|  |  |  |
| --- | --- | --- |
| МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  РОССИЙСКОЙ ФЕДЕРАЦИИ | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего профессионального образования  «Пермский государственный национальный исследовательский университет» | | |
|  | Кафедра математического обеспечения  вычислительных систем | |
|  | | |
| ОТЧЕТ  по индивидуальному заданию по дисциплине  «Системы искусственного интеллекта» | | |
| Работу выполнил  студент группы ПМИ-2-2022  Колесников Александр Сергеевич  «\_\_\_» \_\_\_\_\_\_\_\_2022 г. | | Проверил  проф. кафедры МОВС  Чуприна Светлана Игоревна  «\_\_\_» \_\_\_\_\_\_\_\_2022 г. |
|  | | |
| Пермь 2022 | | |

Содержание

[1 Описание предметной области 3](#_Toc122718847)

[2 Построение предметной онтологии 19](#_Toc122718848)

[3 Реализация программы 21](#_Toc122718849)

[3.1 Используемые инструменты 21](#_Toc122718850)

[3.2 Оценка семантической близости 22](#_Toc122718851)

[3.3 Интерфейс 22](#_Toc122718852)

[Список использованных источников 26](#_Toc122718853)

1. Описание предметной области

**Тема:** «Конвертация UML-диаграмм из графического представления в xmi-формат».

**Ключевые слова:** UML-диаграммы, UML, xmi, распознавание, конвертация, xmi-формат, спецификация OMG, С++, машинное обучение, распознавание текстов на естественном языке, OCR.

**Постановка задачи:** разработать приложение, реализующее конвертацию UML-диаграмм [1] из графического представления в форматах .png/.jpg в xmi-формат. Работа состоит из двух крупных этапов: распознавание UML-диаграмм (визуальных образов элементов модели и сопровождающих их текстов) и реализация подсистемы преобразования полученных данных в xmi-файл по спецификации OMG [2].

**На входе:** графические файлы моделей на языке UML, которые были подготовлены с помощью инструментов создания UML-диаграмм и экспортированы/сохранены в форматах .png/.jpg.

**На выходе:** xmi-файлы по спецификации OMG, соответствующие графическим файлам моделей.

**Ограничения:** набор тэгов xmi-файлов должен соответствовать набору тэгов, который поддерживается системой UML Checker [3], предназначенной для проверки UML-диаграмм

**Исследовательская часть:**

1) Изучение XML Metadata Interchange Specification Version 2.5.1.

2) Изучение методов распознавания UML-диаграмм.

3) Изучение используемых библиотек/программных реализаций распознавания UML-диаграмм. Критерии выбора – качество распознавания, локализация естественного языка и формат данных результата распознавания.

4) Изучение средств и инструментов создания/генерации xmi-файлов моделей. Важный аспект – представление лексической, синтаксической и семантической информации, а также данных о размещении элементов модели.

5) Изучение принципов обеспечения независимости разрабатываемого приложения от инструмента создания UML-диаграмм (входных данных)

6) Изучение качества созданных xmi-файлов путем их тестирования в системе UML Checker при проверке UML-диаграмм.

**Методы для реализации:**

Разработка ведётся на языке С++ или ином, выбор которого будет обоснован потребностями методов конвертации (С++ - нативный язык UML Checker и относительное обеспечение кроссплатформенности). Выбор вспомогательных библиотек (машинное обучение, распознавание текстов на естественном языке и пр.) студенту предлагается осуществить самостоятельно исходя из результатов исследовательских этапов. В реализации допустимо использовать сторонние модули или фрагменты кода, распространяемые по лицензии свободного программного обеспечения, совместимой с GNU GPL v3.

**Оборудование:** специальные требования к оборудованию не предъявляются.

**Категории понятий:**

* задача,
* инструментальное средство,
* метод.

