DOI 10.34132/mspc2025.01.14.22

Микола Фісун,
Ігор Кандиба

СТВОРЕННЯ СЛОВНИКА ПРЕДМЕТНОЇ ГАЛУЗІ ЗАСОБАМИ PYTHON

Тези представляють аналіз можливості побудови словника предметної галузі засобами мови програмування Python. Представлено різні структури даних для зберігання сформованих словників предметної галузі та засоби Python для роботи з цими структурами. Описано варіант застосування моделі з підтримкою SPARQL-запитів. Визначено програмні засоби для аналізу тексту та виділення знайдених термінів. Коротко наведено опис запропонованих інструментів для пошуку термінів. Наведено опис процесу пошуку термінів з використанням засобів обробки природної мови. Запропоновано варіант застосування пошуку шаблонних конструкцій для визначення значень знайдених термінів. Описано можливість застосування регулярних виразів для пошуку та обробки шаблонних конструкцій Розглянуто варіант використання великих мовних моделей для створення словнику предметної галузі.

Ключові слова: *словник предметної галузі, RDF, NLTK, Python.*

Словниками предметної галузі називають структуровані зібрання термінів, понять і визначень, що відображають специфічну лексику певної професійної або наукової галузі. А в сучасних інформаційних системах такі словники використовуються навіть у якості основи для побудови онтологій та семантичних моделей предметної галузі [1].

Словники,зазвичай, розробляються фахівцями предметної галузі, при цьому рівень автоматизації таких процесів залишається ще незадовільним. Тому питання аналізу, вибору та подальшого розвитку інструментів обробки текстової інформації. що подається природньої мовою, залишаються актуальними.

У сучасних інформаційних технологіях для зберігання та обробки словників предметної галузі можуть використовуватися різні формати даних. Одними з найпоширеніших є формат електронних таблиць – Comma-Separated Values (CSV) [2]. Завдяки своїй простоті, цей формат зручний для представлення словників предметної галузі у вигляді таблиці з колонками "Термін", "Визначення", "Категорія", "Синоніми" тощо, що дозволяє легко обробляти такі дані за допомогою мов програмування загального призначення: C#, Java, Python, тощо.

Варто згадати про можливість використання формату JavaScript Object Notation (JSON) для представлення словника предметної галузі. Цей формат має складнішу структуру ніж CSV, але є більш гнучким і дозволяє створювати словники з додатковими властивостями. JSON дозволяє зберігати терміни разом з їхніми визначеннями, синонімами, категоріями та іншими атрибутами в зручній, для подальшої обробки структурі даних. Згаданий формат часто використовується при розробці вебзастосунків та при виконанні семантичного аналізу.

Найчастіше для зберігання та обробки словників предметної галузі використовується Resource Description Framework (RDF). Цей формат використовується для створення онтологій – формалізованих словників з логічними зв’язками. На відміну від описаних вище формат RDF призначений для моделювання даних, розроблений W3C для опису інформації у вигляді трійок "суб’єкт–предикат–об’єкт", що дозволяє формалізовано представляти знання та зв’язки між об’єктами. У контексті словників предметної галузі RDF використовується для опису термінів [2], їхніх визначень, синонімів, ієрархічних або семантичних зв’язків у вигляді графів. Варто зазначити, що на основі RDF начастіше створюють онтології. Онтології є формалізованими структурами, які описують знання про предметну область у вигляді понять (класів), зв’язків між ними (відношень) та властивостей. Ці сутності дозволяють не лише визначити певні терміни предметної галузі, а й відобразити їх зв’язок.

Мова програмування Python підтримує можливість роботи з усіма перерахованими типами даних. Проте, найчастіше для роботи з словниками предметних галузей використовують RDF. Цей факт обумовлений використанням RDF у семантичному вебі та в ПЗ, побудованому на використанні онтологіями. Інтеграція RDF-словники з Python виконується за допомогою бібліотек, що підтримують обробку запитів до даних в форматі RDF, найрозповсюдженіша з яких є RDFlib. Ця бібліотека дозволяє створювати RDF-графи, додавати трійки, зчитувати RDF-файли в різних форматах (Turtle, RDF/XML, N-Triples, тощо) і виконувати SPARQL-запити. Для створення словника предметної галузі необхідно створення відповідного простору імен:

```
from rdflib import Graph, Literal, RDF, URIRef, Namespace
```

```
g = Graph()
EX = Namespace("https://chmnu.edu/")

g.add((EX["Python"], RDF.type, EX["Programming_language"]))
g.add((EX["Python"], EX["Supports_paradigm"], Literal("Object-oriented_programming ")))
g.add((EX["Python"], EX["Supports_paradigm"], Literal("structured")))
g.add((EX["Python"], EX["Supports_paradigm"], Literal("structured")))
g.add((EX["Python"], EX["Has_a_translator_type"], EX["interpreter"]))
g.serialize("python_example.rdf", format="xml")

for subj, pred, obj in g:
    print(f"{subj} -- {pred} --> {obj}")
```

