Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Методи подання знань, засновані на обробці природно-мовних текстів та графів знань
(тема)

Виконав:
студент 2 курсу, групи СШІм-19-2
Дмитрієв Д.Ю.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник проф. Рябова Н.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо–наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Дмитрієву Даниїлу Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи подання знань, засновані на обробці природномовних текстів та графів знань

затверджена наказом університету від 29 03 20 21 р. № 390 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 19 05 20 21 р.

3. Вихідні дані до роботи Функція: Розробка компонентів побудови графів знань з неструктурованих текстових джерел. Організація даних: файлова з прямим доступом. Перелік використовуваних програмних засобів: ОС Windows 10, середовище розробки Jupyter Notebook, середовище розробки Google Colab, мова програмування Python.

4. Перелік питань, що потрібно опрацювати в роботі Аналіз предметної галузі. Загальна характеристика моделі подання знань – графів знань та методів обробки природно-мовних тексті. Огляд методів витягу сутностей та відносин, тегування частин мови, розпізнавання іменованих сутностей, тегування видів граматичних відносин. Опис алгоритму витягу знань з тексту для вирішення задачі побудови графу знань. Постановка цілей і задачі дослідження. Розробка та опис програмного додатку. Проведення аналізу виконаної роботи та формування висновків.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Рисунок 1 – Компоненти інтелектуальної системи; Рисунок 2 – Приклад фрейму; Рисунок 3 – Семантична мережа; Рисунок 4 – Застосування графів знань; Рисунок 5 – Подання знань у графі знань; Рисунок 6 – Схема системи на основі графу знань; Рисунок 7 – Діаграма знань; Рисунок 8 – Вихідний граф знань; Рисунок 9 – Приклад використання методів NLP; Рисунок 10 – Приклад маркування за схемою BIOES; Рисунок 11 – Приклад використання NER та POS; Рисунок 12 – Алгоритм роботи Snowball; Рисунок 13 – Схема дистанційно контрольованого RE; Рисунок 14 – Приклад роботи токенізатора Spacy; Рисунок 15 – Приклад результатів тегування відносин та POS; Рисунок 16 – Процес роботи конвеєра у spaCy; Рисунок 17 – Приклад текстового джерела; Рисунок 18 – Приклад маркування залежностей в реченні; Рисунок 19 – Відносини, що найчастіше зустрічалися в тексті; Рисунок 20 – Видобуті з передобробленого тексту сутності; Рисунок 21 – Приклад вихідного графу знань; Рисунок 22 – Відносина «скомпазовано (кимось)»; Рисунок 23 – Відносина «написано (кимось)»

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	29.03.2021	Виконано
2	Аналіз завдання та об'єкту дослідження	01.04.2021	Виконано
3	Пошук та аналіз тематичної літератури	03.04.2021	Виконано
4	Вибір програмних та технічних засобів	04.04.2021	Виконано
5	Проектування та розробка програмного засобу	08.04.2021	Виконано
6	Аналіз результатів роботи програмного засобу	13.04.2021	Виконано
7	Підготовка пояснювальної записки	18.04.2021	Виконано
8	Проходження нормконтролю та рецензування	23.04.2021	Виконано
9	Перевірка на плагіат	27.04.2021	Виконано
10	Перевірка печатної версії роботи	12.05.2021	Виконано
11	Попередній захист кваліфікаційної роботи	17.05.2021	Виконано
12	Захист кваліфікаційної роботи перед ЕК	19.05.2021	

Дата видачі завдання 29 березня 20 21 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 100 с., 1 табл., 25 рис., 2 дод., 24 джерела.

ГРАФИ ЗНАНЬ, ОБРОБКА ПРИРОДНО-МОВНИХ ТЕКСТІВ, ПОДАННЯ ЗНАНЬ, CSV, JUPYTER NOTEBOOK, PYTHON, RULE–BASED MATCHING.

Об’єкт дослідження – основні підходи до подання знань в інтелектуальних системах.

Предмет дослідження – методи подання знань, засновані на обробці природно-мовних текстів та графів знань.

Мета роботи – дослідження та використання методів подання знань на основі обробки природно-мовних текстів та графів знань.

Методи роботи – методи інженерії знань, методи теорії графів, методи автоматичної обробки природно-мовних текстів, зокрема, сегментації речень, синтаксичний аналіз залежностей, тегування частин мови, розпізнавання сутностей, практична апробація результатів досліджень із застосуванням інтегрованого середовища розробки Jupyter Notebook, мови програмування Python.

Результати кваліфікаційної роботи – в результаті проведених досліджень вирішено задачу порівняння та аналізу різноманітних методів побудови графів за допомогою методів обробки природно-мовних текстів. На основі проведених досліджень створено прототип програмного додатку.

Область застосування – дана розробка може бути корисною для задач аналізу тексту, семантичного пошуку, а також у рекомендаційних системах.

РЕФЕРАТ

Пояснительная записка: 100 с., 1 табл., 25 рис., 2 доп., 24 источника.

ГРАФЫ ЗНАНИЙ, ОБРАБОТКА ЕСТЕСТВЕННО-ЯЗЫКОВЫХ ТЕКСТОВ, ПРЕДСТАВЛЕНИЕ ЗНАНИЙ, CSV, JUPYTER NOTEBOOK, PYTHON, RULE–BASED MATCHING.

Объект исследования – основные подходы к представлению знаний в интеллектуальных системах.

Предмет исследования – методы представления знаний, основанные на обработке естественно-языковых текстов и графов знаний.

Цель работы – исследование и использование методов представления знаний на основе обработки естественно-языковых текстов и графов знаний.

Методы работы – методы инженерии знаний, методы теории графов, методы автоматической обработки естественно-языковых текстов, в частности, сегментации предложений, синтаксический анализ зависимостей, пометка частей речи, распознавания сущностей, практическая апробация результатов исследований с применением интегрированной среды разработки Jupyter Notebook, языка программирования Python .

Результаты квалификационной работы – в результате проведенных исследований решена задача сравнения и анализа различных методов построения графов с помощью методов обработки естественно-языковых текстов. На основе проведенных исследований создан прототип программного приложения.

Область применения – данная разработка может быть полезной для задач анализа текста, семантического поиска, а также в рекомендательных системах.

ABSTRACT

Explanatory note: 100 p., 1 tabl., 25 fig., 2 ann., 24 sources.

CSV.JUPYTER NOTEBOOK, KNOWLEDGE PRESENTATION, KNOWLEDGE GRAPHS, NATURAL LANGUAGE PROCESSING, PYTHON, RULE–BASED MATHCING.

The object of research – methods of knowledge representation, knowledge graphs, processing of natural language texts.

The subject of research – methods of knowledge representation.

The purpose of the work is to study and use methods of knowledge representation based on the processing of natural language texts and knowledge graphs.

Methods of work – methods of constructing knowledge graphs, methods of sentence segmentation, parsing of dependencies, part of speech tagging, entity recognition, Jupyter Notebook integrated development environment, Python programming language.

Results of qualification work – as a result of the conducted researches the problem of comparison and the analysis of various methods of construction of knowledge graphs using natural language texts processing approaches is solved. Based on the research, a prototype software application was created.

Scope – this development can be useful for tasks of text analysis, semantic search, as well as in recommendation systems.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень, термінів 8
Вступ 9
1 Аналіз предметної області та постановка задач дослідження 11
1.1 Огляд і аналіз сучасного стану розгляду проблеми 11
1.2 Подання знань в системах заснованих на знаннях 13
1.3 Методи подання знань 17
1.4 Подання знань у Web-просторі 22
1.5 Постановка задач дослідження 26
2 Графи знань 28
2.1 Сучасний стан та області застосування графів знань 28
2.2 Загальна характеристика графів знань 35
2.3 Алгоритм побудови графу знань з тексту на природній мові 45
3 Методи обробки природно-мовних текстів 51
3.1 Загальна характеристика методів обробки природної мови 51
3.2 Видобування сутностей 56
3.3 Видобування відносин 62
4 Практичне застосування отриманих результатів досліджень 75
4.1 Обгрунтування вибору програмного середовища 75
4.2 Опис алгоритму та демонстрація роботи програмного додатку 80
Висновки 90
Перелік джерел посилання 92
Додаток А Текст програми 95
Додаток Б Відомість кваліфікаційної роботи магістра 100

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІІС – інтелектуальна інформаційна система;

ПрО – предметна область;

ЕЕ – entity extraction – видобування сутностей;

IE – information extraction – видобування інформації;

KG – knowledge graph – граф знань;

KR – knowledge representation – подання знань;

NER – named entity recognition – розпізнавання іменованих сутностей;

NLP – natural language processing – обробка природної мови;

POS – part of speech tagging – тегування частин мови;

RE – relation extraction – витяг відносин.

ВСТУП

На сьогоднішній день у сфері інформаційних технологій найбільш актуальною областю стали різноманітні інтелектуальні технології. Системи на основі штучного інтелекту знайшли широке застосування у великій кількості систем розроблених для виконання широкого спектра задач. Одним з видів таких інтелектуальних систем стали інтелектуальні інформаційні системи, розроблені для здійснення підтримки діяльності людини і пошуку інформації. Однак робота інтелектуальних інформаційних систем зіткнулася з проблемою обробки великих об’ємів неструктурованих даних, які були непридатні для розуміння комп’ютером. Відповідним рішенням цієї проблеми стало перетворення даних у форму знань.

Робоче визначення знань замикається в тому, що знання – це набір сутностей або речей, що існують у реальному світі, їх властивостей та відносин між ними, які краще моделюють реальний мир. Область подання знань у свою чергу сфокусована на побудові моделей, де знання будуть представлені у придатній для обробки комп’ютером та зрозумілій для людей. Подання знань у роботі інтелектуальних інформаційних систем пов'язана з використанням різноманітних моделей для презентації набору тверджень на основі яких система може приймати рішення, давати рекомендації тощо. Таким чином, подання знань – це частина ШІ яка пов'язана з тим, як система використовує те, що вона знає, при прийнятті рішення про те, що робити [1].

Безпрецедентна кількість даних на даний момент зберігається у текстовому вигляді, тому для вилучення знань та подальшого їх подання інформаційні системи потребують методи обробки природної мови. Природна мова, без сумніву, є одним з найбільш важливих для користувача інтерфейсів майбутнього. Вже сьогодні послуги, які можуть запропонувати комп'ютери, значно відстають від послуг, які фактично

пропонуються користувачеві. Користувачі можуть зробити набагато більше, але поки що існує проблема з інтеграцією цих потужних можливостей для використання.

Вирішенням цієї проблеми є знання, що дозволяють комп'ютеру та користувачу ефективно взаємодіяти з сутностями, що знаходяться у природо-мовних текстах. Існує велика кількість видів подання знань, таких як онтології, семантичні мережі, фреймові моделі, але найсучаснішим підходом для вирішення даної задачі стали «графи знань» (Knowledge Graphs, KG), що дозволяють здійснювати організацію та представлення неструктурованих текстових даних у вигляді знань [2].

Графи знань надають найбільш відповідний для інтерпретації і ефективний спосіб зберігання різнорідних знань. Графи знань пропонують простоту обслуговування і самоаналізу, що перевершує багато альтернативи. Було здійснено багато спроб вирішити проблему неструктурованих даних, однак граф знаний – це найсучасніший та найкращий спосіб узгодження даних. Усі дані, джерела даних та бази даних будь-якого типу можуть бути представлені та реалізовані у вигляді графу.

Обробка природної мови – це розблокування знань, виражених природною мовою, які потім можна використовувати для оновлення і перевірки на узгодженість щодо графа знань, який, в свою чергу, може бути доступний через природну мову для запитів і генерації відповідей. Ці дві технології йдуть рука об руку доповнюючи та поліпшуючі одна одну.

Враховуючи вищесказане, в даній роботі буде проведено дослідження методів подання знань, заснованих на обробці природно-мовних текстів та графів знань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Огляд і аналіз сучасного стану розгляду проблеми

Проблеми подання знань та побудови і використання різних видів моделей для подання знань є одним з пріоритетних напрямів розвитку сучасних інтелектуальних інформаційних систем. Подання знань для області інформаційних технологій – це поняття, яке найкраще можна зрозуміти з точки зору п'яти різних ролей, які це поняття відіграє, де кожна з ролей має вирішальне значення [3], [4], [5]:

– подання знань (KR) – це, по суті, сурогат, заміна самої речі, яка використовується, щоб дозволити суті визначати наслідки, думаючи, а не діючи, тобто розмірковуючи про світ, а не діючи в ньому;

– це набір онтологічних зобов'язань, тобто відповідь на питання: в яких термінах треба думати про «світ», тобто модель проблемної області, яка використовується у конкретній інтелектуальній системі;

– це фрагментарна теорія розумного міркування, виражена в термінах трьох компонентів: фундаментальна концепція, яка представляє розумне міркування; набір висновків щодо представлених концептів; набір висновків, які є результатом логічного виведення із заданих концептів на основі прийнятих міркувань;

– це середовище для прагматично ефективних обчислень, тобто обчислювальне середовище, в якому здійснюється мислення. Одним із вкладів в цю прагматичну ефективність є процес управління знаннями, завдяки чому інформація представляється у вигляді, найбільш зручному для виконання рекомендованих висновків;

– це засіб людського вираження, тобто мова, на якій ми говоримо про світ.

Розуміння ролей і визнання їх різноманітності має кілька корисних наслідків. По-перше, кожна роль вимагає чогось трохи відмінного від подання; відповідно, кожне з них призводить до цікавого і різному набору властивостей, які ми хочемо, щоб уявлення мало.

По-друге, ролі забезпечують основу, корисну для характеристики широкого спектру репрезентацій. Фундаментальний «образ мислення» уявлення можна отримати, зрозумівши, як воно розглядає кожну з ролей, і що це виявляє істотні схожості і відмінності.

Така точка зору на подання знань має наслідки як для досліджень, так і для практики. Для дослідження ця точка зору дає одну пряму відповідь на питання, що має фундаментальне значення в даній області. Вона також пропонує прийняти широкий погляд на те, що важливо в поданні знань, і доводить, що однією з найважливіших частин є подання багатства природного світу.

Багато сучасних інформаційних систем використовують засоби штучного інтелекту для більш ефективного надання послуг. Такі системи потребують засоби, що можуть дозволити їм міркувати, робити висновки та надавати на їх основі рекомендації або корисну інформацію. Але через те що процес міркування у таких системах здійснюється комп’ютером, розробка інтелектуальних інформаційних систем стикається з проблемою інтерпретації інформації в формі, зрозумілій для людини та машини. Розробники інформаційних систем дуже потребують чітко формалізовану структуровану інформаційну модель, що буде зручно відображати всі концепти та поняття з якими буде працювати система, що дозволяє більш ретельно підійти до процесу проектування система та ефективно організувати процес побудови інформаційної система. Інтелектуальна інформаційна система у свою чергу за допомогою подання знань отримує можливість організувати процес мислення та надання висновків ґрунтуючись на знаннях, якими людина може описувати об’єкти реального

світу та надаючи висновки з цих знань, які будуть легко сприйматися користувачами системи.

Серед всіх моделей подання знань саме графи знань в останні роки набули значної популярності. Їх використовують у побудові інтелектуального контенту, контекстно-залежних рекомендаціях по контенту, виявлення ліків на основі графа знань, семантичному пошуку, аналізу інвестиційного ринку, виявлення інформації в нормативних документах та великій кількості інших областей.

Такі моделі подання знань можуть зберігату велику кількість знань про різні концепти, терміни та об’єкти реального світу переважно на природній мові, тому для побудови графів знань часто використовуються методи обробки природної мови.

1.2 Подання знань в системах, заснованих на знаннях

Безперечно на сьогоднішній день існує величезна кількість різноманітних даних про необ'ємну кількість сутностей реального світу. Тому з точки зору розробників систем, саме підхід, заснований на знаннях, є найбільш зручним та ефективним способом для роботи з різноманітними даними, бо має такий ряд переваг [1], [5]:

– можливість легко додавати нові знання і легко робити їх залежними від попередніх знань;

– можливість розширення існуючої поведінки системи за допомогою додавання нових знань;

– можливість налагодження неправильної поведінки шляхом виявлення помилкових знань;

– можливість коротко пояснити та обґрунтувати поведінку системи за допомогою знань, що легко сприймаються як людиною, так і комп’ютером.

Але через те, що існує велика кількість знань на різну тематику, багато систем заснованих на знаннях є універсальними і в них не завжди визначено список завдань, які буде вирішувати система. Системи, що працюють зі знаннями наперед не знають, коли ці знання будуть застосовані на практиці, тому дуже актуальною стає проблема збереження знань у найбільш зручній для подальшої обробки формі. Через це розробники систем, заснованих на знаннях, змушені шукати шляхи для реалізації найбільш ефективних моделей для подання знань та їх подальшої обробки. Тому більшість систем, заснованих на знаннях, характеризують такі поняття, як наявність бази знань, набору формалізованих структур, які представляють те, у що система вірить, і які робить висновки під час своєї роботи .