**Типы связей в модели онтологии:**

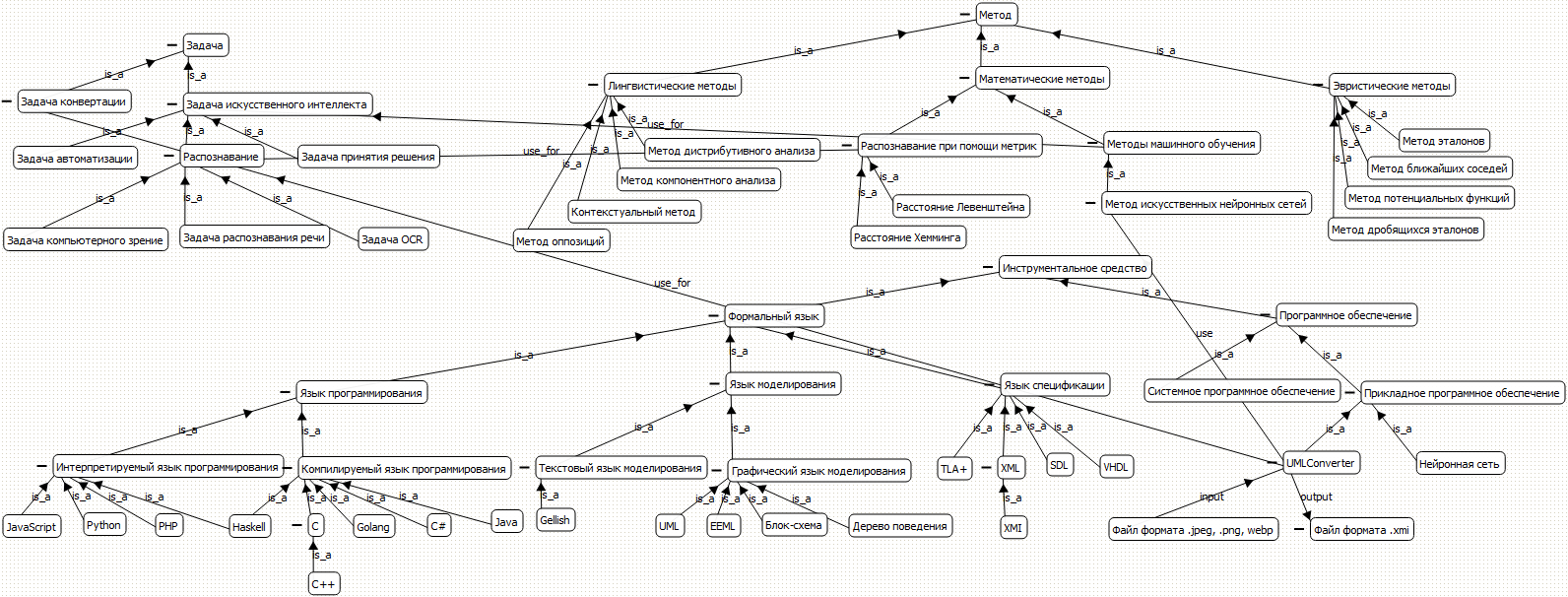
* use – узел с именем программного средства <…> использует/реализует метод/алгоритм/модель, представленный узлом с именем <…> этого метода/алгоритма/модели;
* use\_for – узел с именем программного средства <…> используется для решения задачи с именем <…>;
* is\_a – объект/некоторая сущность (принадлежит категории ПО/задачи/методы/…) является подклассом другой сущности соответствующей категории (родительской сущности);
* a\_part\_of – объект/сущность (принадлежит категории ПО/задачи/методы/…) является составной частью другой сущности соответствующей категории (родительской сущности).

