Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра интеллектуальных информационных технологий

ОТЧЁТ

по курсу «Естественно-языковой интерфейс интеллектуальных систем»

Лабораторная работа №4

«Семантико-синтаксический анализ текстов естественного языка»

Выполнил:

студент гр. 221701

Телица И.Д.

Проверил:

Крапивин Ю.Б.

Минск, 2025

# Цель работы

Освоить принципы разработки прикладных сервисных программ для решения задачи семантико-синтаксического анализа текста естественного языка.

# Задачи лабораторной работы

**1.** Познакомиться с назначением, структурой и функциональностью, предоставляемой базовым программным обеспечением для решения задачи автоматического семантико-синтаксического анализа текста естественного языка.

**2.** Закрепить навыки программирования при решении задач автоматической обработки текста естественного языка.

# Используемые библиотеки

**1. os:** Стандартная библиотека Python для работы с файловой системой.

**2. json:** Стандартная библиотека Python для работы с JSON-данными.

**3. collections:** Стандартная библиотека Python для специализированных контейнеров (defaultdict, Counter).

**4. python-docx:** Библиотека для чтения DOC/DOCX-файлов.

**5. pdfplumber:** Библиотека для извлечения текста из PDF-файлов.

**6. striprtf:** Библиотека для обработки RTF-файлов.

**7. beautifulsoup4:** Библиотека для парсинга HTML-файлов.

**8. nltk:** Библиотека для обработки естественного языка, используется для токенизации.

**9. pymorphy2:** Библиотека для морфологического анализа русского языка.

**10. natasha:** Библиотека для обработки естественного языка на русском языке, используется для синтаксического анализа.

**11. tkinter:** Стандартная библиотека Python для создания графического интерфейса.

# Интерфейс

Интерфейс включает:

* • **Панель управления:** Кнопки «Загрузить текст», «Сохранить корпус», «Показать статистику», «Показать синтаксис», «Показать семантику», «Экспортировать результаты», «Справка».
* • **Панель поиска:** Поле ввода, радиокнопки «Слово»/«Лемма», кнопка «Искать».
* • **Панель редактирования:** Поля для ввода ID документа, предложения, слова, роли/головы, связи и кнопки для добавления ролей или изменения зависимостей.
* • **Область вывода:** Вкладки «Морфология», «Синтаксис», «Семантика» для отображения результатов анализа.