Люди безперечно є кращими у розумінні, міркуванні та інтерпретації знань. Людина зберігає в пам’яті речі, які є знаннями, і відповідно до них виконує різні дії в реальному світі. Але те, як машини працюють зі знаннями, залежить від подання знань і міркувань. Отже, подання знань в системах, заснованих на знаннях грає такі ролі:

– подання знань, як частина інтелектуальних систем, яка пов'язана з мисленням агентів ШІ, сприяє інтелектуальній поведінці агентів;

– відповідає за подання інформації про реальний світ, щоб комп'ютер міг розуміти і використовувати ці знання для вирішення складних проблем реального світу, таких як діагностика стану здоров'я або спілкування з людьми на природній мові;

– описує, як ми можемо уявити знання в інтелектуальних системах. Подання знань – це не просто збереження даних в деякій базі даних, але також можливість інтелектуальної машині вчитися на цих знаннях і досвіді, щоб вона могла вести себе розумно, як людина.

Подання знань грає дуже важливу роль у побудові систем, заснованих на знаннях, тобто систем, працездатність яких частково залежить від міркувань на основі явно представлених знань. Найбільш

відомим та яскравим прикладом систем, заснованих на знаннях, є експертні системи, які містять бази знань, відповідних знанням експертів певної предметної області, та здійснюють логічне виведення на цих знаннях, використовуючи механізм міркувань, який емулює дії експерта при вирішенні проблем. Але існує багато складноструктурованих та слабоформалізованих предметних областей та галузей знань, де доволі успішно використовується підхід, заснований на знаннях, щодо вирішення різних прикладних задач. Подання знань та бази знань часто використовуються в областях розуміння мови, планування, діагностики та навчання. Багато систем штучного інтелекту також в дещо меншій мірі засновані на знаннях – наприклад, деякі ігрові системи і системи технічного зору високого рівня [5].

Ключовим фактором для подання знань у інтелектуальних інформаційних системах є те, які знання зберігаються у таких системах, тому нижче наведені знання, які як правило зберігаються в інтелектуальних інформаційних системах:

– знання про об'єкти: всі факти про об'єкти в нашій області світу. Наприклад, гітари містять струни, а труби – це мідні духові інструменти;

– події: події – це дії, які відбуваються в нашому світі;

– виконання : описує поведінку, яке передбачає знання того, як щось робити;

– мета-знання: це знання про те, що ми знаємо;

– факти: факти – це правдиві знання про реальний світ і про те, що ми представляємо.

Знання реального світу грають життєво важливу роль в інформаційних системах. Знання відіграють важливу роль в демонстрації розумної поведінки в роботі інтелектуальних інформаційних систем. Система може точно діяти засновуючись на деякому вводі, тільки якщо у неї є деякі знання або досвід щодо цього вводу. Припустимо, якщо ви зустріли людину, що говорить на незнайомому вам мовою, то як ви

зможете з цим діяти. Людина може діяти відчуваючи навколишнє середовище і використовуючи знання. Але якщо частина знань не буде представлена тоді, вона не зможе відображати розумну поведінку. Те ж саме стосується розумної поведінки інтелектуальних систем.

Інтелектуальні системи мають наступні компоненти для відображення інтелектуальної поведінки, наведені на рисунку 1.1:

– сприйняття (perception);

– навчання (learning);

– подання знань і міркування;

– планування (planning);

– виконання (execution).

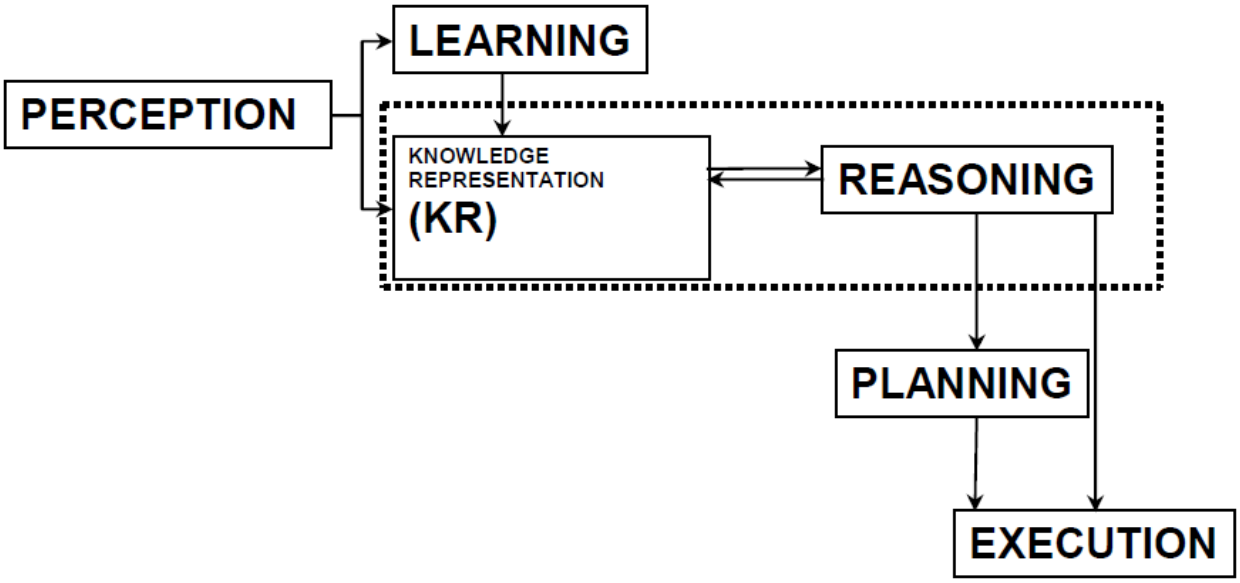

Рисунок 1.1 – Компоненти інтелектуальної системи

На рисунку вище показано, як інтелектуальна система може взаємодіяти з реальним світом і які компоненти допомагають їй виявляти інтелект. В системі є компонент сприйняття (perception), за допомогою якого вона видобуває інформацію з навколишнього середовища. Компонент навчання (learning) відповідає за навчання на основі даних, отриманих за допомогою сприйняття. У повному циклі основними

компонентами є подання знань і міркування (knowledge representation and reasoning). Ці два компоненти є незалежними один від одного, але також пов'язані один з одним. Планування (planning) і виконання (execution) залежать від аналізу подання знань і міркувань.

З наведеного вище можна зробити висновок, що подання знань грає важливу роль в роботі інтелектуальних систем, бо надає системі доступ до знань реального світу та можливостям для їх обробки. Таким чином, можна сформувати основні вимоги до систем, в яких використовується подання знань:

– репрезентативна точність. Система KR повинна мати здатність представляти всі види необхідних знань;

– адекватність виведення. Система KR повинна мати можливість маніпулювати репрезентативними структурами для отримання нових знань, відповідних існуючій структурі;

– ефективність виведення. Здатність направляти механізм висновків в найбільш продуктивних напрямах шляхом зберігання відповідних підказок;

– ефективність придбання. Здатність легко здобувати нові знання за допомогою автоматичних методів.

1.3 Методи подання знань

Подання знань – це область штучного інтелекту, призначена для представлення інформації про світ у формі, яка комп'ютерна система може використовуватись для вирішення різноманітних завдань базуючись на тому, які рішення приймали б люди для вирішення такої задачі. Комп’ютерні додатки засновані на знаннях використовують подання знань для збору інформації про світ та, яка може використовуватися для вирішення складних проблем.

Обгрунтування застосування подання знань полягає в тому, що звичайний процедурний код – не найкращий формалізм для вирішення складних проблем. Подання знань спрощує визначення та обслуговування складного програмного забезпечення, ніж процедурний код, і може використовуватися в експертних системах.

Існують різноманітні методи для представлення знань [6]. Прикладами таких методів виступають логічне представлення, фреймові моделі, продукційні правила та семантичні мережі. Нижче наведено опис кожного з цих підходів.

Першим з таких підходів є логічне уявлення. Знання і логічні міркування відіграють величезну роль в інтелектуальних системах. Однак для забезпечення розумного поведінки часто потрібно щось більше, ніж просто загальні і потужні методи. Формальна логіка – найкорисніший інструмент в цій області.

Логічне уявлення – це мова з деякими конкретними правилами, яка має справу з твердженнями і не має двозначності в поданні. Логічне уявлення означає висновок, заснований на різних умовах. Це уявлення встановлює деякі важливі правила спілкування. Ця мова складається з чітко визначеного синтаксису і семантики. Кожне твердження можна перевести в логіку, використовуючи синтаксис і семантику. В мові логічних уявлень семантика означає правила, за якими можна інтерпретувати твердження в логіці. У свою чергу синтаксис – це правила, які визначають, як ми можемо будувати правильні твердження в логіці.Різні правила логіки дозволяють вам представляти різні речі, що призводить до ефективного висновку. Отже, знання, отримане логічними агентами, буде певним, що означає, що воно буде або правдивим, або хибним. Хоча працювати з логічним поданням складно, воно становить основу мов програмування і дозволяє будувати логічні міркування.

Перевагами логічних моделей є те, що вони є основою мов програмування та дозволяють створювати логічні міркування. Недоліки

полягають в тому, що з логічними уявленнями складно працювати та такий підхід має свої обмеження. Також логічне уявлення не є природним поданням знань, а логічний висновок с такої моделі не є дуже ефективним.

Фреймове представлення знань – це набір атрибутів і пов'язаних з ними значень, який визначає сутність в реальному світі. Це структура, подібна запису, що складається з слотів і їх значень. Слоти можуть бути різних розмірів і типів. У цих слотів є імена і значення. Або у них можуть бути підполя, названі фасетами. Вони дозволяють накладати обмеження на рами. Фасети – це особливості фреймів, які дозволяють нам накладати обмеження на фрейми. Приклад: факти IF-NEEDED викликаються тільки коли потрібні дані будь-якого конкретного слота. Кадр може складатися з будь-якої кількості слотів, а слот може включати в себе будь-яку кількість фасетів, а фасети можуть мати будь-яку кількість значень. Кадр також відомий як уявлення знань слот-фільтром в штучному інтелекті. Фрейми є похідними від семантичних мереж і пізніше перетворилися в сучасні класи і об'єкти. Окремий фрейм не особливо корисний. Система фреймів складається з набору зв'язаних фреймів. У фреймі знання про об'єкт або подію можуть зберігатися разом в базі знань. Приклад фрейма наведено нижче (рис. 1.2).

Slots	Filter
Name	Peter
Profession	Doctor
Age	25
Marital status	Single
Weight	78

Рисунок 1.2 – Приклад фрейму

Перевагами фреймової моделі є простота використовування за рахунок зв’язаних даних. Також таке уявлення є дуже простим для розуміння та візуалізації, а самі фрейми можна легко розширювати. Але фреймова модель має занадто узагальнений підхід. Через це складно розробити механізм логічного висновку за допомогою фреймової моделі.

Подання, засноване на продукційних правилах, має багато властивостей, необхідних для подання знань. Воно складається з виробничих правил, робочої пам'яті і циклу розпізнавання-дії. Це також називається правилами «умова-дія». Згідно поточній базі даних, якщо умова правила правдива, виконується дія, пов'язана з правилом. У продукційних правилах агент перевіряє умову, і якщо умова існує, то спрацьовує продукційне правило і виконується відповідна дія. Умовна частина правила визначає, яке правило може бути застосовано до проблеми. І частина дій виконує відповідні кроки по вирішенню проблем. Цей повний процес називається циклом розпізнавання і дії. Робоча пам'ять містить опис поточного стану вирішення проблем, і правила можуть записувати знання в робочу пам'ять. Ці знання збігаються і можуть призвести до спрацьовування інших правил. Якщо виникає нова ситуація (стан), то кілька виробничих правил будуть запущені разом, це називається набором конфліктів. У цій ситуації агенту необхідно вибрати правило з цих наборів, і це називається вирішенням конфлікту. Хоча продукційним правилам не вистачає точної семантики для правил і вони не завжди ефективні, правила призводять до більш високого ступеня модульности. І це найбільш виразна система представлення знань.

Перевагами продукційних правил є те, що вони виражені природною мовою та є дуже модульними, тому їх легко видалити, додати або змінити. У свою чергу основними недоліками такого підходу є неможливість навчання за допомогою такої моделі, бо в продукційних правилах не зберігається результат завдання. Також під час виконання програми багато

правил можуть бути активними, тому виробничі системи на основі правил неефективні.

Семантична мережа дозволяє зберігати знання у вигляді графічної мережі з вузлами і дугами, що представляють об'єкти і їх відносини. Він може представляти фізичні об'єкти, концепції або навіть ситуації. Він також використовується для підтримки концептуального редагування і навігації. У семантичних мережах можна представити знання у вигляді графічних мереж. Ця мережа складається з вузлів, що представляють об'єкти, і дуг, які описують відносини між цими об'єктами. Семантичні мережі прості для розуміння і легко розширюються. Також семантична мережа є простою в реалізації. Це модель більш природно представляє знання . Така модель дозволяє класифікувати об'єкти в різних формах, а потім пов'язувати ці об'єкти. Вона також має більшу виразністю, ніж логічне уявлення. Приклад семантичної мережі наведено на рисунку 1.3.

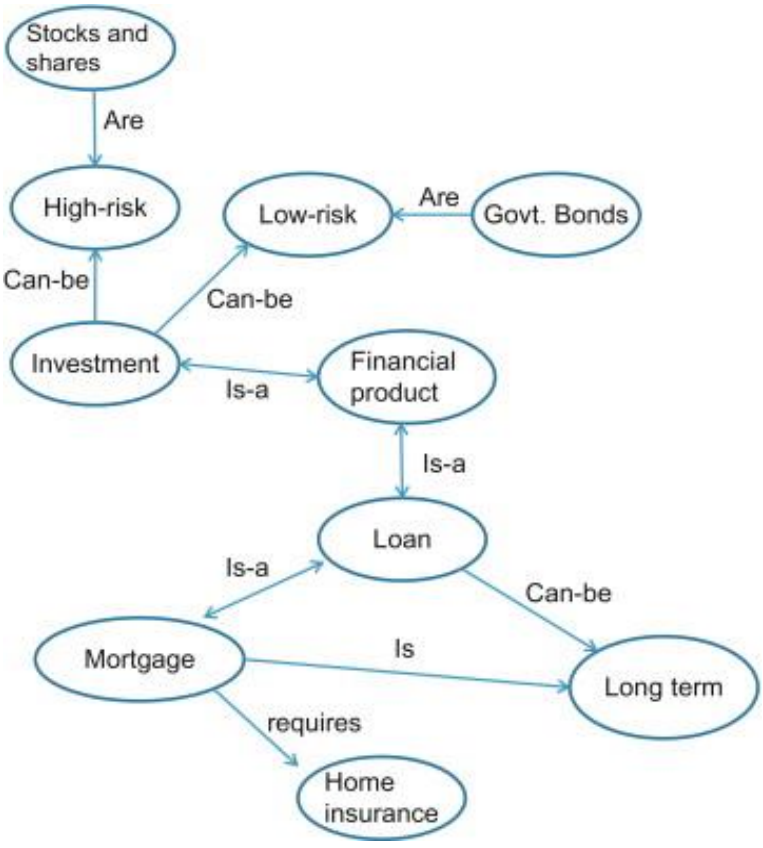

Рисунок 1.3 – Семантична мережа

На рисунку вище різні типи знань представлені у вигляді вузлів і дуг. Кожен об'єкт пов'язаний з іншим об'єктом деяким відношенням. Однак семантичні мережі вимагають більше обчислювального часу під час виконання, оскільки потрібно пройти повне дерево мережі, щоб відповісти на деякі питання. Однак незважаючи на значний недолік Семантичні мережі є природним уявленням знань. Така модель прозоро передає сенс знання, тому що вона є простою для розуміння.

Розвитком концепції подання знань у вигляді семантичних мереж стали концептуальні графи (conceptual graph, CG), запропоновані як мережева мова для моделювання семантики природної мови [7]. Концептуальний граф – це кінцевий, зв'язний, двочастковий граф, вузли якого містять поняття або концептуальні відношення. В таких графах не використовуються мітки дуг, а відношення між поняттями представляються вузлами концептуальних відношень. Основним призначенням CG є представлення смислу природно-мовних речень засобами концептуальних графів. На цей час існують добре обґрунтовані та детально описані правила формування CG та маніпулювання ними. Важливою перевагою CG є можливість перекладу, тобто трансляції, концептуальних графів у представлення, тобто формули, логіки предикатів першого порядку. Таким чином, можна досягти подання знань з певної ситуації або предметної області у вигляді суворої математичної моделі. Але на практиці цей переклад занадто складний, потребує спеціального математичного та програмного забезпечення і, можливо тому, не отримав широкого практичного застосування.