**Глоссарий:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Понятие** | **Категория** | **Определение** | **Комментарий** |
| UML | Инструментальное средство | Графический язык моделирования, представляющий собой систему обозначений, базирующуюся на диаграммах, которые являются графическим инструментом представление данных, и предназначенный для визуализации, спецификации и документирования систем, в которых большая роль принадлежит программному обеспечению. |  |
| Графический язык моделирования | Инструментальное средство | Язык моделирования, который использует диаграммы с именованными символами и линии, для представления концепции и отношений и различные другие графические нотации для представления ограничений. |  |
| Язык моделирования | Инструментальное средство | Язык, обеспечивающий представление объекта моделирования в виде соответствующей ему системы уравнений. |  |
| XMI | Инструментальное средство | Основанный на языке XML, является стандартом обмена метаданными, которые являются информацией о другой информации. |  |
| XML | Инструментальное средство | Расширяемый язык разметки, описанный спецификацией XML, удобный для создания и обработки документов как программами, так и человеком, с акцентом на использование в Интернете. |  |
| Язык программирования | Инструментальное средство | Формальный язык, предназначенный для записи компьютерных программ. |  |
| Формальный язык | Инструментальное средство | Множество конечных слов над конечным алфавитом. |  |
| Компьютерная программа | Инструментальное средство | Последовательность инструкций, предназначенная для исполнения устройством управления вычислительной машины. |  |
| Программное обеспечение | Инструментальное средство | Совокупность программных и документальных средств для создания и эксплуатации систем обработки данных средствами вычислительной техники. |  |
| Системное программное обеспечение |  | Программное обеспечение, управляющие работой компьютера и выполняющие различные вспомогательные функции. |  |
| Прикладное программное обеспечение |  | Программное обеспечение, предназначенное для решения задач пользователя. |  |
| Распознавание | Задача | Отнесение исходных данных к определённому классу с помощью выделения существенных признаков, характеризующих эти данные, из общей массы данных. |  |
| Задача компьютерного зрение | Задача | Задача распознавания, решив которую машины смогут находить, отслеживать, классифицировать и идентифицировать объекты, извлекая данные из изображений и анализируя полученную информацию. |  |
| Задача распознавания речи | Задача | Задача распознавания, решив которую речь человека возможно трансформировать в текст. |  |
| Задача OCR | Задача | Задача распознавания текста на графических изображениях и его перевода в машиночитаемый текстовый формат. |  |
| Распознавание при помощи метрик | Метод | Метод распознавания текста, использующий разнообразные метрики, например расстояние Хэмминга, Левенштейна. |  |
| Распознавание с применением нейронных сетей | Метод | Метод распознавания текста, использующий нейронные сети в качестве главного инструмента. |  |
| Метод искусственных нейронных сетей | Метод | Метод создания математической модели, созданной по подобию биологических нейронных сетей, составляющих мозг живых существ, используемый для решения задач распознавания, классификации и прогнозирования. |  |
| Эвристические методы | Метод | Методы распознавания, основанные на опыте и интуиции разработчика системы распознавания, т.е. ориентированы на решение конкретного типа задач распознавания и непосредственно привязаны к способу синтеза образов. |  |
| Математические методы | Метод | Методы распознавания, опирающиеся на использование классического математического аппарата: методов линейного программирования, корреляционного анализа, теории статистических решений и т.п. |  |
| Лингвистические методы | Метод | Методы распознавания, опирающиеся на использование аппарата алгебры логики и теории формальных языков, и применяющиеся в тех случаях, когда образ представляет собой некоторую структуру, состоящую из так называемых непроизводных (первичных) элементов и признаков, описывающих связи между ними. |  |
| Задача конвертации | Задача | Задача преобразования данных из одного формата в другой. |  |
| Файл формата .xmi | Данные | Файл, содержащий данные, описанные на языке обмена метаданными XMI, и представляют из себя информация о другой информации. |  |
| Файл формата .jpeg, .png, webp | Данные | Файл, содержащий данные о дискретном (пиксельном, точечном) представлении изображения. |  |
| Язык спецификации | Инструментальное средство | Формальный язык, предназначенный для декларативного описания структуры, связей, свойств данных и способов их преобразований, (в отличие от активных языков) без явного упоминания порядка выполняемых действий и использования конкретных значений данных. |  |
| TLA+ | Инструментальное средство | Язык спецификаций, основанный на теории множеств, логике первого порядка и темпоральной логике. |  |
