количество слов, (W_U)– общее количество уникальных стемм.

$$cl(W, W_U) = \frac{count(W)}{count(W_U)} \ . \qquad (1)$$

Значения центроидов кластеров-стемм рассчитываются по формуле (2), где $W_0 \ldots W_{CL}$ – частотные веса стемм:

$$Cst(W_U) = \max(W_0 ... W_{cl}) \ . \qquad (2)$$

Значения центроидов предложений рассчитываются по формуле (3), где *WS* – анализируемое предложение, W_i – вес стеммы в предложении, *SN* – количество стемм в предложении:

$$Cs(W_S) = \max\left(\sum_{i=0}^{SN} W_i\right) \qquad (3)$$

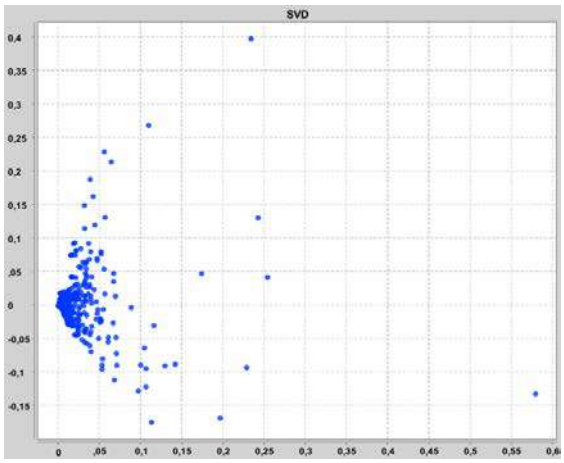

Рис. 2. Проекция частот стемм.

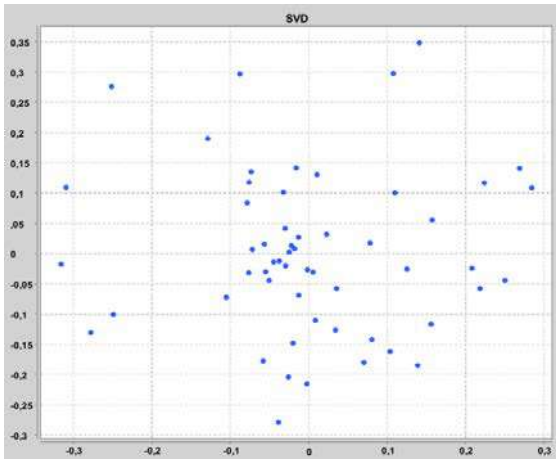

Рис.3. Проекция частот предложений.

Для каждого кластера – стеммы определяется его вес – количество содержащихся в нем точек, на основе которого формируется каркас будущей семантической сети: кластеры-стеммы связываются между собой в порядке увеличения их веса. Результаты кластеризации изображены на рис. 4 и рис. 5.

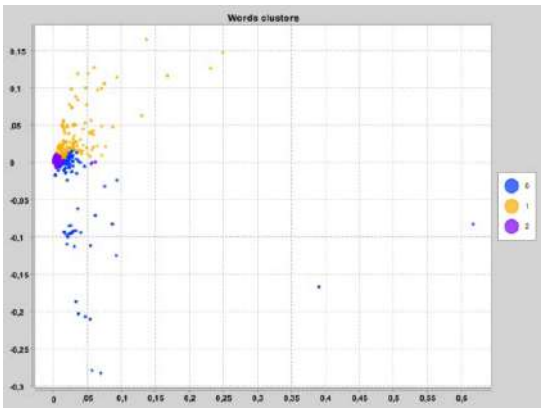

Рис. 4. Кластеры частот стемм

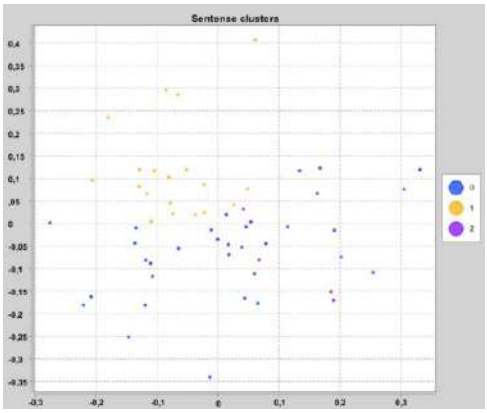

Рис.5. Кластеры частот предложений

Над значениями каждого кластера-стеммы и кластера-предложения выполняется построение контура выпуклой фигуры по алгоритму Джарвиса. Если контур фигуры-предложения пересекается с контуром фигуры-стеммы, то между ними устанавливается связь, семантический вес которой равен количеству точек, содержащихся в площади пересечения. В результате мы получаем структуру семантической сети, изображенную на рис. 6.