1.4 Подання знань у Web-просторі

Стрімке поширення Інтернет та нові можливості обробки, обміну та сумісного повторного використання великих обсягів інформації задля вирішення різноманітних задач в різних предметних галузях сприяло появі

та розвитку нових типів систем, призначених для функціонування у Web-просторі. Такі інтелектуальні інформаційні системи базуються на технологіях Semantic Web та онтологічному підході до організації та та використанні знань. Вони отримали загальну назву Web-базованих онтологічних систем. Основою систем такого типу є принципово нова концепція формування знань щодо інформаційних ресурсів у вигляді взаємопов'язаних онтологічних моделей, та управління ресурсами на основі отриманих знань. Семантичні компоненти онтологічної моделі представляються у вигляді екземплярів (instances), концептів (concepts), їх атрибутів, відношень між концептами (relationships), можливих обмежень, що накладаються на концепти та/або їх атрибути (constraints). На основі онтологічних моделей формується онтологічна база знань, яка забезпечує експліцитну специфікацію концептуальної моделі предметної області у вигляді ієрархічної структури взаємопов'язаних понять і термінів, релевантних ПрО, а також інформаційних ресурсів, пов'язаних із системою [8].

Основою формального представлення ІІС є онтологічні моделі ПрО та її складових частин. Онтологічна модель представляється у вигляді структури, яка складається з множини концептів, релевантних ПрО, їх атрибутів і властивостей, встановлених на множині концептів. Для побудови узагальненої моделі ІІС пропонується підхід на основі інтеграції знань, формалізованих у вигляді окремих онтологічних моделей, відповідних до складових частин ІІС [9]. Пропонований «bottom-up» підхід дозволяє побудувати узагальнену модель ІІС із максимальним урахуванням його специфіки. Задля реалізації запропонованого підходу розробляються та розвиваються методи співставлення, відображення, вирівнювання онтологічних моделей з метою інтеграції знань про ІІС на різних стадіях його життєвого циклу [10]. Особлива увага при цьому приділяється аналізу та класифікації технологій співставлення онтологій як найбільш трудомісткому етапу щодо підготовки їх подальшого

узгодження. Одним з можливих підходів до класифікації таких технологій може бути підхід на основі двох ортогональних вимірів. При цьому горизонтальний вимір включає три рівня, які будуються один поверх іншого. Перший рівень – це рівень даних. Співставлення сутностей виконується шляхом порівняння тільки значень даних для простих або складних типів даних. Другий рівень – онтологічний, він, в свою чергу, поділяється на чотири шари (відповідно до метафори «листкового пирога» Semantic Web). Ці шари знизу вгору становлять відповідно семантичні мережі, дескрипційні логіки, обмеження та правила. На рівні семантичних мереж онтології розглядаються як графи, вершини яких відповідають концептам, а дуги – відносинам між концептами. Співставлення здійснюється тільки шляхом порівняння відповідних дуг та вершин. Шар дескрипційних логік привносить до онтологій врахування формальної семантики. Співставлення може включати, наприклад, визначення таксономічної схожості, яке базується на кількості відношень категоризації (subsumption), що розділяють два концепти. У цьому ж шарі при співставленні враховуються екземпляри класів (інстанси). Так, наприклад, концепти оцінюються як однакові, якщо їх інстанси схожі. Співставлення на рівні обмежень та правил зазвичай базуються на ідеї про те, що якщо між сутностями існують схожі правила, то ці сутності можуть розглядатися як схожі. При цьому зазвичай потрібна обробка відношень більш високого порядку. Контекстний шар пов’язаний із практичним використанням сутностей в контексті їх програмних додатків. Співставлення в цьому випадку виконується шляхом порівняння використаних сутностей в онтобазованих додатках. Вважається, що схожі сутності часто використовуються у схожих контекстах. Вертикальний вимір класифікації методів співставлення онтологій представляє знання, що відображають специфіку ПрО. Такі знання можуть розташовуватися на будь-якому рівні горизонтального виміру. Тут можуть бути використані переваги зовнішніх джерел для опису специфічних знань.

Ще одним підходом до подання знань у Web-просторі є графи знань [11]. Цей підхід стає все більш актуальним та популярним за останні роки. Граф знань (knowledge graph, KG) – це семантичний граф, що складається з вершин (або вузлів) і ребер. Вершини представляють концепції або сутності. Поняття відноситься до узагальнених категорій об'єктів, таких як вчений, автомобіль тощо. Сутність – це фізичний об'єкт в реальному світі, такий як людина, місце розташування і організація. Ребра представляють семантичні відносини між концепціями або об'єктами. Використовуючи графи знань, фрагментовані, значить, частково спостережувані об'єкти та концепції, можна з'єднати разом, щоб сформувати повне та структуроване сховище знань, що полегшує управління, виведення, використання та розуміння змісту інформації. Граф знань – це теж свого роду графова модель. Однак на відміну від семантичних мереж така модель фокусується на зберіганні всіх даних як на рівні схеми, так і на рівні окремих сутностей. На мою думку саме така модель є найкращим вибором для подання знань у інформаційних інтелектуальних системах, тому що саме графи знань можуть надати такі ключові переваги для роботі зі знаннями :

– описи мають формальну семантику, яка дозволяє як людям, так і комп'ютерам обробляти їх ефективним і однозначним чином;

– описи сутностей доповнюють один одного, утворюючи мережу, де кожна сутність являє собою частину опису пов'язаних з нею сутностей і забезпечує контекст для їх інтерпретації.

Саме ці фактори грають ключову роль в роботі інтелектуальних інформаційних систем. Завдяки вищеописаним перевагам в інтелектульних інформаційних системах можна організувати зв'язок між людиною та комп'ютером, а також надати можливість для найефективнішої роботи механізмуміркувань. Через це графи знань були обрані у роботі як основна модель для подання знань.

1.5 Постановка задач дослідження

Метою даної кваліфікаційної роботи є дослідження та використання методів подання знань за допомогою графів знань та методів обробки природно-мовних текстів.

З урахуванням сформульованої мети в даній кваліфікаційній роботі необхідно вирішити низку таких задач:

– провести аналіз науково-технічних публікацій та даних Інтернет-джерел в області сучасних підходів до подання знань задля побудови Web-базованих інтелектуальних інформаційних систем;

– провести порівняльний аналіз основних відомих моделей подання знань, які використовуються в ІІС;

– обґрунтовано здійснити вибір моделі та методів подання знань для проведення досліджень;

– здійснити аналіз основних підходів та відомих проектів до побудови графів знань задля вирішення проблеми подання знань у Web-просторі;

– провести аналіз основних задач та методів NLP з метою потенційного їх застосування задля побудови графів знань;

– здійснити вибір та формування наборів даних для експериментальних досліджень;

– провести огляд методів видобування сутностей та відносин з тексту та проаналізувати, які методи переважають, виділити їх переваги та недоліки;

– провести обґрунтований вибір методів видобування сутностей, відносин, тегування та токенізації та визначити, які з цих підходів будуть найкращими для побудови графів знань;

– здійснити вибір програмних засобів для перевірки працездатності побудованого програмного додатку;

– здійснити розробку пілотної версії програмного додатку для побудови графів знань на підставі методів NLP;

– здійснити дослідження можливостей розширення функціоналу програмного додатку.

У подальших розділах будуть детально роглянуті графи знань як сучасний підхід до подання знань, методи обробки природної мови та можливості їх застосування для побудови графів знань, практичне застосування отриманих результатів досліджень за допомогою розробленого програмного додатку.

2 ГРАФИ ЗНАНЬ

2.1 Сучасний стан та області застосування графів знань

Інформація в Інтернеті фрагментована і представлена в різних джерелах даних, що робить автоматичний збір і розуміння знань складним завданням для машин і навіть людей. Графи знань стали переважати як в промисловості, так і в академічних колах в ці роки, ставши одним з найбільш ефективних і дієвих підходів до інтеграції знань. Методи побудови графа знань можуть отримувати інформацію з структурованих, частково структурованих або навіть неструктурованих джерел даних і, нарешті, інтегрувати інформацію в знання, представлені у вигляді графу. Крім того, граф знань може впорядковувати інформацію в простій в обслуговуванні, зрозумілій та зручній формі.

Існує три різних перспективи, що характеризують використання графів знань та відповідні погляди на графи знань. Хоча можливо безліч способів класифікації типів графів знань, використовуваних в літературі, можна виділити три основні точки зору [12]:

– інструменти подання знань: де основна увага приділяється тому, як мережа знань використовується для представлення деякої форми знань;

– системи управління знаннями: де основна увага приділяється системі управління мережею знань, аналогічно тому, як системи управління базами даних грають цю роль для баз даних;

– служби додатків знань: де основна увага приділяється забезпеченню рівня додатків поверх мережі знань.

Хоча ці три види застосування графів, безумовно, мають незалежну цінність, більшість інтелектуальних систем використовує графи знань об'єднуючи способи їх застосування у вигляді шарів: на першому рівні знаходиться подання знань, на середньому рівні – система управління цими знаннями, а на верхньому рівні – додаток, який використовую та

обробляє ці знання. Дана модель використання графів знань представлена нижче (рис. 2.1).

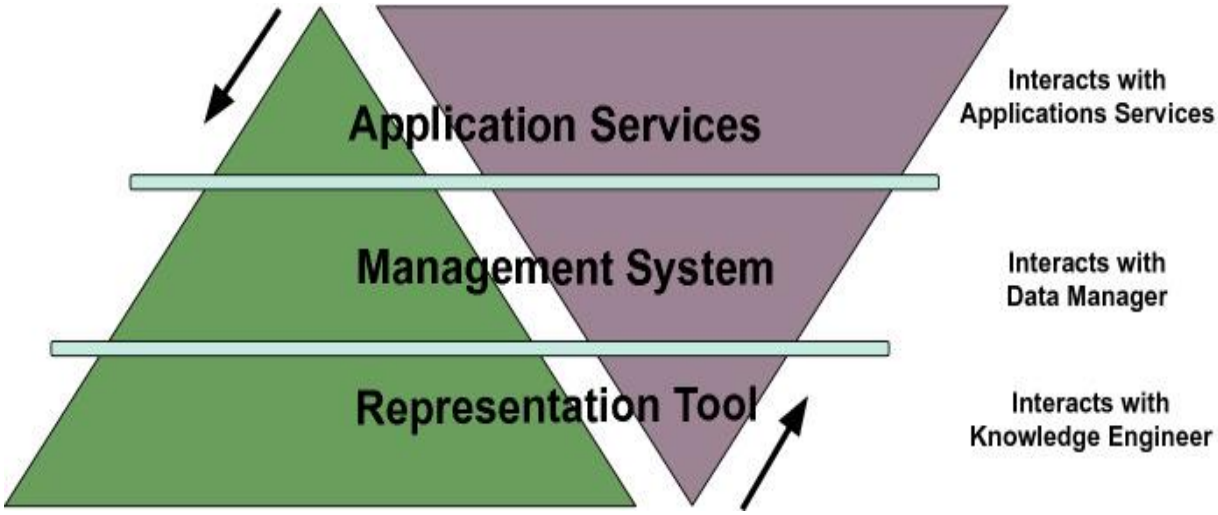

Рисунок 2.1 – Застосування графів знань

Зазвичай є два способи подивитися порядок розташування цих шарів. Деякі спільноти схильні розглядати його зверху вниз з додатком, яке вирішує KG, в інших – від низу до верху, з поданням знань як основного. Цікаво, що є навіть ще один, оскільки спільнота з управління даними часто бачить в центрі уваги систему управління посередині. Межі між цими шарами нечіткі. Багато академічних та промислових систем охоплюють два або три з цих рівнів. У деяких випадках інструменти подання частково виконують деякі характеристики систем управління. Те ж саме стосується додатків, що працюють безпосередньо зі знаннями. Звичайно, очевидно, що для створення відмінної інформаційної системи в цілому необхідно враховувати всі рівні і їх взаємодію; навряд чи можливо забезпечити хороший програмний додаток, що буде працювати зі знаннями, якщо рівень подання знань не підходить для цієї мети.

Графи знань часто використовуються як інструменти подання знань. Це пов’язано з тим, що великий обсяг людських знань можна представити у вигляді багатозв'язного графа. Бінарні відносини кодують факти, які

можуть бути представлені у вигляді трійок типу RDF (голова; предикат; хвіст), де голова і хвіст – це сутності, а предикат – це тип відносини. Графи знань надають способи ефективної організації, управління і вилучення цього типу інформації і все частіше використовуються в якості зовнішнього джерела знань для таких проблем, як рекомендаційні системи, мовне моделювання, відповіді на питання чи класифікація зображень. Один критичний момент, який слід підкреслити, полягає в тому, що, хоча багато графів знань, на сьогоднішній день, містять в якості своїх знань в основному прості наземні дані, все більше і більше додатків вимагають дієвого представлення знань. Певною мірою це вже відноситься до існуючих систем управління базами знань, підтримуваним онтологіями, для яких завдання міркування мають різну обчислювальну складність і виразну потужність. Важливість підтримки неявних знань також стає центральною для графів знань, особливо коли вони є компонентом інтелектуальних корпоративних додатків, до такої міри, що інтенсіональні знання слід розглядати як частину самого графу. Отже, міркування, тобто перетворення інтенсіональні в похідне базове знання, стає невід'ємною частиною визначення графів знань. Наприклад, у фінансовому додатку Enterprise AI, сукупність нормативних знань і правила функціонування конкретної фінансової області мають істотне значення. Як інший приклад, в умовах логістики знання того, як взаємодіють певні етапи ланцюжка поставок, часто більш важливо, ніж чисті дані, що лежать в основі ланцюжка поставок. Можна навести ще багато таких прикладів.

В цілому очевидно, що в сучасних системах на основі графів знань необхідно враховувати і належним чином обробляти багате подання знань, щоб врівноважити збільшену складність з багатьма іншими важливими властивостями, включаючи зручність використання, масштабованість, продуктивність і надійність програмного додатку.

Також графи знань часто використовують як інструменти для управління знаннями. Зокрема ця тенденція сформувалася тому, що графи

знань можуть забезпечувати підтримку для користувача, щоб додати знання в граф знань, отримати нові знання, використовуючи існуючі знання, і отримати дані за допомогою універсальної мови запитів. В інтелектуальних інформаційних системах мотивація наявності графів знань виражається в постановці складних запитів до ширшого набору інтегрованої інформації з різних джерел для виявлення знань і поглибленого аналізу.

Також існує величезний обсяг робіт, в яких KG розглядаються не як інструменти подання знань або системи управління, а як платформа для надання великої кількості найважливіших додатків. Таким чином, замість того, щоб використовувати KG для подання інформації або управління інформацією, це, скоріше, здатність KG спочатку легко підтримувати певні програми, які визначають, що таке KG.

Наприклад, в KG може розглядатися не тільки як граф, що містить всі дані про продукти Amazon, але і як граф, що володіє особливою здатністю спочатку підтримувати дозвіл сутностей (тобто знати, коли два продукти є однаковими) і зв'язування сутностей (тобто знання того, коли два продукти або інші об'єкти пов'язані). Подібні міркування можна знайти в багатьох областях, пов'язаних з графами знань.

Таким чином можна зробити висновок, що графи знань – це модель, що активно використовуються для великої кількості різноманітних цілей та представляє собою дуже потужний інструмент для широкого спертку задач від зберігання знань у зручній та природній формі до маніпуляції цими знаннями в різноманітних системах.

Використання графів знань – це розумний приклад в епоху великих даних, який рухається як вимогами галузі, так і дослідницької мотивацією з боку академічних кіл. В даний час всі основні пошукові системи, такі як Google і Bing, використовують цю модель для надання спеціальних відповідей на пошукові запити, такі як «дружина президента», де на запит можуть відповідати безпосередньо об'єкти, а не набір відповідних веб-

запитів. документи, що містять багато надлишкової інформації. Графи знань можуть робити висновки про концептуальне значення запитів і визначати завдання користувача у веб-пошуку , що може призвести до більш точних пропозицій за запитом [13]. Знання х графу знань дозволяють машинам розуміти текст на природній мові і напівструктуровані веб-таблиці. Успішне будівництво та інтеграція графів знань принесе інтелектуальну функціональність в різні додатки. Завдяки наявності детального високоякісного графу знань, пошукова система може зрозуміти такий запит, як «дружина президента», і негайно отримати відповідь, просто розпізнавши «президент» як текст, в якому згадується сутність з іменем президента і негайно знайти ім'я об'єкта, який має зв'язок «є дружиною» з цією сутністю. Крім додатків, пов'язаних з пошуком в Інтернеті, графи знань можуть служити джерелом даних для створення семантичної бази даних, яка може автоматично використовуватися комп'ютерами, наприклад онлайн-Вікіпедії зі структурованими даними в строгому форматі. Це дозволяє інтелектуальним системам відповідей на питання відповідати на питання, задані людьми та комп'ютерами, на природних мовах або мовах, схожих на запити.

Крім того, графи знань можуть служити сховищем структурованих знань, які підтримують велику кількість додатків, пов'язаних з аналітикою великих даних. У компаніях електронної комерції, таких як Walmart , пошук продуктів і рекомендації можуть підтримуватися товарними кілограмами, які містять продукти, продавців, виробників і т.д. для поліпшення медичного діагнозу. Розглянемо декілька прикладів використання графів знань для виконання широкого спектру задач.