| C++ | Инструментальное средство | Компилируемый, статически типизированный язык программирования общего назначения, использующий приём, при котором переменная, параметр подпрограммы, возвращаемое значение функции связывается с типом не вовремя исполнения программы, а в момент объявления и тип не может быть изменён позже. |  |
| Компилируемый язык программирования | Инструментальное средство | Язык программирования, исходный код которого преобразуется компилятором в машинный код и записывается в файл с особым заголовком и/или расширением для последующей идентификации этого файла, как исполняемого операционной системой. |  |
| Методы машинного обучения | Метод | Класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение за счёт применения решений множества сходных задач. |  |
| Задача искусственного интеллекта | Задача | Задача, решаемая системами искусственного интеллекта (решает задачи, которые априори считаются прерогативой человека).. |  |
| Аналитическая модель | Метод | Формула, представляющая математические зависимости в конкретной предметной области и показывающая, как результат функционально зависит от исходных данных. |  |
| Метод эталонов | Метод | Эвристический метод распознавания. |  |
| Метод дробящих эталонов | Метод | Эвристический метод распознавания. |  |
| Метод ближайших соседей | Метод | Эвристический метод распознавания. |  |
| Метод потенциальных функций | Метод | Эвристический метод распознавания. |  |
| Метод оппозиций | Метод | Лингвистический метод, использующийся при систематизации исследуемых единиц языка / речи. |  |
| Метод дистрибутивного анализа | Метод | Лингвистический метод исследования языка, основанный на изучении окружения отдельных единиц в тексте. |  |
| Метод компонентного анализа | Метод | Лингвистический метод, позволяющий выявить семантические компоненты, меньшие, чем слово (семы). |  |
| Контекстуальный метод | Метод | Лингвистический метод построения когнитивных, концептуальных моделей и схем. |  |
| Задача принятия решения | Задача | Задача искусственного интеллекта в принятии более разумных и рациональных деловых решений. |  |
| Задача автоматизации | Задача | Задача искусственного интеллекта, заключающаяся в автоматизации труда человека. |  |
| VHDL | Инструментальное средство | Язык спецификации описания аппаратуры интегральных схем. |  |
| SDL | Инструментальное средство | Язык спецификаций с формальной семантикой. |  |
| Интерпретируемый язык программирования | Инструментальное средство | Язык программирования, разработанный для непосредственного выполнения исходного кода без необходимости компилировать программу в инструкции на машинном языке. |  |
| JavaScript | Инструментальное средство | Интерпретируемый мультипарадигменный высокоуровневый язык программирования с динамической слабой типизацией и автоматическим управлением памятью, который является реализацией спецификации ECMAScript. |  |
| Python | Инструментальное средство | Интерпретируемый высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. |  |
| PHP | Инструментальное средство | Интерпретируемый мультипарадигмальный скриптовый язык программирования общего назначения. |  |
| Haskell | Инструментальное средство | Интерпретируемый и компилированный чистый функциональный язык программирования общего назначения со статической строгой типизацией и автоматическим выводом типов. |  |
| Golang | Инструментальное средство | Компилируемый многопоточный язык программирования со статической строгой типизацией и автоматическим выводом типов. |  |
| C | Инструментальное средство | Компилируемый статически типизированный язык программирования общего назначения, разработанный как развитие языка B. |  |
| C# | Инструментальное средство | Компилируемый в байт-код статически типизированный объектно-ориентированный язык программирования общего назначения, разработан как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core. |  |
| Java | Инструментальное средство | Компилируемый в байт-код строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems. |  |
| Текстовый язык моделирования | Инструментальное средство | Язык моделирования, который использует формализованные естественные языки для представления концепции и отношений и для представления ограничений. |  |
| Gellish | Инструментальное средство | Универсальный и расширяемый концептуальный язык моделирования данных с общими приложениями. |  |
| EEML | Инструментальное средство | Графический язык моделирования, используемый для абстрактного представления, описания и определения структуры, процессов, информации и ресурсов идентифицируемого бизнеса, государственного органа или другой крупной организации на нескольких уровнях. |  |
| Блок-схема | Инструментальное средство | Графический язык моделирования, использующий различные графические представления для моделирования процесса или алгоритма. |  |
| Дерево поведения | Инструментальное средство | Формальный графический язык моделирования, используемый в основном в системной и программной инженерии. |  |