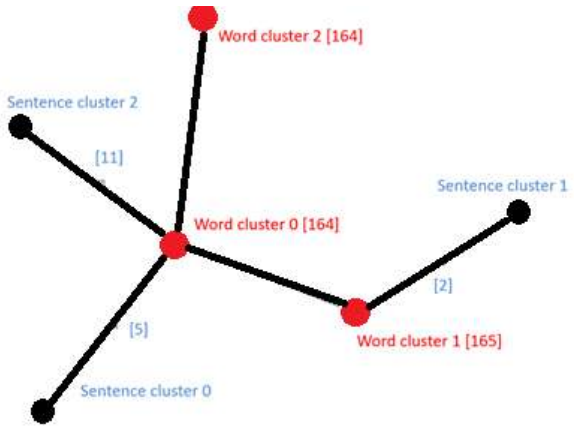

Рис. 6. Семантическая сеть документа.

Множества кластеров-стемм формируют семантические метки документа, по которым, и будет генерироваться результирующий ответ. При поступлении пользовательского поискового запроса в систему, над каждым кластером-сетеммой производится проверка вхождения его элементов в пользовательский запрос. Если для текущего кластера стеммы такое вхождение найдено, то кластер-предложений, связанный с данным кластером стеммой становится кандидатом для включения в результирующий ответ. Если таких связанных кластеров несколько, то в множество кандидатов попадает кластер с максимальным весом связи. Помимо самого текста предложения, в кластере содержатся данные о номере предложения и его весе относительно обрабатываемого документа. Такая операция проводится над каждым документом в корпусе, в результате чего мы получаем множество предложений кандидатов для включения в ответ. Для их нормализации вес каждого кандидата делится на величину количества слов из его исходного текста. В результирующий ответ попадают кандидаты с максимальным нормализированным весом, отсортированные по их изначальному номеру следования в тексте. Если же номера следования предложений совпадают, то сортировка происходит по весу. Размер результирующего ответа определяется как отношение общего количества слов в корпусе к общему количеству предложений. Результат работы системы для поступившего запроса со словом "класс" и корпусом общим размером в сто текстов на такие тематики как программирование, астрономия, экономика, социология. можно увидеть в табл. 1.

Таблица 1

Автоматически сгенерированный ответ семантической поисковой системы на запрос "класс"

При проектировании классов, разработка интерфейса тождественна разработке спецификации (множества методов, который каждый класс, использующий интерфейс должен реализовывать). Интерфейсы, наряду с абстрактными классами и протоколами, устанавливают взаимные обязательства между элементами программной системы, что является фундаментом концепции программирования по контракту (англ. design by contract (DbC). Аппаратно-зависимые подзадачи могут быть строго отделены от других подзадач, что улучшает мобильность создаваемых программ. Модуль — функционально законченный фрагмент программы. Некоторые языки предусматривают объединение модулей в пакеты. Принцип модульности является средством упрощения задачи проектирования ПС и распределения процесса разработки ПС между группами разработчиков. В описании интерфейса определяются имена и сигнатуры входящих в него методов, то есть процедур или функций класса. Использование интерфейсов возможно двумя способами: класс может реализовывать интерфейс. Одним из методов написания модульных программ является объектно-ориентированное программирование. То есть, если класс реализует интерфейс, для любого экземпляра этого класса существуют и могут быть вызваны все описанные в интерфейсе методы. Конкретное яблоко — это в целом некоторое яблоко, вообще яблоко, а любое вообще яблоко — фрукт. Именно поэтому примеры классов в учебных пособиях по объектно-ориентированному программированию так часто упоминают яблоки и грушиВ объектно-ориентированной программе с применением классов каждый объект является "экземпляром" некоторого конкретного класса, и других объектов не предусмотрено. Один класс может реализовать несколько интерфейсов одновременно. Возможно объявление переменных и параметров методов как имеющих тип "интерфейс". Статические поля существуют в одном экземпляре на всю программу (или, в более сложном варианте, — в одном экземпляре на процесс или на поток/нить). Обычные поля создаются по одной копии для каждого конкретного объекта — экземпляра класса. Это позволяет писать обобщённые алгоритмы обработки данных, использующие в качестве типов параметры интерфейсов, и применять их к объектам различных типов, всякий раз получая требуемый результат. Например, интерфейс "Cloneable" может описать абстракцию клонирования (создания точных копий) объектов, специфицировав метод "Clone", который должен выполнять копирование содержимого объекта в другой объект того же типа. Например: "модуль должен быть простым, замкнутым (независимым), обозримым (от 50 до 100 строк), реализующим только одну функцию задачи, имеющим одну входную и одну выходную точку". Первым основные свойства программного модуля более-менее чётко сформулировал Д. Парнас (David Parnas) в 1972 году .Тогда любой класс, объекты которого может понадобиться копировать, должен реализовать интерфейс Cloneable и предоставить метод Clone, а в любом месте программы, где требуется клонирование объектов, для этой цели у объекта вызывается метод Clone. В предельном случае модуль может использоваться даже для заключения в него всего лишь одной процедуры, если необходимо, чтобы выполняемое ею локальное действие было гарантировано независимым от влияния других частей программы при любых изменениях. Впервые специализированная синтаксическая конструкция модуля была предложена Н. Виртом в 1975 г. Сами алгоритмы, то есть действительный программный код, который будет выполнять все эти вычисления, интерфейсом не задаётся, программируется отдельно и называется реализацией интерфейса. Программные интерфейсы, а также классы, могут расширяться путём наследования, которое является одним из важных средств повторного использования готового кода в ООП. Наследованный класс или интерфейс будет содержать в себе всё, что указано для всех его родительских классов (в зависимости от языка программирования и платформы, их может быть от нуля до бесконечности). В них так же, как и в пакетах Ады, явным образом выделяется некоторая "видимая" интерфейсная часть, в которой сконцентрированы описания глобальных типов, констант, переменных, а также приводятся заголовки процедур и функций. В то же время, наследуя класс, мы автоматически наследуем готовый код под интерфейс (это не всегда так, родительский класс может требовать реализации каких-то алгоритмов в дочернем классе в обязательном порядке). Появление объектов в интерфейсной части делает их доступными для других модулей и основной программы. В одном классе просто запрещается реализовывать несколько интерфейсов, имеющих методы с