Проект Cyc і OpenCyc використовує граф знань Cyc – один з найстаріших графів знань, що відноситься до 1980-х років. Заснований на традиційних дослідженнях в області штучного інтелекту, це ретельно підібраний граф знань, розроблений і підтримуваний CyCorp Inc.

OpenCyc – це скорочена версія Cyc, яка є загальнодоступною. Також існує кінцева точка семантичного Інтернету для OpenCyc, що містить посилання на DBpedia і інші набори даних LOD. OpenCyc містить приблизно 120 000 примірників і 2,5 мільйона фактів, визначених для цих примірників. Також його схема включає ієрархію типів приблизно з 45 000 типів і 19 000 можливих відносин.

Freebase, велика онлайн коллоборативна база знань, яка містила в основному метадані, зібрані інтернет-спільнотою, на сьогодні є складовою частиною бази знань Knowledge Graph компанії Google, однієї з представників великої п’ятірки в області штучного інтелекту FAANG (Facebook, Amazon, Apple, Netflix, Google). Створення універсального графа знань – завдання, яке є нездійсненним для більшості людей і організацій. На сьогоднішній день більше 900 людино-років було витрачено на створення Cyc , але все ще існують прогалини. Таким чином, розподіл цих зусиль на максимально можливу кількість плечей за допомогою краудсорсингу – це спосіб, обраний Freebase, загальнодоступним редагованим графом знань з шаблонами схем для більшості типів можливих об'єктів (наприклад, людей, міст, фільмів тощо Після того, як MetaWeb, компанія, керуюча Freebase, була придбана Google, Freebase була закрита 31 березня 2015 року.

Остання версія Freebase містить близько 50 мільйонів сутностей і 3 мільярди фактів. Схема Freebase включає приблизно 27 000 типів сутностей і 38 000 типів відносин.

Проект DBpedia, по суті своїй – це граф знань, який сформовано зі структурованих даних у Вікіпедії. Основним джерелом цього формування є пари ключ-значення з інформаційних скриньках Вікіпедії. В процесі краудсорсингу типи інформаційних скриньок зіставляються з онтологією DBpedia, а ключі використовувані в цих інформаційних блоках зіставляються з властивостями в цій онтології. На основі цих представлень можна сформувати граф знань.

Остання версія DBpedia (тобто DBpedia 2015-04, видобута з англійської Вікіпедії на основі дампов за лютий / березень 2015 р.) містить 4,8 мільйона об'єктів і 176 мільйонів тверджень про ці об'єкти. Онтологія включає 735 класів і 2800 відносин.

Проект NELL (Never Ending Language Learning), у якому на відміну від DBpedia, де використовується частково структурований контент в якості основи, запропоновані методи видобування графів знань з неструктурованих даних. Проект NELL став одним з перших підходів, які працювали в веб-масштабі. Проект працює на великомасштабному корпусі веб-сайтів і використовує пов'язаний процес, який вивчає текстові шаблони, відповідні твердженнями типу і відносини, а також застосовує їх для вилучення нових сутностей і відносин. Процесс міркування в системі застосовується для перевірки несуперечності і видалення суперечливих аксіом. Хоча дані в NELL не публікувалися з використанням стандартів семантичної павутини, було показано, що вони можуть бути перетворені в RDF і надані також як пов'язані відкриті дані.

У своїй останній версії (тобто 945-й ітерації) NELL містить приблизно 2 мільйони сутностей і 433 000 відносин між ними. Онтологія NELL визначає 285 класів і 425 відносин.

Мережа знань Google.

Мережа знань Google була представлена публіці в 2012 році, коли сформувався термін «граф знань». Сам Google досить приховано відноситься до побудови своєї мережі знань. Є лише кілька зовнішніх джерел, які обговорюють деякі механізми потоку інформації в мережу знань на основі досвіду. Ґрунтуючись на цьому, можна припустити, що основні напівструктуровані веб-джерела, такі як Вікіпедія, вносять свій вклад в граф знань, а також в структуровану розмітку (наприклад, schema.org Microdata) на веб-сторінках і по змісту соціальної мережі Google – Google+. Мережа знань Google містить 18 мільярдів тверджень

про 570 мільйонах сутностей зі схемою з понад 500 типів сутностей і 35 000 типів відносин.

Враховуючи вищезгадане, можна зробити висновок, що за останні роки графи знань перетворилися в найважливіші будівничи компоненти для вирішення задач подання знань, їх організації і розуміння. У медійному та науковому просторі була представлена велика кількість досліджень, додатків і продуктів, пов'язаних з графами знань. Тим не менш, є ще багато проблем і можливості в області розвитку цієї моделі.

2.2 Загальна характеристика графів знань

У 1980-х роках дослідники з Університету Гронінгена і Університет Твенте в Нідерландах спочатку ввели термін «граф знань» для формального опису своєї системи, заснованої на знаннях, яка об'єднує знання з різних джерел для подання природної мови [13], [14]. Автори запропонували граф знань з обмеженим набором відносин, що буде зосереджений на якісному моделюванні, включаючи людську взаємодію, що явно контрастує з ідеєю графу знань, що була активною темою розмов в медійному та науковому просторі в останні роки. У 2012 році Google представив мережу знань як семантичне розширення функції пошуку Google, яка зіставляє рядки, але дозволяє шукати «речі», іншими словами, об'єкти реального світу. Хоча повідомлення в блозі не містить ніяких подробиць реалізації, за даними Google Scholar, воно цитувалася більше 100 разів. З 2012 року термін «граф знань» також використовується для опису сімейство додатків. Часто згадувалися реалізації – це DBpedia, YAGO (ще одна велика онтологія), Freebase, Wikidata, помічник по семантичного пошуку Yahoo Spark, сховище знань Google, Satori від Microsoft і граф сутностей Facebook . Ці додатки відрізняються один від одного за своїми характеристиками, таким як архітектура, операційне призначення і стек використаних технологій, що ускладнює досягнення

консенсусу і створення визначення графа знань. Крім того, більш конкретний термін «граф корпоративних знань» використовувався кількома невеликими компаніями, наприклад, SindiceTech і компанія Semantic Web. Обидві компанії прагнуть описати аналогічну модель, яка видобуває і зберігає різноманітні корпоративні дані в потрійному сховище і аналізує їх за допомогою методів машинного навчання, щоб отримати нові знання з даних і повторно використовувати їх в інших додатках. Через такий широкий спектр областей, де згадувався термін граф знань виникла плутанина в тому, що ж саме означає це поняття. Тому для отримання повної картини такої моделі знань, як граф знань, треба треба ретельно підійти до всіх особливостей, що характеризують цю модель.

Граф знань (KG) – це модель, що забезпечує значну добре структуровану інформацію для моделювання абстрактних концепцій, а також конкретних сутностей у реальному світі. Саме ця модель знань привернула велику увагу в останні роки, бо вона має добре формализований набір взаємозв'язаних описів сутностей – реальних об'єктів і подій або абстрактних концепцій (наприклад, документів).

Прийнято вважати, що термін «граф знань» був придуманий компанією Google для позначення використання ними семантичних знань в веб-пошуку («речі, а не рядки»), а недавно також використовується для позначення баз знань семантичних мереж, таких як DBpedia або YAGO. З більш широкої точки зору, будь-яке графічне представлення деяких знань може вважатися графом знань (це може включати будь-який вид набору даних RDF, а також онтології логіки опису). Однак немає єдиного визначення того, що таке граф знань і що ні. Замість спроби формального визначення того, що таке граф знань, можна обмежетися мінімальним набором характеристик графів знань, щоб відрізнити графи знань від інших наборів знань, які не можна розглядати як графи знань. Граф знань:

– в основному описує сутності реального світу і їх взаємозв'язку, організовані у вигляді графа;

– визначає можливі класи і відносини сутностей в схемі;

– дозволяє потенційно пов'язувати довільні об'єкти один з одним;

– охоплює різні тематичні області.

Перші два критерії чітко визначають, що графи знань представляють фактичні екземпляри, при цьому схема грає лише другорядну роль. Зазвичай це означає, що кількість операторів рівня екземпляра на кілька порядків більше, ніж кількість операторів рівня схеми. Навпаки, схема може залишатися досить поверхневою з невеликим ступенем формалізації. У цьому сенсі прості онтології без будь-яких екземплярів (такі як DOLCE) не розглядатимуться як графи знань. Точно так само WordNet не розглядається як граф знань, оскільки він в основному пов'язаний з загальними іменниками і словами і їх відносинами (хоча кілька назв, тобто екземплярів, також включені). Третій критерій вводить можливість визначати довільні відносини між екземплярами, які не обмежені в їх області та діапазоні. Це властивість, яке навряд чи можна знайти в реляційних базах даних, які дотримуються суворої схеми. Більш того, графи знань повинні охоплювати принаймні більшу частину доменів, існуючих в реальному світі, і не повинні обмежуватися тільки одним доменом (наприклад, географічними об'єктами). У цьому сенсі великі, але однодоменних набори даних, такі як, наприклад, GeoNames, не вважатимуться графом знань.

Робоче визначення «Графу знань» – це об'єкти, властивості і відносини, що зберігаються в базі знань, що має выгляд, графу у вигляді вузлів і ребер [15]. В такій інтерпретації граф знань представляється як граф екземплярів, призначених для накопичення і передачі знань про реальний світ, вузли якого представляють об'єкти, а ребра представляють відносини між цими об'єктами. Граф знань відповідає моделі на основі графа, яка може бути орієнтованим графом з мітками ребер, графом властивостей тощо. Під знанням мається на увазі те, що відомо. Такі знання можуть бути отримані із зовнішніх джерел або вилучені з самого

графа знань. Знання може складатися з простих тверджень, таких як «Сантьяго – столиця Чилі», або кількісних тверджень, таких як «все столиці – міста». Прості інструкції можуть накопичуватися у вигляді ребер в графі даних. Якщо граф знань має намір накопичувати кількісні затвердження, потрібно більш виразний спосіб представлення знань – наприклад, онтології або правила. Потім можна використовувати дедуктивні методи для отримання і накопичення додаткових знань (наприклад, «Сантьяго – це місто»). Додаткові знання, засновані на простих або кількісних твердженнях, також можуть бути вилучені з графа знань і накопичені за допомогою індуктивних методів.

Графи знань часто збираються з безлічі джерел і, як наслідок, можуть бути дуже різноманітними з точки зору структури і деталізації. Щоб усунути цю різноманітність, представлення схеми, ідентичність і контекст часто грають ключову роль, де схема визначає структуру високого рівня для графа знань, ідентичність означає, які вузли в графі (або в зовнішніх джерелах) відносяться до однієї і тієї ж сутності реального світу, в той час як контекст може вказувати на конкретну настройку, в якій деяка одиниця знання вважається дійсною. Як згадувалося вище, для росту і поліпшення графа знань з часом потрібні ефективні методи вилучення, збагачення, оцінки якості та уточнення. Для подальшого розширення графу, знання, тобто сутності, властивості і відносини, добуті з джерел даних, таких як текст, голос, зображення тощо можуть бути структуровані як вузли та ребра і завантажені в бази даних графів з мінімальним опором на відміну від реляційних баз даних.

В типовому графі знань дані, що мають велику кількість відносин з різними сутностями, впорядковуються в формі потрійних фактів (head entity, relation, tail entity), які скорочуються як (h, r, t) [16]. Подання знань у графі знань можно представити на простому прикладі (рис. 2.2).

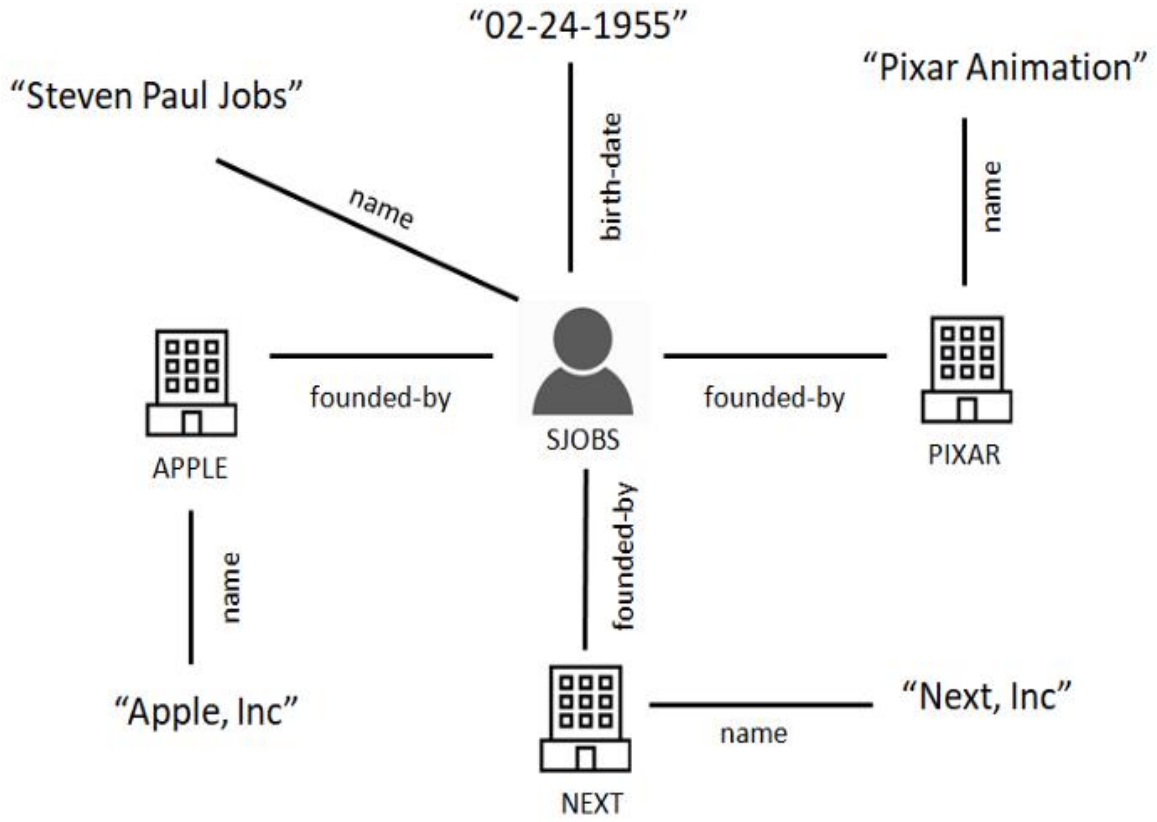

Рисунок 2.2 – Подання знань у графі знань

У прикладі наведеному на рисунку вище сутностями є SJOBS, NEXT, APPLE, PIXAR. Як видно на прикладу сутностями можуть бути будь які об’єкти реального світу. Властивості та відношення (дата народження, засновник, ім'я) пов'язують сутності між собою. У свою чергу значення (02-24-1955», «Next, Inc.», «Стівен Пол Джобс», «Apple, Inc», «Pixar Animation») несуть у собі інформацію про про об’єкти реального світу.

Роблячи висновки з прикладу вище у KG знання представлені у вигляді набору сутностей, їх властивостей і відносин між ними. Разом вони складають концептуальну, або семантичну модель проблемної області («моделі світу»). Сутності, їх властивості, а також відносини між виявленими сутностями утворюють граф вузлів і ребер, що робить структуру графа природним поданням знань. Знання, структуроване в цій формі, є ефективним способом організації людського мислення, воно

пояснює передбачуваний сенс і підтримує легку інтеграцію «знань», а графічна абстракція робить знання легкими для розуміння, як людиною, так і комп'ютером.

Аналізуючи поточну дослідницьку роботу, яка визначає чи розглядає графи знань, можна виявити дві фундаментальні проблеми: (а) запис в блозі Google про їх Граф знань цитується так, як якщо б вона дає правильне пояснення для побудови графа знань, і (б) терміни «граф знань» і «база знань» взаємозамінні [17] . Друга проблема призводить до помилкового припущення, що термін «граф знань» є синонімом бази знань, яка сама по собі часто використовується як синонім онтології. Прикладом такої плутанини є те, що і в сховище знань, і в графі знань Google їх творці назвали великомасштабною базою знань. Ще один приклад – YAGO, який – згідно своїй назві – онтологія, але називається як базою знань, так і графом знань.

Точно так як співробітники Yahoo не проводять чіткого відмінності між базою знань, графом знань і онтологією. Вони заявляють, що створюють свою базу знань, погоджуючи нові суті, відносини і інформацію з їх загальної онтологією. Таким чином, неповна, непослідовна і, можливо, неточна інформація перетворюється в багатий, уніфікований і однозначний граф знань. Грунтуючись на цій інформації, їх розуміння графа знань – це очищена база знань, яка представляє собою сукупність (наприклад, екземпляри) їх онтології. Для того щоб розрізняти терміни, їх необхідно чітко пояснити.