1. Построение предметной онтологии

По глоссарию была построена онтология предметной области, которая представлена на рисунке 1.



Рисунок

1 – Предметная онтология

1. Реализация программы
   1. Используемые инструменты

Для разработки программы был выбран язык программирования Python из-за богатой коллекции библиотек. Разработка осуществлялась в текстовом редакторе Visual Studio Code [4].

Графический интерфейс приложения реализовывался при помощи фреймворка PyQt5 [5]. Qt — это набор кроссплатформенных библиотек C++, которые реализуют высокоуровневые API для доступа ко многим аспектам современных настольных и мобильных систем. PyQt5 — это полный набор привязок Python для Qt v5. Он реализован в виде более чем 35 модулей расширения и позволяет использовать Python в качестве языка разработки приложений, альтернативного C++, на всех поддерживаемых платформах, включая iOS и Android.

Для лемматизации текстов документов, была использована библиотека Natasha [6]. Данная библиотека решает базовые задачи НЛП для русского языка: токенизация, сегментация предложений, встраивание слов, морфологическая маркировка, лемматизация, нормализация фраз, синтаксический анализ, NER-маркировка, извлечение фактов.

При работе с онтологией, было решено представить ее в виде графа. Для работы с сетевым представлением использовалась библиотека NetworkX [7]. NetworkX — это пакет Python для создания, управления и изучения структуры, динамики и функций сложных сетей. Возможности данной библиотеки:

* Структуры данных для графиков, орграфов и мультиграфов;
* Множество стандартных графовых алгоритмов;
* Сетевая структура и меры анализа;
* Генераторы для классических графов, случайных графов и синтетических сетей;
* Узлами может быть "что угодно" (например, текст, изображения, XML-записи);
* Ребра могут содержать произвольные данные (например, веса, временные ряды) и др.

Для работы с внутренним представлением онтологий в формате JSON использовался готовый парсер на Python, который был предоставлен преподавателем.

* 1. Оценка семантической близости

Оценка семантической близости осуществлялась при помощи модифицированной Efreq-Rnum метрики:

,

где  - кол-во различных путей в графе онтологии между двумя понятиями; bi\* - количество дуг с весом большим β ∈{1, 2, 5, 10} понятия ci; b∗j – количество дуг с весом большим β ∈{1, 2, 5, 10} понятия cj; μb – среднее число связанных с текущим понятием (мощность понятия); eij – кол-во конкордансов, где встретились оба понятия ci, cj (вес дуги); sij – мера семантической близости между понятиями ci и cj.

Для вычисления итоговой оценки документа значение каждой вычисленная меры семантической близости масштабируется в диапазон [0; 1] по формуле:

,

где 𝑠′ − нормализованная мера семантической близости для пары понятий в документе; S – вектор всех мер семантической близости для документа.

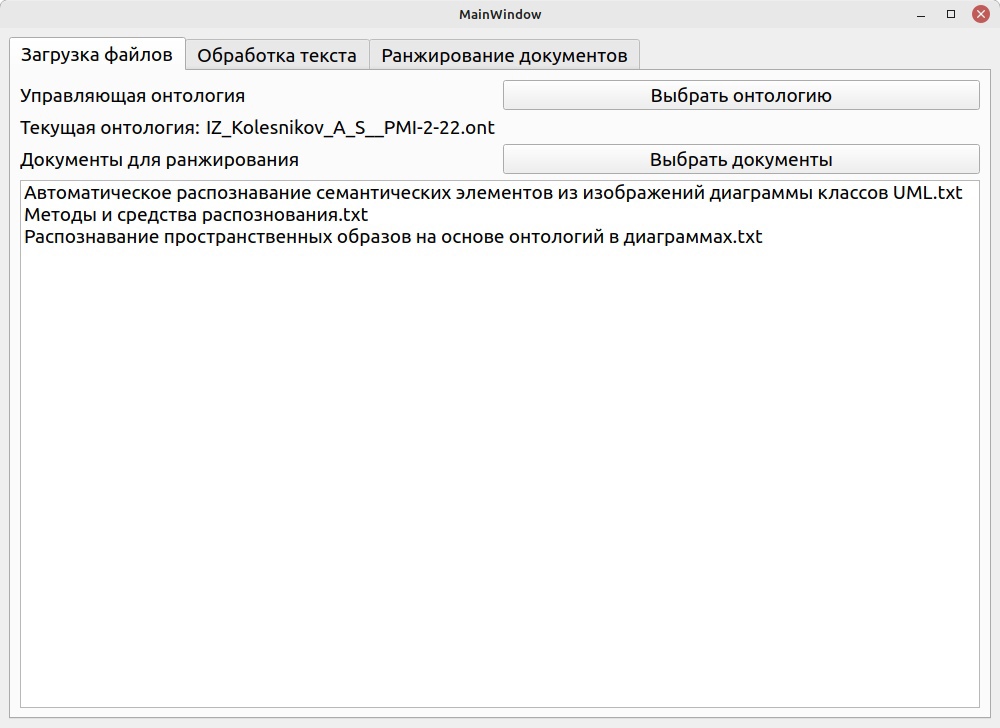
Итоговая оценка соответствия текста документа онтологической базе знаний вычисляется по формуле:

,

где 𝑠′𝑘− нормализованная мера семантической близости между k-й парой понятий в документе.