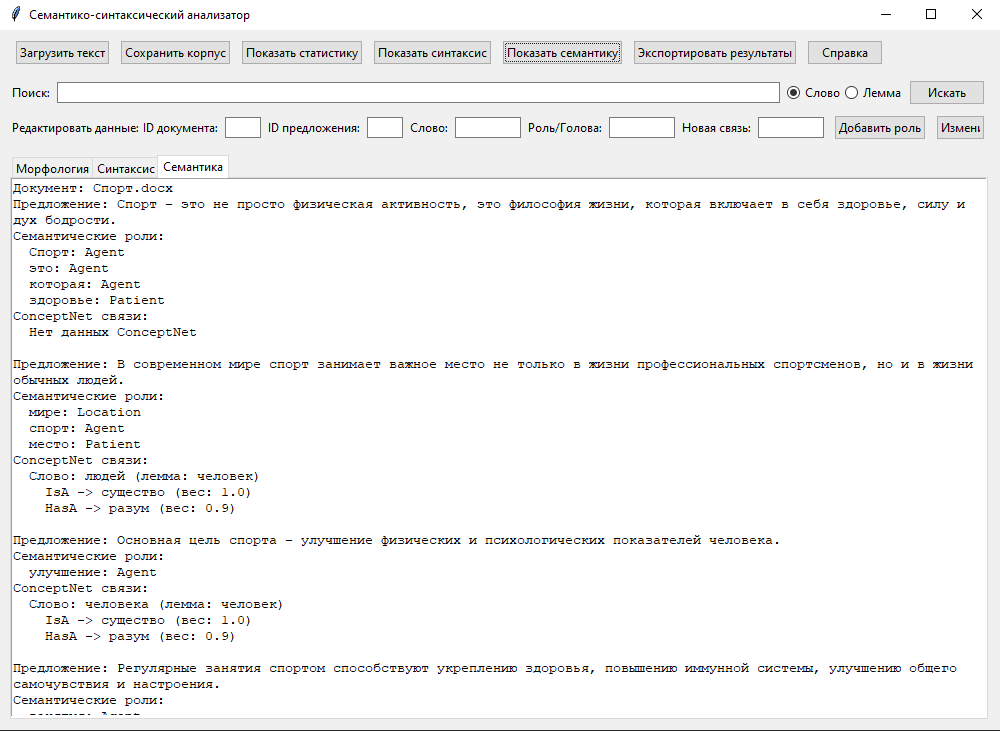


Рисунок 1: Интерфейс приложения

# Структура хранения данных

Данные хранятся в списке словарей (corpus), где каждый словарь представляет документ:

* text: Исходный текст.
* tokens: Список токенов с информацией (слово, лемма, часть речи, теги, семантические связи).
* metadata: Метаданные (название, источник, дата, тип).
* syntax: Список предложений с синтаксическими зависимостями и семантическими ролями.

Индексы:

* word\_index: Индекс слов для поиска по подстроке.
* lemma\_index: Индекс лемм для поиска по нормализованным формам.

Пример структуры:

[

{

"text": "Спорт очень важен для здоровья",

"tokens": [

{

"word": "Спорт",

"lemma": "спорт",

"pos": "NOUN",

"tags": "существительное, неодушевлённое, masc sing, именительный падеж",

"semantics": []

},

{

"word": "очень",

"lemma": "очень",

"pos": "ADVB",

"tags": "наречие",

"semantics": [

{

"relation": "Synonym",

"target": "крайне",

"weight": 1.0

},

{

"relation": "RelatedTo",

"target": "сильно",

"weight": 0.8

}

]

},

{

"word": "важен",

"lemma": "важный",

"pos": "ADJS",

"tags": "прилагательное (краткое), Qual masc, единственное число",

"semantics": []

},

{

"word": "для",

"lemma": "для",

"pos": "PREP",

"tags": "предлог",

"semantics": []

},

{

"word": "здоровья",

"lemma": "здоровье",

"pos": "NOUN",

"tags": "существительное, неодушевлённое, neut sing, родительный падеж",

"semantics": []

}

],

"metadata": {

"title": "1111.txt",

"source": "Загруженный файл",

"date": "2025-05-15",

"type": "unknown"

},

"syntax": [

{

"sentence": "Спорт очень важен для здоровья",

"dependencies": [

[

"Спорт",

"важен",

"именное подлежащее"

],

[

"очень",

"важен",

"наречный модификатор"

],

[

"важен",

"root",

"корень"

],

[

"для",

"здоровья",

"падежная связь"

],

[

"здоровья",

"важен",

"косвенное дополнение"

]

],

"semantic\_roles": [

{

"word": "Спорт",

"role": "Agent"

}

],

"tokens": [

{

"word": "Спорт",

"lemma": "спорт",

"pos": "NOUN",

"tags": "существительное, неодушевлённое, masc sing, именительный падеж",

"semantics": []

},

{

"word": "очень",

"lemma": "очень",

"pos": "ADVB",

"tags": "наречие",

"semantics": [

{

"relation": "Synonym",

"target": "крайне",

"weight": 1.0

},

{

"relation": "RelatedTo",

"target": "сильно",

"weight": 0.8

}

]

},

{

"word": "важен",

"lemma": "важный",

"pos": "ADJS",

"tags": "прилагательное (краткое), Qual masc, единственное число",

"semantics": []

},

{

"word": "для",

"lemma": "для",

"pos": "PREP",

"tags": "предлог",

"semantics": []