Система, заснована на знаннях, використовує штучний інтелект для вирішення проблем і складається з двох частин: бази знань і механізму виведення. База знань – це набір даних з формальної семантикою, який може містити різні види знань, наприклад правила, факти, аксіоми, визначення, затвердження і примітиви. Таким чином, Knowledge Vault можна класифікувати як справжню базу знань, тому що вона розширює

ідею чистого семантичного сховища за допомогою можливостей міркувань і, отже, більше схожа на систему, засновану на знаннях.

Онтологія – це формальна явна специфікація загальної концептуалізації, яка характеризується високою семантичної виразністю, необхідною для підвищеної складності задач . Онтологічні уявлення дозволяють семантичне моделювання знань і тому зазвичай використовуються в якості баз знань в додатках штучного інтелекту (ІІ), наприклад, в контексті систем, заснованих на знаннях. Застосування онтології в якості бази знань полегшує перевірку семантичних відносин і висновків з відомих фактів для виведення (тобто міркування). Явно підкреслюється, що онтологія не відрізняється від бази знань, хоча онтології іноді помилково класифікуються як знаходяться на тому ж рівні, що і схеми баз даних . Фактично, онтологія складається не тільки з класів і властивостей (наприклад, owl: ObjectProperty і owl: DatatypeProperty), але також може містити екземпляри (тобто сукупність онтології).

З одного боку, розмір часто згадується як важлива характеристика графів знань, тому KG можна описати як дуже велику онтологію. Однак відзначається, що графи знань в чомусь перевершують онтології і надають додаткові можливості. Таким чином, різниця між графом знань і онтологією можна інтерпретувати або як питання кількості (наприклад, велика онтологія), або як питання розширених вимог (наприклад, вбудованого засобу міркувань, яке дозволяє отримувати нові знання).

Друга інтерпретація призводить до припущення, що граф знань – це система, заснована на знаннях, яка містить базу знань і механізм міркувань. Зосередившись на існуючих автоматично згенерованих «графах знань», можна визначити ще одну важливу характеристику: збір, вилучення та інтеграція інформації з зовнішніх джерел розширює чисто засновану на знаннях систему концепцією систем інтеграції. Більшість додатків з відкритим вихідним кодом реалізують аспект інтеграції зі

зв'язаними даними. Рисунок 2.3 ілюструє комбінацію цих припущень що дає абстрактну архітектуру графа знань.

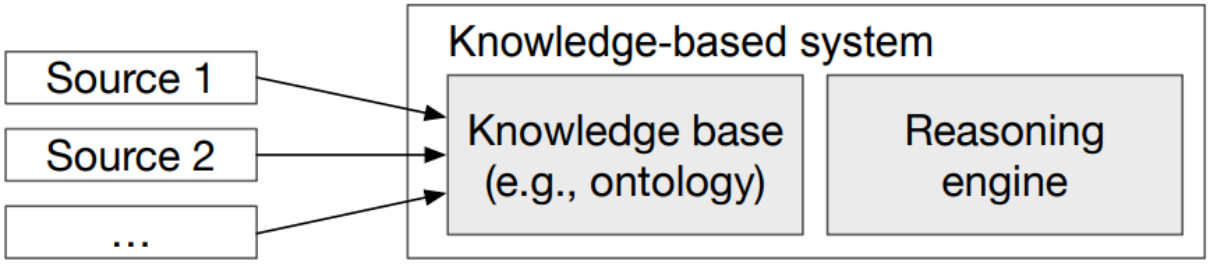

Рисунок 2.3 – Схема системи на основі графу знань

Грунтуючись на цій архітектурі і виведених з термінологічного аналізу, граф знань визначається наступним чином: граф знань збирає і інтегрує інформацію в онтологію і застосовує логіку міркувань для отримання нових знань.

Це визначення узгоджується з припущенням, що граф знань в деякому роді краще і складніше, ніж база знань (наприклад, онтологія), тому що він застосовує механізм міркувань для генерації нових знань і інтегрує один або кілька джерел інформації. Отже, створений вручну граф знань, який не підтримує аспекти інтеграції, є простою базою знань або системою, заснованою на знаннях, якщо вона забезпечує можливості міркувань. Визначення не бере до уваги кількісний аспект (розмір), особливо щодо великого ABox онтології, оскільки неясно, що можна вважати «великим». Замість цього здатності до міркування виділяються як важлива характеристика для отримання нових знань і диференціації KG від баз знань.

Крім того, виникає питання, в чому полягає різниця між семантичною мережею і графами знань. Менші KG, наприклад, графи корпоративних знань, можна чітко відрізнити від семантичної павутини через їх обмежену область. Завдання великих пошукових систем –

сканувати і обробляти всю доступну інформацію в мережі, що призводить до підвищеного інтересу до повсюдного впровадження семантичних технологій.

Хоча практично відсутня будь-яка інформація про технології, які застосовуються в Google Knowledge Graph і Microsoft Satori, але Yahoo Spark і у Knowledge Vault явно використовуються стандарти семантичної павутини, такі як RDF. Розглядаючи шари семантичної павутини, граф знань, для порівняння, розгортає або точно таку ж технологію для кожного рівня, або аналогічну технологію, яка пропонує ті ж функції. Наприклад, компанії можуть використовувати власні ідентифікатори замість URI і JSON-LD5 як формат серіалізації, що заміняє XML і RDF. Однак ці технології є лише прикладами, і, зокрема, на рівні синтаксису XML часто замінюється більш легкими форматами, а також фораматами, що легко зчитуються, такими як Turtle, N-Triples або N-Quads в співтоваристві Semantic Web. Наостанок, Семантичну мережу можна інтерпретувати як найбільш повний граф знань або, навпаки, граф знань, який переглядає всю мережу, можна інтерпретувати як автономну семантичну павутину.

На практиці графи знань (KG) активно застосовуються для підтримки певного класу програмних додатків, орієнтованих на домен або, в більш загальному сенсі, на підприємство [18] . І KG, розроблені з відповідними онтологіями для зберігання відповідних знань, будуть підтримувати ці програми. Наприклад такі моделі можуть зберігати знання про всі речі, що належать до конкретної предметної області або підприємства, в залежності від обставин. Взагалі Графи знань служать загальним субстратом знань, що постійно розвивається в рамках організації або спільноти. В сучасних розробках можна розрізнити два типи графів знань: графи відкритих знань і графи корпоративних знань. Графи відкритих знань публікуються в Інтернеті, що робить їх зміст доступним для загального користування. Найбільш відомі приклади – DBpedia, Freebase, Wikidata, YAGO тощо – охоплюють безліч доменів і або

взяті з Вікіпедії, або створені спільнотами добровольців. Графи відкритих знань також були опубліковані в конкретних областях, таких як ЗМІ, уряд, географія, туризм, науки про життя тощо. Графи корпоративних знань зазвичай є внутрішніми для компанії і застосовуються в комерційних випадках використання. Відомі галузі, що використовують графи корпоративних знань, включають веб-пошук (наприклад, Bing, Google), торгівлю (наприклад, Airbnb, Amazon, eBay, Uber), соціальні мережі (наприклад, Facebook, LinkedIn), фінанси (наприклад, Accenture, Banca d'Italia, Bloomberg, Capital One, Wells Fargo) і ін. Програми включають пошук, рекомендації, особистих агентів, рекламу, бізнес-аналітику, оцінку ризиків, автоматизацію і багато іншого.

Вже сьогодні графи знань використовуються великими корпораціями при розробці безлічі сучасних технологій і додатків. Серед прикладів додатків і технологій – поліпшення якості пошукової системи Google, розробка Семантичної павутини, проект DBpedia, Facebook Graph Search і багато інших. Вважається, що термін граф знань виник відносно нещодавно однак дослідження графічного подання знань проводилося протягом десятиліть, і термін «граф знань» не є новою технологією. Швидше, це модне слово, заново винайдене Google і прийняте іншими компаніями і академічними колами для опису різних додатків представлення знань. Існують суттєві відмінності в способах створення додатків уявлення знань: від повністю створених вручну баз знань до автоматично оброблюваних графів знань. Отже, термін «граф знань» не підходить для опису всіх наведених вище додатків, і його слід використовувати більш обережно. Деякі додатки не потрібно називати графами знань, тому що терміни база знань і онтологія описують їх досить і більш точно. Беручи до уваги різноманітність програм, KG більше схожий на абстрактну структуру, ніж на математичну структуру.

2.3 Алгоритм побудови графу знань з тексту на природній мові

Звідки беруться сутності, властивості і відносини, що становлять знання? Одним з джерел знань на підприємстві є корпоративні реляційні бази даних, що складаються з таблиць, що містять суті, атрибути і відносини в якості зовнішніх ключів до таблиць, що містить інші сутності. Ці сутності, атрибути та відносини легко перетворюються в вузли і ребра графа знань, після чого використовуються в програмних корпоративних додатках. Точно так само будь-яке структуроване джерело даних має необхідну метаінформацію, щоб направляти структуризовані дані в вузли і ребра графа знань. Крім структурованих джерел, іншим важливим джерелом знань є текст (неструктуровані дані).

Підприємства накопичують величезну кількість тексту в вигляді записок, контрактів, статей, електронних листів, оглядів продуктів, посібників, звітів, новин тощо [19]. На практиці ці гори текстової інформації являють собою неосяжне джерело знань, яке в свою чергу дуже складно використовувати без попередньої обробки та зручної моделі знань. Це не дивно, оскільки люди є кінцевими виробниками і споживачами знань, ці знання (сутності, властивості і відносини) проявляються в усній і друкованої мови (тексті). Текст – це де-факто джерело, наповнене знаннями. Для коректного подання алгоритму побудови графу знань з тексту не зайвим буде представити кроки алгоритму на прикладі (рис. 2.4). Наприклад, існує речення: «72-річний Джордж Мартін живе в Санта-Фе, штат Нью-Мексико, зі своєю дружиною Періс Макбрайд». Це речення містить сутності, відносини, властивості, що роблять його джерелом знань. Сутностями в цьому реченні виступають мовні конструкції, такі як «Джордж Мартін», «Санта-Фе«», «Періс Макбрайд», «Нью-Мексико». Відносинами і властивостями виступають: «живе в», «дружина». Також в реченні можна визначити значення віку людини: «72». Хоча деяких мовних конструкцій не представлено у реченні, з відповідними

лінгвістичними правилами можна створити відносини «знаходиться в» і «вік».

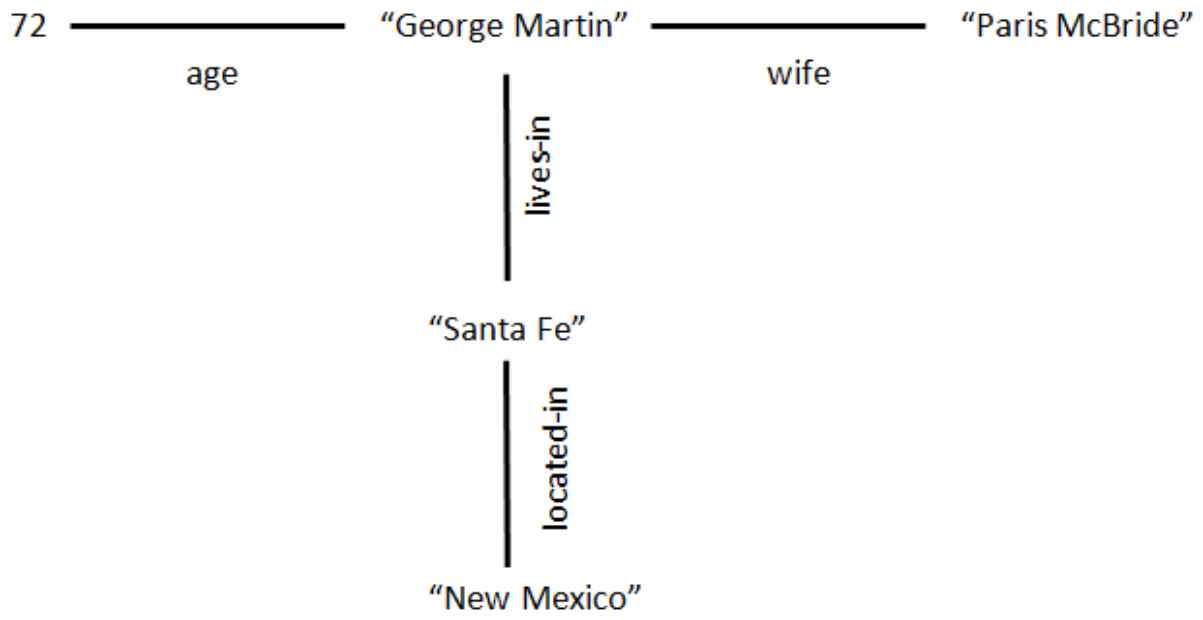

Рисунок 2.4 – Діаграма знань

Для того, щоб видобуті з речення знання можна було додати в граф знань, повинні виконати наступні кроки. Наприклад, рядок «Джордж Мартін» повинен бути перетворений у властивість «ім’я» об’єкту реального світу «Джорджа Мартіна», щоб усунути неоднозначність серед в попередніх знаннях, якщо вони існують. Як видно з прикладу, наскрізні конвеєри мовної обробки повинні бути здатні видобувати переданий в реченні сенс понад тим, що явно зазначено в тексті.

Вилучення знань підпадає під більш широке поняття – видобування інформації. Видобування інформації (IE) займається вилученням структурованої інформації, що відноситься до певної теми, з структурованих, напів структурованих і неструктурованих джерел даних, таких як форми введення даних, бази даних, веб-сайти, текстові документи тощо. BI включає видобування тексту-наповнювача з шаблонних слотів, таких як форми введення даних, заявки на роботу, форми реєстрації тощо.

BI більше використовується при добуванні інформації з веб-ресурсів, після видалення конкретних даних, таких як коди білків людини, коди продуктів GS1, поштові індекси, IP. Адреси, адреси електронної пошти.

З іншого боку, BI ускладнюється при добуванні знань, що добре проілюстровано в прикладі речення вище. Процес складається з декількох етапів в конвеєрі. Цими етапами є попередня обробка,розпізнавання сутностей (NER), видобування відносин, зв’язування з базою знань, інтеграція в цільовий граф знань.

Ці наскрізні конвеєри включають обробку на основі правил, класифікатори на основі машинного навчання і статичні конфігурації. Велика частина рішення полягає в перетворенні поверхневих словоформ в правильну концептуальну сутність при усуненні неоднозначності схожих слів, які насправді є різними концептуальними сутностями. Фаза зв'язування конвеєра додає семантичні анотації до сутностей на основі цільового графа знань. Типи сутностей зіставляються з відповідними класами RDF і URI примірників. Наприклад, рядки «Гейтс», «Засновник Microsoft» або «Філантроп» повинні бути в URI «Білла Гейтса», або слово «Яблуко» має перетворюватися в фрукт або компанію, що виробляє «iPhone». Open Information Extraction використовує відкриті джерела даних, що відносяться до конкретної предметної області, такі як Вікіпедія, як джерела навчальних даних і в якості цільової бази знань для фази зв'язування. Підприємства можуть додатково використовувати внутрішні бази даних.

Видобування сутностей (або розпізнавання іменованих сутностей – NER) включає вилучення та маркування послідовностей токенов слів з тексту в заздалегідь визначені класи. Правильне розпізнавання та тегування включає в себе усунення «со-посилань», усунення неоднозначності подібних згадок тощо. Як було зазначено у прикладі вище, слово «Яблуко» може ставитися до організації «Apple Inc.» або фрукт «Яблуко» в залежності від контексту. На різних етапах обробки

синтаксичні і семантичні правила, специфічні для предметної області, визначають ідентифікацію сутностей і усунення неоднозначності значення і контексту. Видобуті сутності класифікуються як люди, місця, організації, події, продукти, числа (дати, суми у валюті, номери телефонів, час, тривалість, частота, ваги тощо), які допомагають машинам зрозуміти, про що йде мова. Це змушує зазначену інформацію відповідати на фундаментальні питання, такі як «хто, що і коли ».

Точно так як видобування сутностей, видобування відносин націлений на отримання значущих асоціацій між сутностями з словотворів, які можуть включати дієслова. Підходи, в яких використовуються пошукові списки вказівних фраз для зіставлення відомих відносин, неефективні і не масштабуються. Природна мова допускає безмежну різноманітність «індикативних фраз» для конкретних відносин, і класифікатори машинного навчання краще працюють в цих умовах. Наприклад, відносина «працює на» між «Людиною» і «Організацією» може проявлятися в довільно складних і різноманітних виразах природної мови. Наприклад, «Сатья Наделла з Microsoft ...», «Генеральний директор Microsoft Сатья Наделла ...», «Сатья Наделла і його колеги з Microsoft ...» та інших подібних реченнях. Великі попередньо навчені моделі, точно налаштовані для навчальних завдань, таких як мовний наслідок або висновок, дають розумні передбачення тегів відносин.

