УДК 004.934.2

РОЗРОБКА КОНЦЕПЦІЇ СИСТЕМИ ДЛЯ ОЦІНКИ ТА СИНТАКСИЧНОГО АНАЛІЗУ ТЕКСТУ

Науковий керівник – к.т.н., доцент кафедри ІТ Рудніченко М. Д.

Бакалавр – Жизнєв Д. І.

РАЗРАБОТКА КОНЦЕПЦИИ СИСТЕМЫ ДЛЯ ОЦЕНКИ И СИНТАКСИЧЕСКОГО АНАЛИЗА ТЕКСТА

Научный руководитель – к.т.н., доцент кафедры ИТ Рудниченко Н. Д.

Бакалавр – Жизнев Д. И.

SYSTEM CONCEPT FOR ASSESSMENT AND SYNTAXIC TEXT ANALYSIS DEVELOPMENT

Academic advisor – Ph.D., IT dept. assoc. prof. Rudnichenko M. D.

Bachelor – Zhyzniev D. I.

Анотація: *розглянуто специфіку та актуальність систем синтаксичного аналізу тексту, описано один зі способів технічної реалізації концепції системи та процес створення дерева залежності для синтаксичного аналізу тексту.*

Ключові слова: *синтаксичний аналіз тексту, веб-додаток, машинне навчання, natural language toolkit.*

Аннотация: *рассмотрена специфика и актуальность систем синтаксического анализа текста, описан один из способов технической реализации концепции системы и процесс создания дерева зависимостей для синтаксического анализа текста.*

Ключевые слова: *синтаксический анализ текста, веб-приложение, машинное обучение, natural language toolkit.*

Abstract: *considered specificity and relevance of text parsing systems, described one of the ways of technical implementation of the system concept and the process of creating a dependency tree for text parsing.*

Keywords: *syntax analysis, web-application, machine learning, natural language toolkit.*

Обробка формальної та природньої мови непроста алгоритмічна задача, вирішення якої допомагає в структуруванні даних (створення самих даних, їх опис, створення спеціалізованих конфігураційних даних, тощо), процесах трансляції коду, машинному перекладі та генерації текстів[1-3].

Однією з неочевидних цілей є процес знаходження залежності слів в тексті природньої мови. Вона досягається за допомогою синтаксичного та лексичного аналізу тексту, результатом чого є синтаксичне дерево залежності частин мови[4-6].

Сьогоднішні технологічні здобутки дають можливість виконати синтаксичний аналіз тексту різними способами, одним з яких є використання методів машинного навчання та фрейморків мови програмування Python.

Синтаксичний аналіз тексту– процес, який визначає чи належить деяка послідовність лексем (послідовність слів) мові, що породжує граматику. Цей процес виглядає наступним чином [7-9]:

- підготовка даних;
- обробка тексту;
- виділення частин мови;
- створення правил граматики;
- побудова синтаксичного дерева.

Підготовка даних. Цей етап має на меті процес отримання даних. Дані можна отримати, виконавши парсинг веб-сайту, написавши текст вручну або завантажити базу знань з текстами художної літератури. Після чого дані приводяться в формат, необхідний для наступної обробки. Мова програмування Pythonнадає вбудовані бібліотеки та модулі для підготовки даних [10].

Обробка тексту. Цей етап включає в себе перевірку орфографії, переведення усього тексту в нижній регістр, токенізацію (розбиття тексту на окремі слова або речення як окрему одиницю з якої можна працювати), пошук та видалення стоп-слів (слова, які є не важливими для синтаксичного аналізу) та лематизацію тексту (приведення слова до стандартного вигляду). Програмно це можна реалізувати за допомогою бібліотеки *NaturalLanguageToolkit (NLTK)*[11].

В лістингу 1 наведено програмний код для виконання перших двох етапів синтаксичного аналізу за допомогою бібліотеки NLTKта Python 3.8

Лістинг 1 – Приклад використання NLTK

```
importnltk
```

```
from nltk import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
# Текстовий приклад
example_text = “The little bear saw the fine fat trout in the brook”
# Текст в нижній регістр
example_text = example_text.lower()
#Токенізаціятексту
tokenized_text = word_tokenized(example_text)
#Видаляємо стоп-слова
#Завантажуємо словник зі списком стоп-слів англійської мови
stopwords = stopwords.words(‘english’)
#Пошук та видалення стоп-слів
text_without_stopwords = []
for word in tokenized_text:
   if word not in stopwords:
text_without_stopwords.append(word)
# Створюємо об’єкт класу WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()
lemmatized_words = [wordnet_lemmatizer(word) for word in text_without_stopwords]
```