* 1. Интерфейс

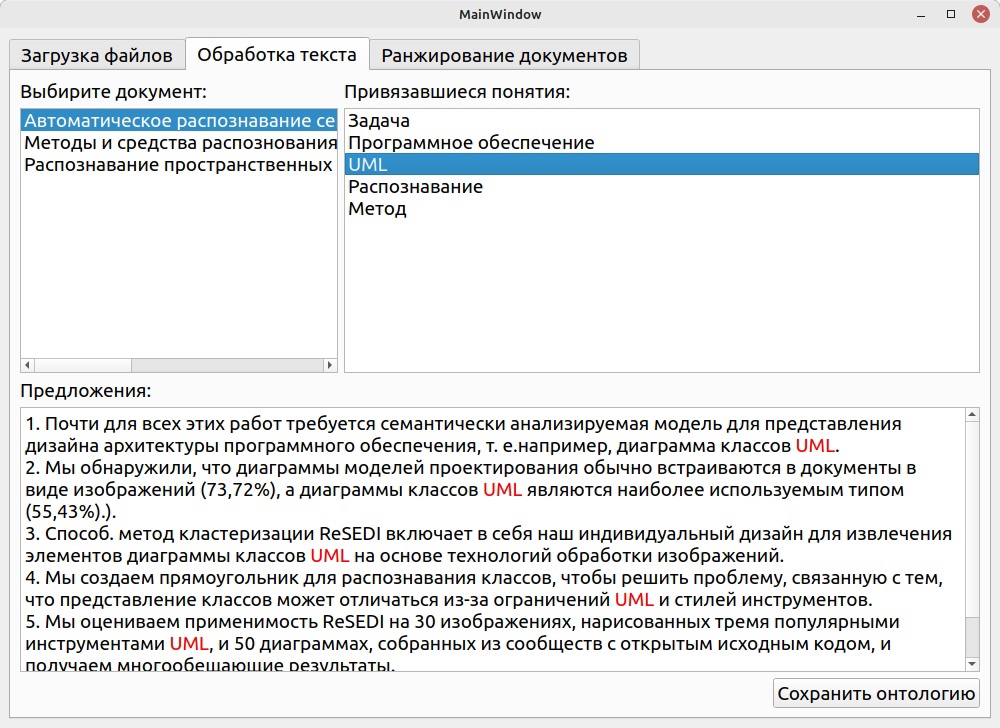
Приложения содержит 4 рабочих окна. Первое (рис. 2) предназначено для загрузки онтологии и исследуемых документов.



Рисунок

– Окно загрузки онтологии и документов

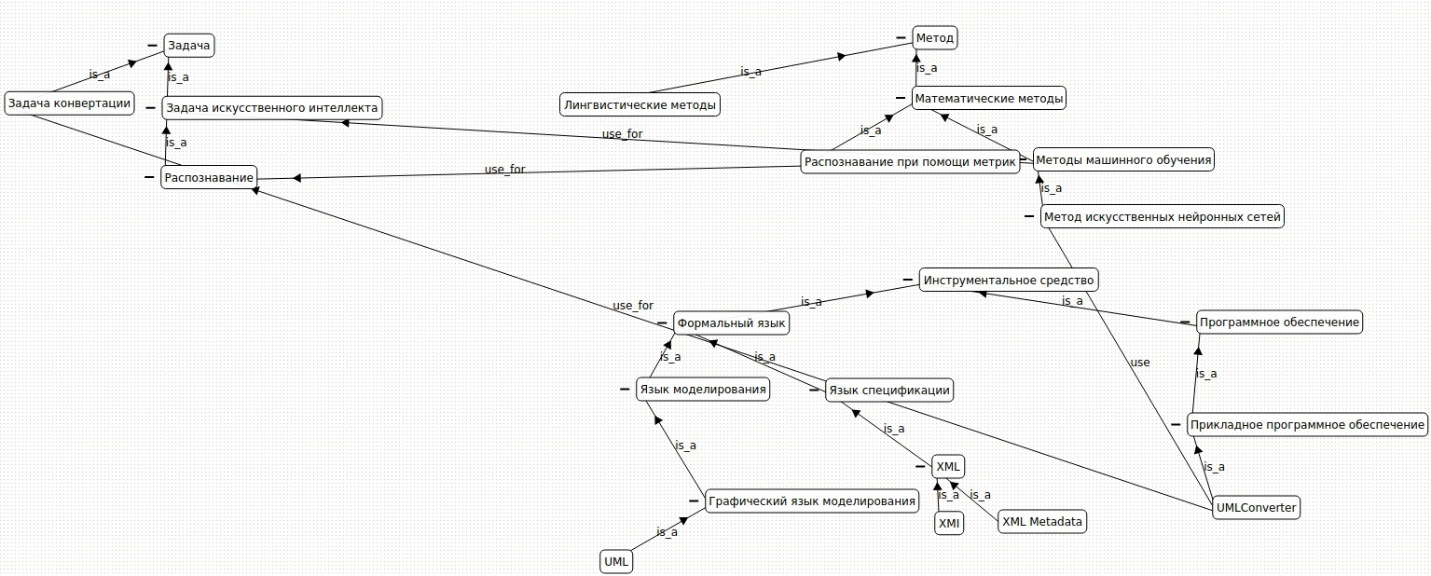
Второе окно (рис. 3) служит для исследования документов и определения какие ключевые понятия содержаться в каждом документе. Также оно позволяет просмотреть список предложений, в которых ключевые понятия встречались.