Зв'язування сутностей включає зіставлення згадок (поверхневих форм) з відповідними записами в цільовому репозиторії (базі знань). Процес зв'язування усуває неоднозначність і зіставляє рядки слів з їх правильними сутностями в цільовій базі знань. Під час вилучення інформація з області репозиторіїв відкритих даних, таких як DBpedia, ці репозиторії служать цільовою базою знань, відносно якої відбувається видобування сутностей. Spotlight – це веб-сервіс для автоматичного анотування «текстових згадок» ресурсів DBpedia, але в разі приватного підприємства цільова база знань, швидше за все, буде власним сховищем

даних. Кураторні словники також використовуються для зв'язування сутностей. Для деяких відомих об'єктів, таких як згадки країн, списки всіх країн доступні у відкритому доступі. Такі програмні додатки підтримують виявлення нових сутностей за допомогою додаткових процесів курирування.

Існують різні способи виконання зв'язування сутностей, від зіставлення рядків та застосування складних правил для точних і нечітких зіставлень, до використання класифікаторів машинного навчання в контрольованому налаштуванні і методів семантичного дозволу, які використовують зіставлення підграфів. На рисунку 2.5 наведено вихідний граф знань, що виникає в кінці конвеєрної обробки наведеного у прикладі речення.

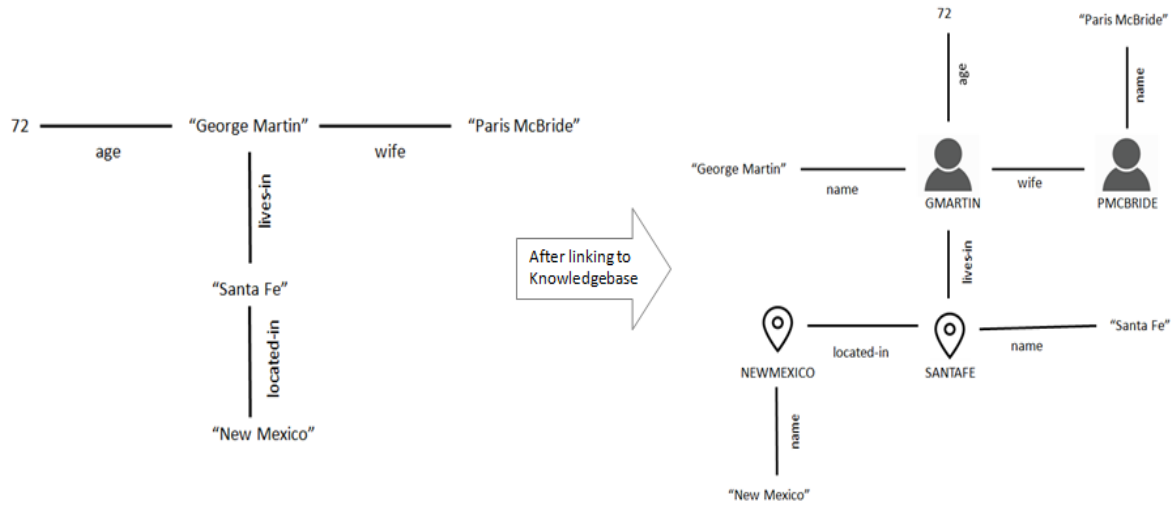

Рисунок 2.5 – Вихідний граф знань після зв’язування з базою знань

У лівій частині рисунку показані виділені фрагменти тексту з тегами. Дві незгаданих відносини «місцезнаходження» і «вік» генеруються з використанням спеціального набору лінгвістичних правил. Етап зв'язування зіставляє відмічені вилучення з правильними записами в базі знань і в кінцевому підсумку створює граф знань на правій частині

рисунку. Ярлики великими літерами під значками – це унікальні ідентифікатори об'єктів в базі знань.

Узагальнюючи усе вищезгадане можна зробити висновок – граф знань є найбільш зручним засобом для інтерпретації знань, видобутих з тексту, бо саме ця модель може ефективно відобразити об'єкти реального світу та відносини між ними.

3 МЕТОДИ ОБРОБКИ ПРИРОДНО-МОВНИХ ТЕКСТІВ

3.1 Загальна характеристика методів обробки природної мови для побудови графів знань

В області штучного інтелекту завжди передбачалося, що машини можуть імітувати функціонування і здатності людського розуму. Мова вважається одним з найбільш значних досягнень людини, який прискорює прогрес людства. Тому не дивно, що проводиться велика робота по інтеграції мови в область штучного інтелекту в формі обробки природної мови (NLP). NLP в першу чергу включає розуміння природної мови (від людини до машини) і генерацію природної мови (від машини до людини). В останні роки спостерігається зростання неструктурованих даних у вигляді тексту, відео, аудіо та фотографій. NLP допомагає видобувати цінну інформацію з тексту, таку як дані соціальних мереж, опитування клієнтів і скарги.

Одним з найбільш цінних ресурсів зберігання інформації і знань є природно-мовні тексти, які активно використовуються для побудови KG. У зв'язку з цим особливої актуальності набувають методи обробки природної мови (Natural Language Processing, NLP), які використовуються для здобуття знань з текстів.

Обробка природної мови (NLP) – це область на стику комп'ютерних наук, штучного інтелекту та лінгвістики. Мета полягає в тому, щоб комп'ютери обробляли або «розуміли» природну мова для виконання таких завдань, як мовний переклад, відповіді на питання, надання рекомендацій тощо.

З появою голосових інтерфейсів і чат-ботів, NLP стало однією з найважливіших технологій інформаційного століття і важливою частиною штучного інтелекту. Повне розуміння і представлення значення мови – надзвичайно важка мета, бо людська мова є дуже особливою. Нижче буде

представлено основні особливості природної мови, що можуть викликати труднощі при її обробці:

– людська мова – це система, спеціально створена для передачі сенсу, усно або письмово. Це не просто екологічний сигнал, а свідоме спілкування. Також природна мова часто піддається змінам;

– людська мова – це в основному дискретна, символічна, категоріальна сигнальна система;

– категоріальні символи мови можуть бути закодовані як сигнал для спілкування декількома способами: звук, жест, лист, зображення тощо. Людська мова може бути будь-яким з них;

– людські мови неоднозначні (на відміну від програмування і інших формальних мов). Таким чином, існує високий рівень складності в поданні, вивченні і використанні лінгвістичних, ситуаційних, контекстних, словесних та візуальних знань по відношенню до людської мови.

Ця область досліджень знаходить застосування у широкому спектрі задач. Через це існує швидкозростаюча колекція корисних додатків використовуючих методи NLP. Вони варіюються від простих до складних. Прикладами задач, що вирішують такі додатки, може виступати перевірка орфографії, пошук за ключовими словами, пошук синонімів, видобування інформації з веб-сайтів (наприклад, ціна продукту, дати, розташування, люди або назви компаній), класифікація текстів за складністю, машинний переклад, голосовий діалог тощо.

Додатки що використовують методи NLP знаходять своє місце в промисловості (рис. 3.1) : від пошуку (письмового та усного) до зіставлення онлайн-реклами, від автоматичного або допоміжного переводу до аналізу настроїв для маркетингу або фінансів. Також такі додатки можуть використовуватися для розпізнавання мови в чат-ботах, діалогових агентах для вирішення задач автоматизації та підтримки клієнтів, управління пристроями, замовлення товарів. У даному випадку методи NLP будуть використані для вирішення такого завдання як інтеграція

даних. Інтеграція даних – це процес об'єднання даних з різних джерел і надання користувачеві єдиного уявлення даних.

Один з підходів до інтеграції даних, заснований на використанні глобальной схеми, яка фіксує взаємозв'язки між елементами даних, представленими в різноманітних джерелах даних. Створення глобальної схеми – надзвичайно складний процес, тому що існує безліч неструктурованих джерел даних, які складно організувати у вигляді схеми.

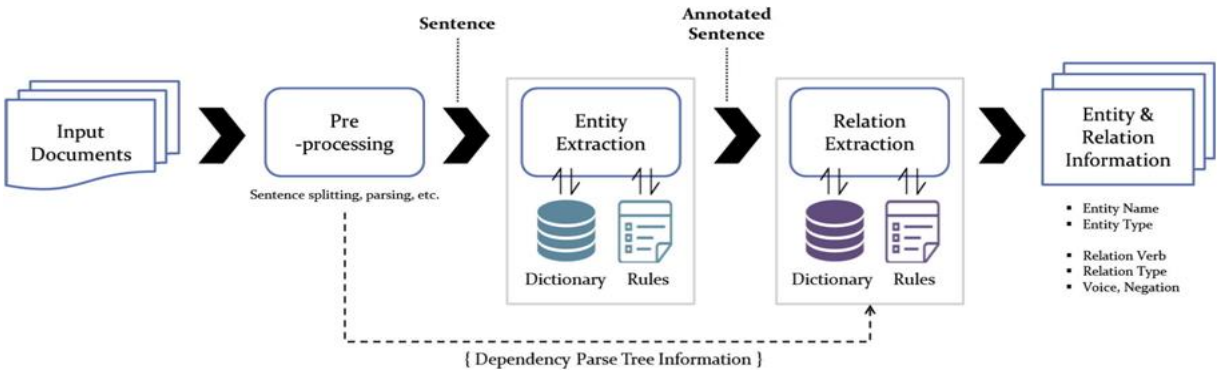

Рисунок 3.1 – Приклад використання методів NLP

Через проблеми, пов'язані зі створенням глобальної схеми, цю проблему зручно обійти організувавши дані у вигляді знань. Наприклад, якщо брати текстове джерело даних, то для побудови глобальної схеми можно виділити в тексті основні сутності та відносини між ними у вигляді трійок, після чого організувавши їх у вигляді зв’язаної бази знань, тобто графу знань.

В сучасному світі представлено широкий спектр текстових джерел, що стосуються різноманітних сфер життя. Наприклад у діловій сфері можна зіткнутися з фінансовими новинами, документами комісії з цінних паперів і бірж, і безлічі веб-сайтів, що стосуються бізнесу, таких як Wall Street Journal. Ці текстові джерела містять інформацію, що має велике значення для безлічі бізнес-завдань, таких як дослідження ринку, бізнес-аналітика тощо. Методи обробки природної мови (NLP) в даному випадку

можуть бути використані для обробки текстових джерел і створення графів знань, які можуть підтримувати різну аналітику. Однак NLP – це спеціалізована і складна технологія. Тому першим чином треба познайомитись з деякими базовими концепціями NLP, які можуть бути корисні для створення графа знань. Метою NLP у цьому випадку є виділення в тексті окремих сутностей, їх властивостей, а також відносин між ними.

Основними задачами NLP в даному випадку, що мають безпосереднє відношення до побудови графа знань, є видобування сутностей і видобування відносин [20]. Видобування сутностей – це завдання ідентифікації ключових сутностей (наприклад, організацій, людей, місць тощо) з тексту. Ці сутності зазвичай складають вузли в графі знань. Видобування відносин – це завдання, в якому, маючи дві сутності, що представляють інтерес, та деякий текст, треба видобути з тексту відносини між ними (наприклад інформацію про робітників конкретних компаній тощо). Іноді видобування відносини також використовується для вилучення властивостей деяких сутностей. Видобуті відносини і властивості зазвичай стають відносинами або властивостями вузла в графі знань.

Таким чином, для побудови графів знань доцільно використовувати такі методи NLP як видобування сутностей та залежностей. Також для коректної роботи з текстом знадобляться такі методи NLP як сегментація речень та тегування частин мови. Далі буде надано короткий опис названих вище методів NLP.

Тегування частин мови готує речення до подальшого синтаксичному аналізу шляхом присвоєння семантичного опису мовним структурам в реченні. Для побудови графу знань програмі треба чітко розуміти відносини між сутностями. Але в методах обробки природної мови (NLP) часто використовуються моделі засновані на Bag of Words. Але такі моделі не можуть відобразити синтаксичні відносини між словами. Тому виходом

з цієї ситуації стало тегування частин мови . Теги частини мови (далі звані POS) корисні для побудови дерев синтаксичного аналізу, які використовуються при побудові NER (більшість іменованих сутностей – це іменники) і вилучення відносин між словами. Маркування POS – це процес розмітки слова в корпусі до відповідної частини мовного тега на основі його контексту і визначення.

Видобування сутностей – це метод вилучення інформації, який відноситься до процесу ідентифікації та класифікації ключових елементів тексту за заздалегідь визначеними категоріями. Таким чином, він допомагає перетворити неструктуровані дані в структуровані, а значить, машинозчитувані і доступні для стандартної обробки, яка може застосовуватися для вилучення інформації, вилучення фактів і відповідей на питання. Видобування сутностей може забезпечити корисне враження про невідомих наборах даних за рахунок негайного виявлення, як мінімум, того, хто і що містить ця інформація.

Сегментація тексту при побудові графа розбиває текстові документи або статті на речення, з яких будуть вилучені суті і відносини між ними. Сегментація речень – це процес поділу деякого фрагменту писемної мови на складові речень.

Видобування залежностей дозволяє отримати з речення суб'єкт і об'єкт, а також виділити відносини між ними, ґрунтуючись на семантичному описі частин мови в реченні і семантичних залежностях між ними. Видобування відносин – це завдання прогнозування атрибутів і відносин для сутностей в реченні. Наприклад, з огляду на речення «Барак Обама народився в Гонолулу, Гаваї», класифікатор відносин націлений на пророкування відносини «народився у місті». Видобування відносин є ключовим компонентом для побудови графів знань про взаємозв'язки і має вирішальне значення для додатків обробки природної мови, таких як структурований пошук, аналіз тональності, відповіді на питання і узагальнення.

Таким чином методи NLP допомагають видобувати дані з тексту у зручному форматі знань для подальшої обробки або інтеграції у графи знань. У наступних розділах ці методи будуть описані більш детально.

3.2 Видобування сутностей

Видобування сутностей, також відомий як ідентифікація сутності, розбиття сутностей на частини і розпізнавання іменованих сутностей (NER), являє собою процес пошуку і класифікації згадок сутностей в фрагментах тексту. Це робиться за допомогою системи зумовлених категорій, які можуть включати в себе все, від людей або організацій до тимчасових і грошових значень.

Процес вилучення сутності додає структуру і семантичну інформацію до раніше неструктурованого тексту . Це дозволяє алгоритмам машинного навчання виявляти згадки певних сутностей в тексті і навіть резюмувати великі фрагменти контенту. Це також може бути важливим етапом попередньої обробки для інших завдань обробки природної мови (NLP). Завдяки широкому спектру потенційних варіантів використання, від оптимізації підтримки клієнтів до оптимізації пошукових систем, видобування сутностей грає життєво важливу роль у багатьох моделях NLP, які використовуються щодня [21], [22].

Першим чином для розуміння роботи методів видобування сутностей треба розібратися, що саме з себе представляють ці сутності. Взагалі, сутність – це все, що має окреме і автономне існування. Це може бути фізичне тіло, наприклад собака, дерево або будинок. Сутність також може бути концептуальним поняттям, наприклад нацією, організацією або почуттям. З лінгвістичної точки зору переважна більшість іменників можна розглядати як об'єкти в тій чи іншій формі.

Тепер, коли поняття сутності було описано, можна розібратися з тим, що з себе представляє процес видобування сутностей. У цьому контексті

видобування включає виявлення, підготовку і класифікацію згадок сутностей в заданому фрагменті необроблених текстових даних. Це можна зробити вручну, використовуючи інструменти анотації, такі як BRAT, або за допомогою рішення NER. Система категоризації, в яку упорядковано ці об'єкти, може бути унікальною для кожного проекту. Суті можна розділити на групи як широкі, як люди, організації та місця, так і вузькі, як розчинні і нерозчинні хімічні речовини.

Перш ніж стане можливим отримувати сутності з тексту, необхідно виконати кілька завдань попередньої обробки. Наприклад, токенізація допомагає визначити межі одиниць в тексті, такі як початок і кінець слова. Позначка частини мови (POS) анотує одиниці тексту з їх граматичної функцією, що допомагає закласти основи для розуміння відносин між словами, фразами і пропозиціями. Ці процеси допомагають машині визначити положення об'єктів і почати екстраполювати їх ймовірну роль в тексті в цілому. Незважаючи на те, що вони грають важливу роль в отриманні сутностей, вони зазвичай не визначаються як такі. Замість цього фахівці з обробки даних зазвичай припускають, що ці кроки попередньої обробки вже виконані, під час обговорення свого проекту вилучення.

Однак це не змінює того факту, що видобування сутностей – складний багатоетапний процес. Тому процесс видобування сутностей краще розбирати на конкретному прикладі. Далі буде розглянуто наступний фрагмент тексту з новин: «52-річна Сесілія Лав, слідчий поліції на пенсії, яка живе в Массачусетсі, сказала, що вона заплатила близько 370 доларів за квиток з податком на прямі рейси United Airlines в Сакраменто з Бостона на випускний для своєї племінниці середньої школи в червні 2020 року.»