Виділення частин мови. Для побудови дерева залежності необхідно розуміти між чим необхідно визначити залежність. Для цього використовується розмітка за частинами мови (part-of-speechtagging). Засоби NLTKможуть встановити лише частину речення. Для додаткового морфологічного аналізу (відмінок, рід, число) використовуються самописні додатки, наприклад, *rnnmorph*та *pymorphy2*, які працюють на основні нейронних мереж та можуть з великою точністю (90%+) правильно виділити частину мови [2].

Створення правил граматик. Цей етап є досить складним в реалізації, оскільки вбудовані правила граматики в NLTKне дозволяють досить чітко встановити залежності між частинами мови, а отже між словами, але для створення робочого прототипу цього вистачить. Тому при модернізації системи, в першу чергу, треба вирішити проблему створення граматичних правил.NLTKстворює граматичні правила на основі NER-розмітки (namedentityrecognition).

Побудова синтаксичного дерева. На цьому етапі на основі раніше створених правил граматики будується синтаксичне дерево (рис.1) [3].

В якості графічного інтерфейсу користувача можна використати веб-фреймворк Django. Він написаний на Python, тому перенесення програмної логіки в back-endне викличе труднощів. Однак слід пам’ятати, що стандартна архітектура MVC (Model-View-Controller) представлена в Djangoяк MTV (Model-Template-View), де бізнес логіка розташовується в шарі View, що не є правильним з точки зору подальшого масштабування та безпеки додатку. Для збереження можливості безболісного масштабування додатку необхідно створити новий шар (на практиці створюється окремий файл в каталозі з проектом), в якому буде розміщена бізнес-логіка додатку.

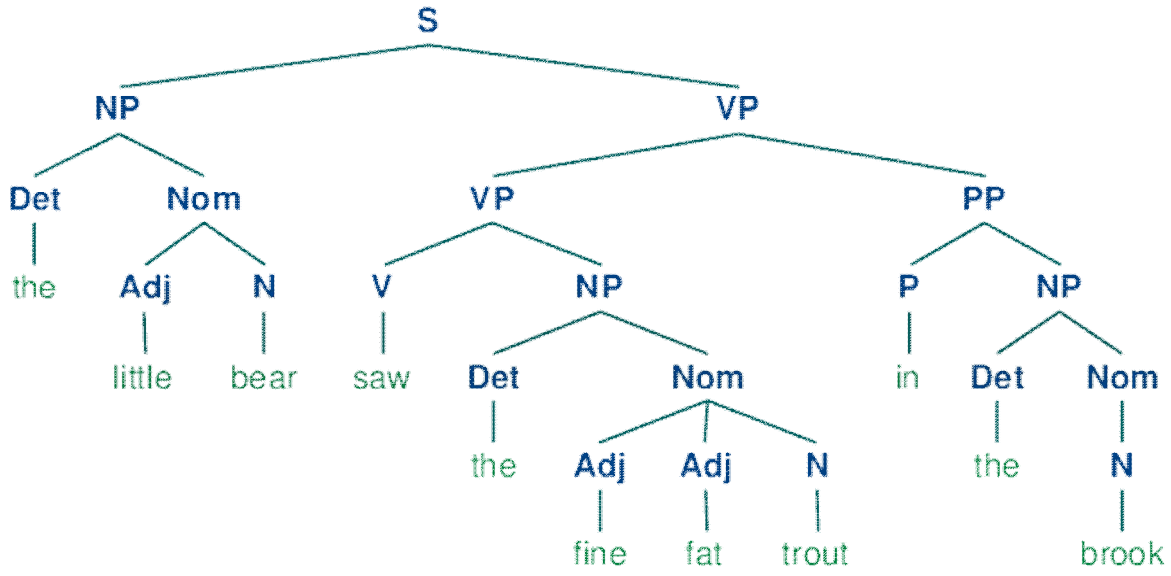

Рисунок 1 – Синтаксичне дерево

Оформлення та вибір елементів керування GUIзалежить від цілей, які переслідує розробник. Потрібно враховувати, що користувач може вводити дані як власноруч (у відповідне текстове поле) так і завантаживши файл з текстом.