Рисунок

– Окно отображения ключевых понятий в предложениях

Данное окно предоставляет функциональность по сохранению подграфа исходной онтологий (подонтология) каждого документа. Данная подонтология содержит только привязавшиеся понятия к конкретному документу. Пример такой подонтологии представлен на рисунке 4.



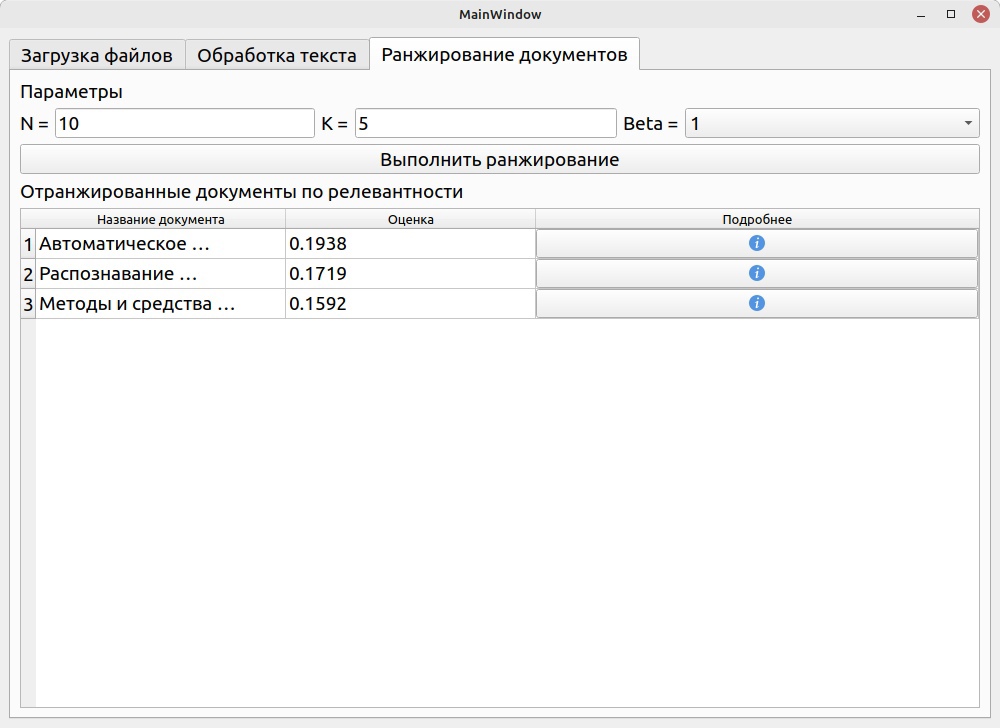
Рисунок

– Онтология привязавшихся понятий документа

Оценка семантической близости осуществляется на третьем окне (рис. 5). Оно позволяет задать такие параметры, как N, K и Beta.