В даному прикладі розглядаються названі сутності в наведеному вище тексті. Іменована іменована сутність – це все, на що можна посилатися за допомогою власного імені: людина, місце розташування, організація і т. Д. Визначення іменованої суті зазвичай розширюється, щоб

включати в себе речі, які не є сутностями як такі, включаючи дати, час. , І інші види виразів, що включають час, і навіть числові вирази, наприклад, ціни. Ось текст вище із зазначеними іменованими об'єктами: «[PER Сесілія Лав], 52 роки, слідчий поліції на пенсії, яка проживає в [LOC New Jersey], сказала, що вона заплатила близько [MONEY $ 370] квиток з податком на прямий рейс [ORG United Airlines] в [LOC Sacramento] з [LOC Boston]] на випускній церемонії її племінниці в [TIME, червень, 2020]».

У тексті вище міститься сім названих сутностей, одне з яких – фізична особа (позначено PER), три – розташування (позначено LOC), одне – гроші (позначено MONEY), одне – організація (позначено ORG) і одне – час (позначається TIME). Залежно від області застосування можна ввести більш-менш іменовані типи сутностей. Наприклад, в задачі ідентифікації ключових термінів у тексті є тільки один тип сутності, який фіксує ключовий термін.

Завдання вилучення сутностей корисна в багатьох різних додатках. Наприклад, при відповіді на питання він може допомогти отримати відповіді з отриманого фрагмента тексту. В текстовому редакторі це може допомогти автоматично зв'язувати суті, з'являються в тексті, з додатковою інформацією (наприклад, визначеннями, фактами тощо) про ці сутності. При такому підході можна розглядати видобування сутностей як проблему маркування. В процесі видобування сутностей відбувається зв’язування мітки з кожним словом, після чого наступним кромок стає завдання передбачення мітки. Для видобування сутностей часто використовуються три широкі підходи: маркування послідовностей, моделі глибокого навчання і підходи на основі правил.

Для полегшення маркування також часто використовують схему маркування, відому як BIOES, в якій значення різних тегів наступні: B позначає початок об'єкта, I позначає внутрішню частину об'єкта, O позначає слово, яке не є частиною об'єкта, E позначає кінець об'єкта, а S позначає об'єкт , що складається з одного слова. Наприклад, слова в

наведеному вище фрагменті тексту будуть позначені, як показано нижче (рис. 3.2).

Cecilia	B	Love	E	,	O	52	O	,	O
a	O	retired	O	police	O	investigator	O	who	O
lives	O	in	O	Massachusetts	S	,	O	said	O
she	O	paid	O	around	O	$370	S	a	O
ticket	O	with	O	tax	O	for	O	nonstop	O
United	B	Airlines	E	flights	O	to	O	Sacramento	S
from	O	Boston	S	for	O	her	O	niece's	O
high	O	school	O	graduation	O	in	O	June	B
,	I	2020	E						

Рисунок 3.2 – Приклад маруквання за схемою BIOES

У машинному навчанні маркування послідовностей – це тип завдання розпізнавання образів, який включає в себе алгоритмічне привласнення категоріальної мітки кожному члену послідовності спостережуваних значень. Типовий приклад завдання маркування послідовностей – це маркування частин мови, яка спрямована на присвоєння частини мови кожного слова у вхідному пропозиції або документі. Маркування послідовностей можна розглядати як набір незалежних завдань класифікації, по одній на кожного члена послідовності.У підході з маркуванням послідовностей навчається один з алгоритмів машинного навчання (наприклад, умовні випадкові поля), використовуючи такі функції, як: відношення до частини мови, наявність слова в списку стандартних слів, ембедінги слів, наявність в слові префіксу або суфіксу, розпізнавання регістру. При розробці функцій потрібні значні зусилля, оскільки продуктивність конкретного вибору функцій і алгоритму машинного навчання може варіюватися в залежності від домену. Метод маркування послідовностей, може використовувати залежності між інформація для підвищення продуктивності вилучення. Він

також заснований на статистичній теорії і, отже, має сильні можливості узагальнення. У багатьох додатках, зокрема при обробці природної мови, він може перевершити метод, заснований на правилах.

У підході глибокого навчання використовуються ембединги слів, що вводяться в мовну модель. Замість того, щоб передбачати наступне слово, мовна модель тепер пророкує один з п'яти тегів (B, I, O, E, S), які потрібні для розпізнавання сутностей. Для адаптації мовної моделі до нової задачі відбувається попереднє навчання, з використанням текстового корпусу для заданої області, а потім навчається задля виконання конкретного завдання. У навчанні мовної моделі для конкретних завдань зазвичай забезпечується додавання виділеного токену [CLS], що позначає початок об'єкта, і другого виділеного токену [SEP], який позначає кінець об'єкта. Таке навчання дозволяє моделі передбачати виділені теги у відповідь на введення тексту. Таких прогнозів досить, щоб зробити один з п'яти обов'язкових тегів для кожного слова.

Нарешті, в підході, заснованому на правилах, кожен визначає правила маркування на формальній мові запитів. Правила можуть включати регулярні вирази, посилання на словники, семантичні обмеження, а також можуть викликати автоматичні екстрактори і структури довідкових таблиць. Правила також можуть викликати модулі машинного навчання для конкретних завдань. Застосування правил може бути впорядковано таким чином, що спочатку використовуються правила високої точності, потім виконується пошук в стандартному списку імен, потім евристики на основі мови, а коли все інше не вдається, застосовуються методи імовірнісного машинного навчання. Метод, заснований на правилах, намагається використовувати регулярність появи певної інформації в мовних виразах, щоб знайти спільні мовні шаблони, які відповідають цим виразами. Звичайному користувачеві це легко зрозуміти. Метод може забезпечити гарну продуктивність при обробці деяких частково структурованих документів (наприклад, веб-сторінок на

основі шаблонів). Його недолік полягає в тому, що його рудиментарні механізми навчання не можуть забезпечити достатньо можливості узагальнення. Це ускладнює отримання хорошої продуктивності в складні ситуації (наприклад, у разі видобування сутностей з тексту на природній мові).

При використанні методів видобування сутностей у задачі побудови графів знань не зайвим буде зазначити деякі проблеми, що можуть виникнути при обробці текстових джерел цими методами. Незважаючи на те, що для конкретних завдань екстрактори сутностей можуть демонструвати точність і відгук вище 90%, але отримання хорошої продуктивності у всіх областях може виявитися складним завданням. При маркуванні сутностей можуть виникати численні випадки неоднозначності. Наприклад, у випадку об’єкта «Louis Vuitton» назва може ставитися або до фізичної особи, або до організації, або до комерційного продукту. Усунення такої двозначності неможливо, якщо в текстовому джерелі не буде представлено значного контексту, в якому цей об’єкт було знайдено.

Для видобування сутностей з використанням підходу машинного навчання потрібна величезна кількість даних. На практиці дані можуть бути відсутні або в значній мірі неповні. Під час навчання моделі з використанням неповних даних серйозно страждає продуктивність.

Одним з варіантів завдання вилучення сутностей є визначення ключових фраз в тексті. Оскільки ключові фрази не обмежуються кількома класами, завдання визначення конкретного класу, відповідного ключовій фразі, може стати ще більш складним. Іноді ключові фрази занадто складні (наприклад, дублювання клітини шляхом поділу), а іноді і занадто загальні (наприклад, прикріпити), що дуже ускладнює застосування єдиної техніки для всіх.

Сутності можуть з'являтися в багатьох різних поверхневих формах, наприклад, у вигляді синонімів, акронімів, у формі множини і

морфологічних варіаціях. Загалом, видобування сутностей має враховувати лексичні знання, які зазвичай не існують при обробці тексту, що відноситься до нової області знань. В результаті для підвищення продуктивності вилучення сутностей видобування лексичного значення стає важливим пов'язанним завданням.

Прагнучи застосувати методи і технології з практичної інформатики, такі як створення компіляторів і штучний інтелект, до проблеми автоматичної обробки неструктурованих текстових даних, вилучення сутностей стало важливою субдисципліною мовної інженерії, в галузі інформатики. В даний час важливість вилучення інформації визначається зростаючим обсягом інформації, доступної в неструктурованій (тобто без метаданих) формі, наприклад, в Інтернеті. Видобування сутностей страждає від невизначеності і імплікації природної мови. Обидві ці проблеми важко піддаються механічному або автоматичному вилученню, іноді навіть людині.

Але незважаючи на недоліки підходу видобування сутностей, його методи є абсолютно необхідними для здатності машин розуміти мову. Оскільки обсяг нових доступних даних зростає експоненціально, процес, який додає структуру до цієї хаотичної лавині шуму, стає більш цінним, ніж будь-коли. Незалежно від того, чи використовується він як етапу попередньої обробки або є самостійною кінцевою метою, видобування сутностей може зіграти величезну роль в діловому світі майбутнього. Зокрема методи видобування сутностей розглядаються як один з основних компонентів для побудови зручної та природної моделі подання знань – графів знань.

3.3 Видобування відносин

У мережі існує величезна кількість неструктурованого електронного тексту, включаючи стрічку новин, блоги, електронну пошту, урядові

документи, журнали чатів тощо. Як можна допомогти людині зрозуміти всі ці дані? Популярна ідея – перетворити неструктурований текст в структурований шляхом анотування семантичної інформації. Однак через величезний обсяг і різнорідность даних використання людських ресурсів для анотування неможливе. Замість цього ми б хотіли б, щоб комп'ютер анотував всі дані та подавав їх у вигляді цікавлячої нас структури. Зазвичай нас цікавлять відносини між сутностями, такі як людина, організація і місцезнаходження. Прикладами відносин є приналежність особистості та місцезнаходження організації. Сучасні розпізнавачі іменованих об'єктів (NER), можуть автоматично маркувати дані з високою точністю. Однак весь процес вилучення відносини – нетривіальне завдання. Комп'ютер повинен знати, як розпізнати фрагмент тексту, який цікавить своїми семантичними властивостями, щоб зробити правильну анотацію. Таким чином, виявлення семантичних відносин між об'єктами в тексті на природній мові є вирішальним кроком на шляху до додатків для розуміння природної мови.

Видобування відносин (RE) – це задача вилучення семантичних відносин з тексту, які зазвичай виникають між двома або більше об'єктами. Ці відносини можуть бути різних типів. Наприклад, «Париж знаходиться у Франції» означає, що «знаходиться у» це відносина між Парижем і Францією. Це можна позначити за допомогою трійок (Париж, знаходиться у, Франція). Видобування інформації (IE) – це область вилучення структурованої інформації з тексту на природній мові. Ця галузь часто використовується для різних завдань НЛП, таких як створення графів знань, систем типу питання-відповідь, узагальнення тексту тощо. Видобування відносин сам по собі є підгалуззю видобування інформації (IE).

Розглядаючи фрагмент тексту з прикладу, що було наведено у розділі про видобування сутностей, можна видобути такі відносини, як Сесілія Лав живе в Массачусетсі, United Airlines вилітає з Бостона, United

Airlines летить в Сакраменто. У типовій задачі вилучення відносин сутності зазвичай вже ідентифіковані, і в цьому сенсі ця задача розширює завдання вилучення сутностей. Відносини, які потрібно видобувати, також можуть вказуються заздалегідь.

Популярним прикладом завдання вилучення відносин є отримання інформації з інформаційних джерел Вікіпедії. Очевидне застосування цього завдання – поліпшити результати пошуку в Інтернеті. Інформаційні блоки Вікіпедії визначають такі відносини, як є наступником, є дітьми, є чоловіком тощо. Досягнення високої точності в цьому завданні може бути складною задачею через безліч різноманітних випадків. Наприклад, Ларрі Кінг був одружений кілька разів, і тому екстрактор повинен врахувати цей час, протягом якого існував шлюб.

Також існують відносини, що залежать від предметної області. Наприклад, Unified Medal Language Systems підтримує такі відносини, як причини, лікування, порушення тощо. Крім стандартних відносин, таких як «є підкласом», є відносини, що залежать від предметної області і зазвичай вимагають деякої попередньої роботи з цією областю. Є кілька підходів до вилучення відносин, які не вимагають завчасного вибору відносин, але виявилося, що корисність таких підходів на практиці обмежена.

Існує п'ять різних методів видобування відносин (relation extraction, RE):

– RE на основі правил;

– слабо контрольований RE;

– контрольований RE;

– дистанційно контрольований RE;

– неконтрольований RE.

Процес видобування відносин спочатку буде продемонстровано на прикладі, після чого буде розглянуто основні підходи для видобування відносин. Класичний підхід до видобування відносин заснований на

синтаксичних патернах, відомих як патерни Херста. За допомогою цього підходу можна ідентифікувати синтаксичні патерни, які є сильними індикаторами відносин, що являють собою приналежність до підкласу. Наступні п'ять синтаксичних патернів існують вже досить давно і виявилися надзвичайно ефективними на практиці (табл. 3.1) [23].

Таблиця 3.1 – Приклад синтаксичних патернів для видобування відносин за допомогою підхода Херста

Назва патерну	Приклад
«такі як»	«... роботи таких авторів, як Геррік, Голдсміт і Шекспір ...»
«або інші»	«Синці, рани, зламані кістки або інші травми ...»
«та інші»	«... скарбниці та інші цивільні будівлі.»
«включаючи»	Усі правові країни, включаючи Канаду та Англію ...»
«особливо»	«Більшість європейських країн, особливо Франція, Англія та Іспанія, ...»

З подібним підходом можна відкрити нові синтаксичні шаблони для будь-яких відносин з вибору наступним чином. Спочатку треба зібрати приклади, для яких відомо, що відносина знайдена вірно. Потім треба провести пошук речень, в яких є вірні відносини. Виявивши загальні риси в таких реченнях, можна визначити новий патерн. Наступним кроком буде тестування таких шаблонів на текстовому корпусі. Один з таких

алгоритмів відомий як розширення подвійної ітеративної відносини шаблонів (DIPRE). Для зручності демонстрації процесу видобування відношень на прикладі буде проілюстровано його роботу над проблемою вилучення пар (автор, назва) з корпусу. Все починається з невеликого набору відомих пар (автор, заголовок), і вже заздалегідь зазначені все їх входження в корпусі, і з них може бути згенеровано більше шаблонів. Алгоритм рекурсивно продовжує роботу, використовуючи нові шаблони для пошуку нових книг і відкриваючи нові шаблони. Як конкретний приклад, з огляду на вихідну пару (Вільям Шекспір, Комедія помилок) можна представити наступні речення:

«Комедія помилок Вільяма Шекспіра була … »;

«Комедія помилок Вільяма Шекспіра … »;

«Комедія помилок, одна з найбільш ранніх спроб Вільяма Шекспіра...»;

«Комедія помилок, одна з найбільш … ».

З такого набору речень можна вивести наступні закономірності: x, по y, x, один з y.

Далі детальніше розглянемо зазначені вище підходи до видобування відношень з неструктурованого тексту.

Видобування відношень на основі правил є першим підходом до RE. Багато екземплярів відносин можуть бути ідентифіковані за допомогою створених вручну шаблонів, які шукають трійки (X, α, У), де X – сутності, а α – слова між ними. Для прикладу «Париж знаходиться у Франції» α = «знаходиться в». Його можна видобувати за допомогою регулярного виразу. Також у процесі видобування відношень важливу роль грає розпізнавання іменованих сутностей (NER) та тегування частин мови (POS), бо часто сутностями виступають назви об’єктів, а у якості відношень – мовні конструкції з глаголами (рис. 3.3, 3.4).

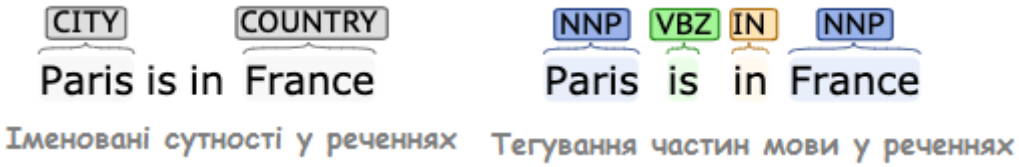

Рисунок 3.3 – Приклад використання NER та POS

Однак один лише перегляд збігів ключових слів також призведе до безлічі помилкових спрацьовувань. Це можна пом'якшити, фільтруючи по іменованим об'єктам, тобто тільки видобуваючи реально існуючі поняття (Місто, знаходится у, Держава). Також зручним рішенням буде прийняття до уваги тегів частин мови (POS) для видалення додаткових помилкових спрацьовувань алгоритму.

На цьому прикладі зазначено використання паттерну послідовності слів, тому що правило визначає шаблон спираючись на порядок тексту. На жаль, правила такого типу не підходять для патернів з більш довгим діапазоном і послідовностей з великою різноманітністю. Наприклад. «Федор і Альона одружилися» не може бути успішно оброблено шаблоном послідовності слів.