Підтримка запитів мовою SPARQL є невід'ємною частиною ПЗ для створення словника предметної галу на основі RDF. SPARQL (SPARQL Protocol and RDF Query Language) являє собою мову запитів для вибірки та обробки даних, збережених у форматі RDF. Ця мова фактично є аналогом SQL у реляційних базах даних, SPARQL дозволяє звертатися до RDF-графів, шукати трійки, фільтрувати дані, обирати змінні, виконувати агрегацію та комбінувати результати:

```
qres = g.query("""
    SELECT  ?Programming_language ?Supports_the_paradigm
    WHERE  {
        ?Programming_language <https://chmnu.edu/Supports_paradigm> ?Supports_the_paradigm .
    }
""")

for row in qres:
    print(f"Programming language: {row.Programming_language}, Supports_the_paradigm: {row.Supports_the_paradigm}")
```

Варто зауважити, що повністю автоматизувати процес створення словника предметної галузі на основі аналізу тексту практично не можливо, але виділити основні терміни можливо. Засобами Python можлива часткова автоматизація процесу створення словника предметної галузі. Для цього необхідно використати додаткові бібліотеки для обробки природної мови: NLTK, Rake, Spacy тощо. Обробка природної мови (Natural Language Processing, NLP) засобами Python є вагомою галузей штучного інтелекту, що полягає в реалізації аналізу, інтерпретації та генерувати тексти природною мовою за допомогою програмного забезпечення.

Одним з інструментів для автоматизованого створення словнику предметної галузі є бібліотека rake, що є реалізацією однойменного алгоритму. RAKE (Rapid Automatic Keyword Extraction) є алгоритмом для автоматичного виявлення ключових слів і фраз із тексту без потреби в попередньому навчанні чи складному лінгвістичному аналізі. Rake працює на основі статистики, розділяючи текст на фрази за «стоп-словами», оцінюючи значущість кожного слова, а потім обчислюючи вагу фраз як суми ваг її складових.

Хоча використання бібліотеки rake дасть змогу виділи основні ключові слова, але для формування словника предметної галузі необхідно також виділити їх визначення. Для реалізації цього процесу необхідно використання інструментарію NLTK. NLTK (Natural Language Toolkit) є однією з найпопулярніших бібліотек Python для обробки природної мови, що реалізує широкий функціонал для: токенізації, стемінгу, лематизації, визначення частини мови, синтаксичного розбору тощо [4].

Процес створення словника предметної галузі засобами NLTK необхідно виконати виділення термів. Для цього потрібно виконати кілька дій: видалити сполучники та розділові знаки методом stopwords.words( ), провести токенізацію word_tokenize( ), розпізнати частину мови pos_tag( ) та виділити лише іменники startswith('NN'), де «NN» – позначення іменників в однині. Такий підхід обмежений можливістю вибору лише термів іменників, але може бути модифікований шляхом додавання нового функціоналу і включати у якості термів інші частини мови. В Python код для такої обробки тексту з файлу «text.txt» виглядатиме наступним чином:

```
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger') # Для англійської POS-розмітки
nltk.download('averaged_perceptron_tagger_eng')
nltk.download('stopwords')

with open("text.txt", "r", encoding="utf-8") as file:
    text = file.read()

tokens = word_tokenize(text.lower())
filtered = [w for w in tokens if w.isalpha() and w not in stopwords.words('english')]
tagged = pos_tag(filtered)
terms = [word for word, pos in tagged if pos.startswith('NN')]
```

Проте після виділення термів неможливо створити словник предметної галузі. Для цього потрібно виділити також значення самих термів, що можливо зробити, виконавши пошук шаблонних конструкцій з визначенням цих термів: «X – це ...», «X представляє собою…», «X є…» «Під X розуміють ...», «X визначається як ...», де X – це терм, визначений на попередньому кроці. Цей процес можна автоматизувати додатковими засобами: spaCy, Stanza, AllenNLP тощо, але ці засоби не підтримують можливість роботи з українською мовою. Можна здійснити пошук цих конструкцій за допомогою регулярних виразів використавши модуль RE та функцію re.findall( ), що знаходить всі відповідності регулярним виразам.