N – это ограничение на длину семантического расстояния между вершинами онтологии при поиске взаимосвязанных вершин (целое число, равное количеству дуг в онтологии между вершинами, представляющими соответствующие взаимосвязанные понятия).

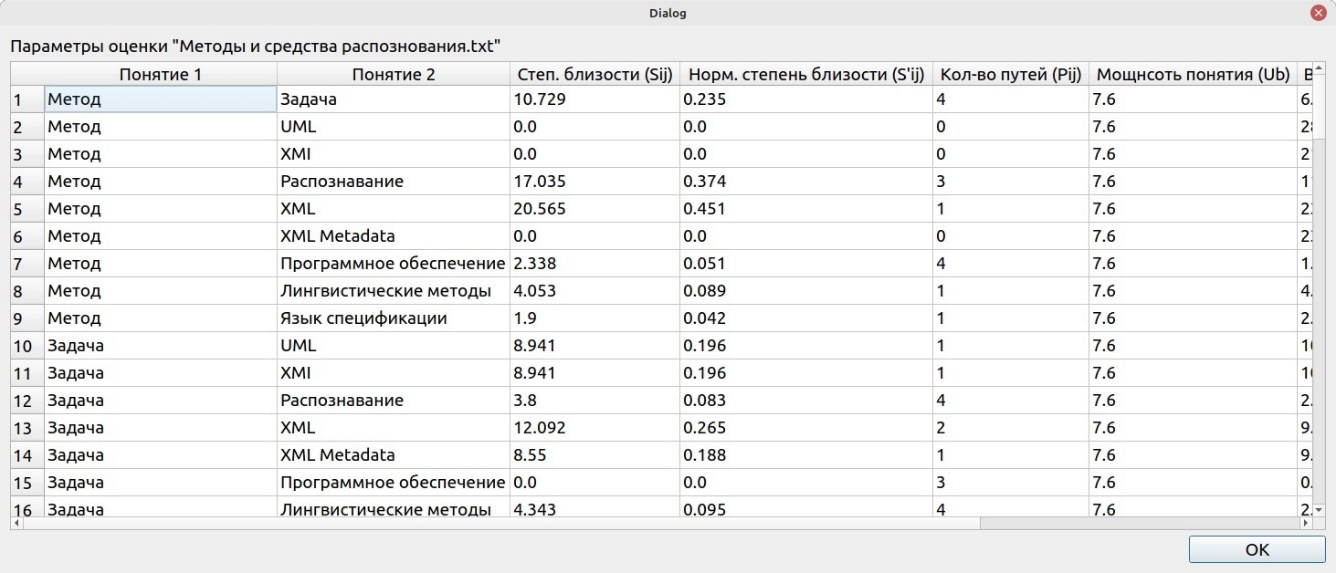
K – допустимое для учета взаимосвязей расстояние между понятиями в исходном тексте (целое число, равное количество предложений между понятиями в исходном тексте). Ранжированные документы представлены в таблице, где указано название документа и его оценка. Также есть возможность просмотреть вычисленные меры.



Рисунок

– Окно ранжирования документов

На рисунке 6 представлено окно, содержащее подробную информацию об оценке конкретного документа. Оно содержит информацию о каждом вычисленной мере, который необходим для подсчета семантической близости.



Рисунок

– Окно с вычисленными параметрами

1. Список использованных источников
2. About the Unified Modeling Language Specification Version 2.5.1. https://www.omg.org/spec/UML/2.5.1/About-UML
3. About the XML Metadata Interchange Specification Version 2.5.1. https://www.omg.org/spec/XMI
4. Gasheva T.S., Vlasov D.I., Otinov A.V., Datsun N.N. Validation Automation of UML Diagrams Created by Students. Trudy ISP RAN/Proc. ISP RAS, vol. 33, issue 4, 2021, pp. 7-18. DOI: 10.15514/ISPRAS-2021-33(4)-1
5. Visual Studio Code - Code Editing. Redefined // URL: https://code.visualstudio.com/ (дата обращения: 23.12.2022).
6. PyQt // Riverbank Computing URL: https://www.riverbankcomputing.com/software/pyqt/ (дата обращения: 23.12.2022).
7. Natasha // GitHub URL: https://github.com/natasha/natasha (дата обращения: 23.12.2022).
8. NetworkX // URL: https://networkx.org/ (дата обращения: 23.12.2022).