Додатково можна надати користувачеві можливість самостійно навчити комп’ютер виконувати синтаксичний аналіз тексту засобами машинного навчання з можливістю серіалізації та десеріалізації моделі. Для цього користувачеві необхідно обрати алгоритм навчання, налаштувати гіперпараметри та додати датасет для навчання моделі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ахо А. Теорія синтаксичного аналізу, перекладу та компіляції / А. Ахо, Дж. Ульман. – М.: Мир, 1978. – 614 с.

2. BecomingHuman. Getting started with NLP using NLTK [Електронний ресурс]. Режим доступу: https://becominghuman.ai/nlp-for-beginners-using-nltk-f58ec22005cd

3. NLTK Documentation. Analyzing Sentence Structure. Context Free Diagram [Електронний ресурс]. Режим доступу: https://www.nltk.org/book/ch08.html

4. Rudnichenko N. Information Systemfor the Intellectual Assessment Customers Text Reviews Tonality Basedon Artificial Neural Networks / N. Rudnichenko, S. Antoshchuk, V. Vychuzhanin, A. Ben, I. Petrov // Proceedingsofthe 9th InternationalConference "InformationControlSystems& Technologies", Odessa, Ukraine, September 24–26, 2020. – Odessa: Odessa National Polytechnic University, 2020. − P. 371-385.

5. Rudnichenko N. Decision Support System for the Machine Learning Methods Selectionin Big Dat aMining / N. Rudnichenko, V. Vychuzhanin, I. Petrov, D. Shibaev // Proceedings 0f The Third International Workshopon Computer Modeling and Intelligent Systems (CMIS-2020): session 6 “Intelligent Information Technologies” April 27-May 1, 2020. – Zaporizhzhia: NU “Zaporizhzhia Polytechnic” (editedby S. Subbotin), 2020. − P. 872-885.

6. Рудниченко Н. Д. Разработка програмного приложения решения задач классификации на основе использования методов машинного обучения / Н. Д. Рудниченко, В. В. Вычужанин, Н. О. Шибаева, Н. И. Гежа // Матеріали IX Міжнародної науково-практичної конференції «Інформаційні управляючі системи і технології» 24–26 вересня 2020 р., м. Одеса. - 2020. - С.307-310.

7. Філінський О. А. Розробка проекту програмного забеспечення обробки та аналізу текстової інформації / О. А. Філінський, М. Д. Рудніченко, Т. В. Отрадська // Матеріали IX Міжнародної науково-практичної конференції «Інформаційні управляючі системи і технології» 24–26 вересня 2020 р., м.Одеса. - 2020. - С. 205-207.

8. Рудниченко Н. Д. Разработка програмного обеспечения інтерактивного интеллектуального анализа больших данных на базе методов машинного обучения / Н. Д. Рудниченко, В. В. Вычужанин, Н. О. Шибаева, Д. С. Шибаев, Т. В. Отрадская, И. М. Петров // Актуальные проблемы информационных систем и технологий: монография / Борисова Н., Васянин В. идр.; под науч. ред. проф. Вычужанина Владимира. - Одесса: НУ "ОМА", 2020. - С. 59-71.

9. Vychuzhanin V. V. Analysis and structuring diagnostic large volume data of technical condition of complexe quipmentintransport / V. V. Vychuzhanin, N. R. Rudnichenko, Z. Sagova, M. Smieszek, V. V. Cherniavskyi, A. I. Golovan, M. V. Volodarets // 24th Slovak-Polish International Scientific Conferenceon Machine Modelling and Simulations - MMS 2019, 3-6 September 2019, LiptovskýJán, Slovakia.

10. Гежа М. І. Аналіз існуючих програмних продуктів побудови та дослідження моделей машинного навчання / М. І. Гежа, С. Є. Тищенко, М. Д. Рудніченко // Інформатика, інформаційні системи та технології: тези доповідей шістнадцятої всеукраїнської конференції студентів і молодих науковців. Одеса, 23 квітня 2021 р. - Одеса, 2021. - С. 24-25.

11. Задунайська О. Г. Аналіз можливостей застосування методів обчислювального інтелекту для вирішення завдань інтелектуального аналізу даних великого обсягу / О. Г. Задунайська, Г. Р. Волошко, М. Д. Рудніченко // Сімнадцята всеукраїнська конференція студентів і молодих науковців «Інформатика, інформаційні системи та технології». Одеса, 24 квітня 2020 р. – Одеса, 2020. – С. 31-32.