одинаковыми сигнатурами. Важная особенность модулей заключается в том, что компилятор размещает их программный код в отдельном сегменте памяти. Слова "приватный" и "публичный" в данном случае являются так называемыми "модификаторами доступа". Это хорошо подходит для случаев, когда одноимённые методы разных интерфейсов идентичны по предполагаемой функциональности, но может вызвать нежелательные эффекты, если поведение этих методов должно различаться. Существует основное правило: ничто в одном классе не может видеть приватных элементов другого класса. Наследование по типу private делает все члены родительского класса (в том числе public и protected) private-членами класса-наследника (C++); protected (защищённый, внутренний член иерархии классов) — обращения к члену допускаются из методов того класса, в котором этот член определён, а также из любых методов его классов-наследников

В результате, в финальную выборку попали три текста на тему программирования. Наиболее интересным свойством полученного результата является то, что в ответе присутствуют предложения, которые не содержат в себе слово класс или его производных, однако, при этом семантически связаны с понятием класса и объектно-ориентированного программирования. Это и информация о модульности программ, пример с яблоками и грушами, описывающий понятие класса и понятия интерфейса, Частотной связи этих предложений с запросом нет, и без соответствующего семантического контекста довольно сложно понять, в рамках какой концепции используются эти понятия. Однако разработанный подход позволяет формировать семантически связанные с запросом ответы с минимальными затратами на формирование входящих данных в систему, без необходимости предварительной разметки документов в корпусе, без привязки системы к некоторому предметному корпусу текстов. Помимо запросно-ответных поисковых систем, созданная модель была успешно применена так же и для выявления семантически слабых предложений в тексте промышленной инструкции [4], и автоматическому определению связности текста [5].

Выводы

Была сформулирована и реализована модель построения семантической сети текстового документа, не требующая предварительного заполнения эталонной базы знаний и независимая от отрасли своего применения. На основе данных, полученных из семантической сети, разработан подход к генерации текстового ответа в поисковой запросно-ответной семантической системе и реализована соответствующая прикладная программная система на языке Java. Полученные в процессе тестирования результаты показывают, что система формирует корректный ответ на поступивший запрос, включая в него предложения, не только частотно связанные с запросом, а и предложения имеющие исключительно семантическую или контекстную связь, что подтверждает семантические свойства разработанной прикладной системы.

Список использованной литературы

1. Использование семантических категорий в задаче классификации отзывов о книгах: Материалы ежегодной международной конференции "Диалог" (г. Москва, 29 мая – 2 июня 2013 г.) – М. : Институт лингвистики РГГУ, 2013. – 193 с.
2. Галактика Zoom. Оценка модификации метода формирования инфопортрета: Материалы третьего российского семинара по оценке методов информационного поиска (г. Ярославль, 6 октября 2005 г.). – СПб. : НИИ Химии СПбГУ, 2005. – 226 с.
3. Волковский О.С. Анализ современных подходов к задаче автоматической генерации текстов на естественном языке / О.С. Волковский, Е.Р. Ковылин // Системные технологии. – 2016. – №5 (106). – С. 3-12.
4. Волковский О.С. Компьютерная система автоматичного определения связности текста / О.С. Волковский, Е.Р. Ковылин // Системные технологии. – 2017. – № 5 (112). – С. 11-17.
5. Волковский О.С. Компьютерная система автоматического анализа промышленных инструкций / О.С. Волковский, Е.Р. Ковылин // Системные технологии. – 2018. – № 3 (116). – С. 28-37.