},

{

"word": "здоровья",

"lemma": "здоровье",

"pos": "NOUN",

"tags": "существительное, неодушевлённое, neut sing, родительный падеж",

"semantics": []

}

]

}

]

}

]

# Структурная схема приложения

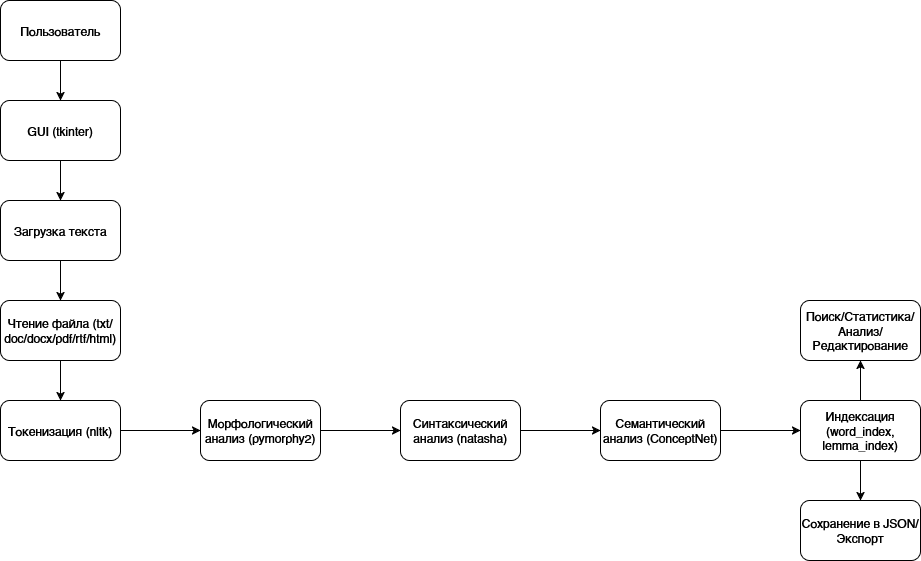


Рисунок 2: Структурная схема приложения

# Описание алгоритмов

## Анализ текста

1. Пользователь загружает файл (TXT, DOC, DOCX, PDF, RTF, HTML).
2. Текст извлекается с помощью соответствующей библиотеки (os, python-docx, pdfplumber, striprtf, beautifulsoup4).
3. Текст токенизируется с помощью nltk.
4. Для каждого токена выполняется морфологический анализ с помощью pymorphy2, учитывающий синтаксический контекст.
5. Производится сегментация текста на предложения с помощью natasha.
6. Выполняется синтаксический анализ (natasha), определяются зависимости между токенами.
7. Определяются семантические роли (Agent, Patient, Instrument и др.) на основе синтаксических связей и морфологии.
8. Для каждого токена извлекаются семантические связи из локального словаря (эмуляция ConceptNet).
9. Токены индексируются (word\_index, lemma\_index).
10. Результаты сохраняются в корпус и отображаются в GUI.

## Редактирование данных

1. Пользователь вводит ID документа, предложения, слово и тип роли или новую зависимость.
2. Данные добавляются или обновляются в структуре корпуса.
3. Обновленные результаты отображаются на вкладках «Синтаксис» или «Семантика».

# Тест производительности

Тест проводился на корпусе из 10 документов (5000 слов). Среднее время выполнения на среднестатистическом ПК:

* • Добавление документа (включая морфологический, синтаксический и семантический анализ): 3,0 с.
* • Поиск по слову: 0,03 с.
* • Получение семантического анализа: 0,15 с.

Основное время занимает семантический анализ.

# Вывод

Разработан анализатор для семантико-синтаксического анализа текстов. Приложение поддерживает загрузку текстов различных форматов, морфологический, синтаксический и семантический анализ, поиск, статистику, редактирование и экспорт результатов. Интерфейс интуитивно понятен, включает систему подсказок и вкладки для разных типов анализа. Перспективы: интеграция с настоящим ConceptNet, визуализация семантических графов, поддержка других языков.