Інший варіант формування словника предметної галузі – це застосування засобів ШІ, наприклад GPT-моделі [5]. Для цього можливо використати бібліотеку OpenAI, що дозволяє використання різних моделей. Перевагою цього методу є те, що модель може виконувати пошук по наявному тексту або знайти визначення певних термів самостійно використовуючи інформацію з мережі Інтернет.

Використання мовної моделі-GPT дозволяє автоматизувати завдання обробки природної мови. За допомогою бібліотек, таких як OpenAI, розробники можуть інтегрувати доступ до моделей GPT у свої застосунки, надсилаючи текстові запити та отримуючи текстову відповідь. Це відкриває широкі можливості для створення інтелектуальних пошукових систем, асистентів для написання коду чи текстів, а також інструментів для аналізу великих обсягів текстової інформації. GPT-моделі можна адаптувати під конкретні потреби, використовуючи додаткові параметри запиту або навчання на прикладах.

References

1. Veselovskaya, G. V., & Lebed, O. S. (2018). Models of the subject field in computer multimedia projection technologies. Applied Questions of Mathematical Modeling, 2, 203–211. https://doi.org/10.32782/2618-0340-2018-2-203-211
2. Nayak, A., Božić, B., & Longo, L. (2022). Data Quality Assessment of Comma Separated Values Using Linked Data Approach. У *Business Information Systems Workshops* (p. 240–250). Springer International Publishing
3. Dongo, I., Cardinale, Y., & Chbeir, R. (2018). RDF-F: RDF Datatype inFerring Framework. Data Science and Engineering, 3(2), 115–135.
4. Nelli, F. (2018). Textual Data Analysis with NLTK. У Python Data Analytics (p. 487–506). Apress.
5. Cao, Z., & Ren, Q. (2024). Towards python program repair with generative pre-trained transformer (GPT-3.5). Theoretical and Natural Science, 43(1), 102–112.

DOI 10.34132/mspc2025.01.14.22

Mykola Fisun,
Ihor Kandyba

PYTHON-BASED CREATION OF A DOMAIN DICTIONARY

This abstract presents an analysis of the feasibility of creating a domain-specific dictionary using the Python programming language. Various data structures for storing the generated domain-specific dictionaries and Python tools for working with these structures are presented. A use case for a model with SPARQL query support is described. Software tools for text analysis and the extraction of identified terms are defined. A brief description of the proposed tools for term discovery is provided. The process of term discovery using natural language processing tools is described. An approach involving the search for pattern constructions (template matching) to determine the meanings of the identified terms is proposed. The applicability of regular expressions for searching and processing these pattern constructions is described. The option of using large language models for creating the domain-specific dictionary is considered.

Key words: *domain vocabulary, RDF, NLTK, Python.*

Відомості про автора / Information about the Author

Микола Фісун, д-р техн. наук, професор кафедри інженерії програмного забезпечення факультету комп'ютерних наук Чорноморського національного університету імені Петра Могили, м. Миколаїв, Україна. E-mail: ntfis@chmnu.edu.ua, orcid: https://orcid.org/0000-0003-1297-6230.

Mykola Fisun, Doctor of Technical Sciences, Professor of the Department of Software Engineering Faculty of Computer Science the Petro Mohyla Black Sea National University, Mykolaiv, Ukraine. E-mail: ntfis@chmnu.edu.ua, orcid: https://orcid.org/0000-0003-1297-6230.

Ігор Кандиба PhD, старший викладач кафедри інженерії програмного забезпечення факультету комп'ютерних наук Чорноморського національного університету імені Петра Могили, м. Миколаїв, Україна. E-mail: ihor.kandyba@chmnu.edu.ua, orcid: https://orcid.org/0000-0002-8589-4028

Ihor Kandyba, PhD, Senior Lecturer of Software Engineering Faculty of Computer Science the Petro Mohyla Black Sea National University, Mykolaiv, Ukraine. E-mail: ntfis@chmnu.edu.ua, orcid: https://orcid.org/0000-0002-8589-4028

Стаття отримана редакцією 19.05.2025.