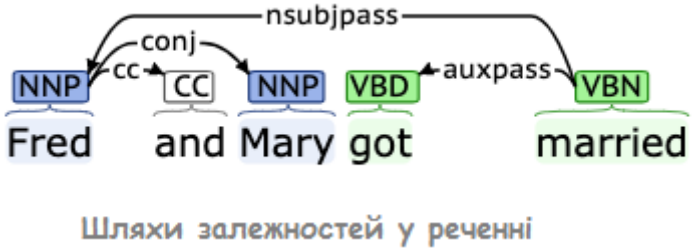

Рисунок 3.4 – Приклад використання шляхів залежності у реченні

Замість правил наведених вище можна використовувати шляхи залежності в реченнях, які дають можливість знаходити слово, що має

граматичну залежність від іншого слова. Це може значно збільшити охоплення правила без додаткових зусиль.

Також можна перетворити речення перед застосуванням правила. Наприклад. «Торт спік Гаррі» або «Торт, який спік Гаррі» можна перетворити в «Гаррі спік торт».Тобто можна змінити порядок, щоб працювати тільки з «лінійним правилом», одночасно видаляючи зайве слово-модифікатор між ними.

Перевагами підходу заснованого на правилах є те, що можна вручну створювати патерни, що будуть мати високою точністю, а також те що такі патерни можуть бути адаптовані до конкретних текстових джерел.

У якості недоліків можна виділити: а) людські патерни як і раніше часто мають малий відгук (занадто багато мов); б) задіяно багато ручної роботи, щоб створити всі можливі правила; в) для більшої точності треба створювати правила для кожного типу відносин.

Другим методом видобування відносин з природно-мовних текстів є слабо контрольований RE. Ідея тут полягає в тому, щоб почати з набору створених вручну правил і автоматично знаходити нові з немаркованих текстових даних за допомогою ітеративного процесу (початкового завантаження). В якості альтернативи можна почати з набору вихідних кортежів, що описують сутність з певною відносиною. Наприклад джерело = {(ORG: IBM, LOC: Armonk), (ORG: Microsoft, LOC: Redmond)} вказує сутності, що мають відношення «на основі». Процесс обробки за допомогою слабо-контрольованого підходу можна відобразити на прикладі алгоритму Snowball. Snowball – досить старий приклад алгоритму, який виконує наступні кроки:

– починає з набору готових кортежів (або видобуває вихідний набір з немаркованого тексту за допомогою декількох створених вручну правил);

– видобуває входження з немаркованого тексту, яке відповідає кортежам, і проводить їх тегування за допомогою NER (розпізнавання іменованих сутностей);

– створює шаблони для цих випадків, наприклад «ORG базується в LOC»;

– генерує нові кортежі з тексту, наприклад (ORG: Intel, LOC: Santa Clara) і додає цей набір до загальної бази з кортежами.

Після ітерації алгоритм пбо переходить до кроку 2 або завершує роботу і використовувати створені шаблони для подальшого видобування відносин.

На відміну від підходу на основі правил в цьому підході може бути виявлено більше зв'язків(вищий рівень відгуку). Також як перевагу можна відзначити те, що для цього підходу потрібно менше людських зусиль (потрібно тільки високоякісна вибірка кортежів).

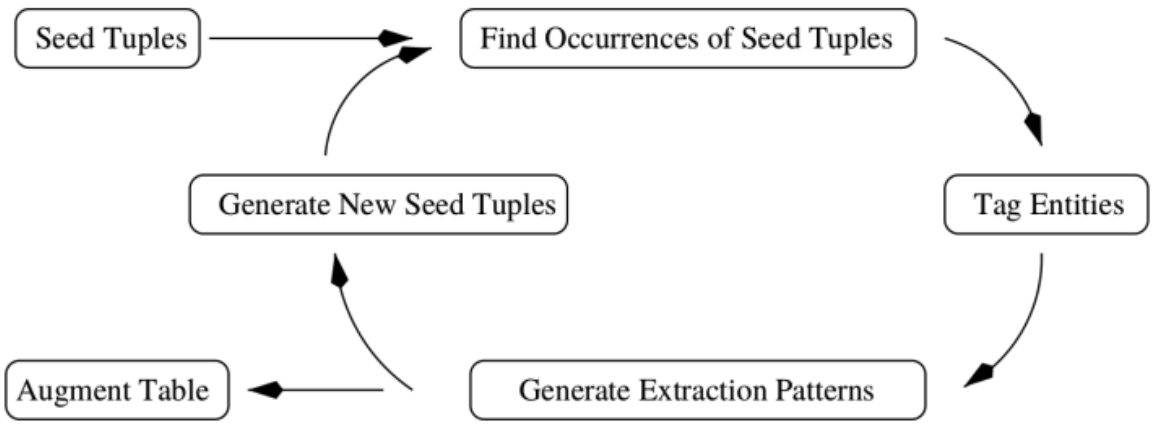

Рисунок 3.5 – Алгоритм роботи Snowball

Але все ж таки такий підхід має свої недоліки. Набір шаблонів стає більш схильним до помилок після кожної ітерації. Також необхідно дотримуватися обережності при створенні нових шаблонів за допомогою входжень кортежів, наприклад «IBM закрила офіс в Хёрслі» може бути помилково класифікована як відносина «розташовано в» при генерації шаблонів.

Контрольований RE є третім методом видобування відносин з природно-мовних текстів. Поширеним способом виконання контрольованого видобування відносин є навчання складеного бінарного класифікатора (або звичайного двійкового класифікатора), щоб визначити, чи існує конкретний зв'язок між двома об'єктами. Ці класифікатори приймають особливості тексту в якості вхідних даних, таким чином, вимагаючи, щоб текст спочатку був анотований іншими модулями НЛП. Типовими функціями є: контекстні слова, теги частини мови, шляхи залежностей між об'єктами, теги NER, токени тощо. При контрольованому підході до видобування сутностей існує ряд кроків за допомогою яких буде навчатися система.

Першим чином треба вручну позначати текстові дані в залежності від того, чи є речення релевантним для певного типу відносини. Наприклад. для відносини «генеральний директор» речення «Генеральний директор Apple Стів Джобс сказав Біллу Гейтсу...» є релевантним, а речення «Боб сказав Біллу Гейтсу» – ні. Далі треба вручну позначати відповідні речення як позитивні або негативні, якщо вони виражають відносину. Наприклад для речення «Генеральний директор Apple Стів Джобс сказав Біллу Гейтсу», вираз «Стів Джобс, генеральний директор Apple » є позитивним «Білл Гейтс, генеральний директор Apple» – негативним. Наступним кроком буде навчити бінарний класифікатор визначати, чи є речення актуальним для типу відносини. Після цього бінарний класифікатор будет навчатися за відповідними реченнями, щоб визначити, чи висловлює речення відносину, що відносится до конкретного типу, чи ні. Після навчання бінарного класифікатора його можна використовувати для виявлення відносин в нових текстових даних.

Такий підхід передбачає, що в процесі навчання системи для видобування відносин усі види відносин будут позначатися вручну, тобто буде здійснюватися контроль на всіх етапах навчання. Перевагою такого підходу є високоякісний нагляд (забезпечення актуальності видобутих

відносин). Також в процесі позначення залежностей з'являються явні негативні приклади, що може дуже допомогти у навчанні. Але при застосування такого підходу дуже складно маркувати приклади, відносини їх до одного з типів відносин. Також дуже складно додавати нові відносини, бо кожного разу буде потрібно будет навчати новий класифікатор. Цей підхід не підходить для нових предметних областей та з його допомогою видобування можна здійснювати тільки для невеликого набору типів відносин.

Метод дистанційно контрольованого RE видобування відносин передбачає поєднання ідеї використання вихідних даних, як для підходу слабо контрольованого видобування відносин, з навчанням класифікатора, як для контрольованого видобування відносин (рис. 3.6). Однак замість створення набору вихідних кортежів можна взяти його з існуючої бази знань (КВ), такої як Wikipedia, DBpedia, Wikidata, Freebase, Yago.

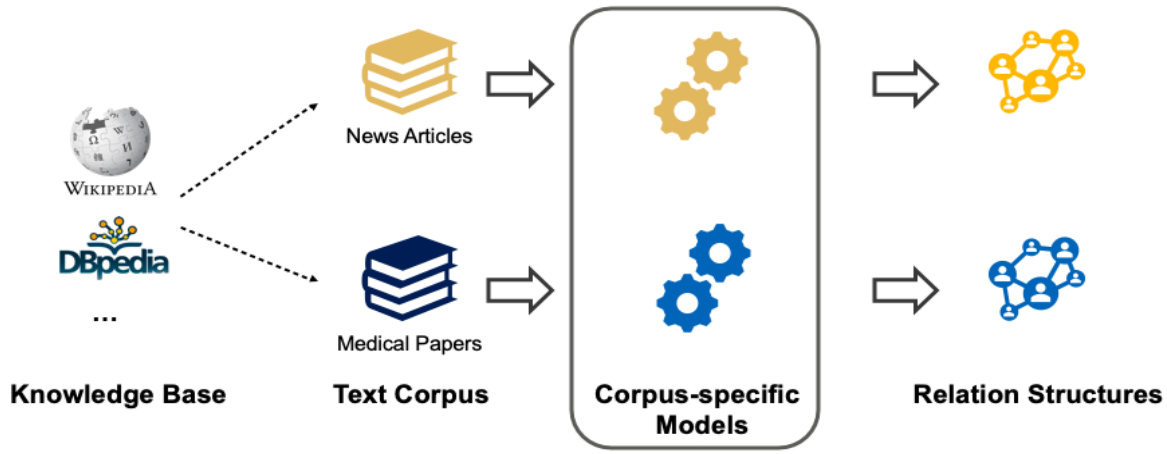

Рисунок 3.6 – Схема дистанційно контрольованого RE

Типи відносин та кортежи з цими відносинами при такому підході знаходятся в обраній базі знань. Алгоритм видобування відношень при такому підході відображають наступні кроки:

– вибираються речення з немаркованих текстових даних, які відповідають вихідним кортежам (обидва слова кортежу збігаються в реченні), і припускається, що ці речення є позитивними прикладами для цього типу відносини;

– відбувається видобування особливостей з цих речень(наприклад, POS, контекстні слова тощо);

– на основі видобутих речень навчається класифікатор.

Перевагами такого підходу є зменшення роботи, що має бути зроблена вручну, а також система може масштабироваться для використання великої кількості помічених даних і безлічі відносин. Однак при застосуванні в такому підході бази знань, речення що містять обидва слова в кортежі, насправді можуть не описувати конкретної відносини. Також існує обмеження у вигляді кількості речень, що містяться базі знань, а також кількості описаних відносин та предметних областей, що в ній зберігаються. Також через різноманітність даних може знадобитися ретельна настройка під конкретну задачу.

Метод неконтрольованого RE видобування відносин припускає, що відносини з тексту видобуваються без необхідності маркування будь-яких навчальних даних, надання набору вихідних кортежів або необхідності писати правила для захоплення різних типів відносин в тексті. Замість цього алгоритм покладається на набір дуже загальних обмежень і евристик. Відкрите видобування інформації (Open IE) зазвичай відноситься до цієї парадигми. Дія алгоритму буде описана для трьох етапів його роботи на прикладі алгоритму TextRunner.

Першим етапом цього алгоритму буде навчання самостійно контрольованого класифікатора на невеликому текстовому корпусі. Для цього кожному проаналізованному реченню знаходят всі пари словосполучень (X, У) з послідовністю слів r, що з'єднує їх. Якщо вони відповідають усім обмеженням їх маркують як позитивні приклади, в іншому випадку – негативні. Далі відбувається заставлення кожного

триплету (X, r, У) з векторним поданням ознак Цими ознаками може бути відношення до конкретної частини мови, тег NER тощо. Все це робиться для того, щоб навчити бінарний класифікатор визначати надійних кандидатів.

Наступним етапом в роботі алгоритму буде пропуск всього корпусу і видобування можливих відносин. На цьому етапі потенційні відносини виймаються з корпусу. Потім класифікатор вирішує залишаються або відхиляються триплети в залежності від того, чи дійсно відносини в них мають відношення до конкретного класу.

Останнім етапом роботи алгоритму є рангова оцінка відносин на основі надмірності тексту. На цьому етапі відбувається нормалізація (відхиляються несуттєві модифікатори) і об'єднуються однакові відносини. Потім підраховується кількість різних речень, в яких присутні відносини, і призначаються ймовірності кожному відношенню.

При застосування такого підходу майже не потрібні марковані дані для навчання, а також він не вимагає маркування кожної відносини, що нас цікавить, вручну. Замість цього він враховує всі можливі типи відносин.

Однак при такому підході продуктивність системи багато в чому залежить від того, наскільки добре побудовані обмеження і евристики та відносини не так нормалізовані, як у підходах з попередньо заданими типами відносин.

Таким чином було описано усі основні підходи до видобування відносин. Після детального огляду цих підходів можна сказати, що основним завдання при добуванні відносин стало отримання необхідних навчальних даних. Тому підхід слабо контрольованого видобування відносин є дуже перспективним, оскільки дані навчання не обов'язково повинні бути ідеальними.

Розробка нових підходів до слабо контрольованих функцій маркування є предметом поточних досліджень. Однак хоч слабо контрольований або неконтрольований підходи і мають свої переваги, все

ж таки продуктивність таких підходів цілком залежить від того, як ретельно побудовані обмеження в системі та результат роботи такої системи може бути не дуже точним. Тому підходи, що потребують навчальні дані, а також підхід із заданням конкретних патернів, досі залишаються ефективною основою для вирішення задачі видобування відносин з природно-мовних текстів.

4 ПРАКТИЧНЕ ЗАСТОСУВАННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ ДОСЛІДЖЕНЬ

4.1 Обгрунтування вибору програмного середовища та набору даних

Для реалізації методів обробки природно-мовних текстів, а також для побудови графу знань було обрано мову Python, бо саме вона включає бібліотеки, необхідні для вказаних вище задач.

Першою з таких бібліотек є spaCy. Це безкоштовна бібліотека з відкритим вихідним кодом для розширеної обробки природної мови (NLP) в Python. Бібліотека spaCy розроблена спеціально для виробничого використання і допомагає створювати додатки, що обробляють і «розуміють» великі обсяги тексту. Бібліотеку spaCy можна використовувати для побудови систем видобування інформації або розуміння природної мови або для попередньої обробки тексту для глибокого навчання [24].

Ця бібліотека є дуже важливим та зручним рішенням для наступного типу задач:

– токенізація. Сегментування тексту на слова, розділові знаки тощо;

– тегування частини мови (POS). Присвоєння токенів типів слів, таких як дієслово або іменник;

– аналіз залежностей. Призначення синтаксичних міток залежностей, що описують відносини між окремими токенами, такими як суб'єкт або об'єкт;

– лематизація. Призначення основних форм слів. Наприклад, лема «було» – це «бути», а лема «щури» – «щур»;

– визначення меж речення. Пошук і сегментування окремих речень;

– розпізнавання іменованих сутностей (NER) Маркування іменованих «реальних» об'єктів, таких як люди, компанії та місцезнаходження;

– зв'язування сутностей (EL) Усунення неоднозначності текстових сутностей;

– подібність. Порівняння слів, фрагментів тексту і документів та їх схожості між собою;

– класифікація тексту. Призначення категорій або міток всьому документу або його частин;

– зіставлення на основі правил. Пошук послідовностей токенов на основі їх текстів і лінгвістичних анотацій, аналогічно регулярними виразами.

У той час як деякі функції spaCy працюють незалежно, інші вимагають завантаження навчених конвеєрів, які дозволяють spaCy передбачати лінгвістичні анотації, наприклад, чи є слово дієсловом або іменником. Навчений конвеєр може складатися з декількох компонентів, які використовують статистичну модель, навчену на помічених даних. В даний час spaCy пропонує навчені конвеєри для безлічі мов, які можна встановити як окремі модулі Python. Пакети конвеєрів можуть відрізнятися за розміром, швидкості, використання пам'яті, точності і містяться в них даними.

У даній роботі було використано модель конвеєру «en_core_web_sm», що надає можливість обробляти англійський текст. Зазвичай процесс обробки тексту включає такі етапи: лінгвістичні анотації, токенізацію, тегування залежностей і частин мови , розпізнавання іменованих сутностей.

spaCy надає безліч лінгвістичних анотацій для кращого розуміння граматичної структури тексту. Це включає типи слів, таких як належність до частини мови,а також анотування того, як слова пов'язані один з одним. Навіть якщо документ обробляється – наприклад, розділений на окремі слова і анотований – він як і раніше містить всю інформацію джерела, таку як, пробільні символи. Таким чином spaCy має допоміжні засоби, що запобігають втраті інформації.

Під час обробки spaCy спочатку токенізує текст, тобто сегментує його на слова, розділові знаки тощо. Це робиться шляхом застосування правил, специфічних для кожної мови. Якщо є збіг, застосовується правило, і токенізатор продовжує цикл, починаючи зі знову розділених підстрок. Таким чином, spaCy може розділяти складні, вкладені токени, такі як комбінації абревіатур і декількох розділових знаків. Хоча правила пунктуації зазвичай досить загальні, виключення токенізатора сильно залежать від специфіки окремої мови. Ось чому у кожного доступного мови є свій власний підклас, наприклад англійська або німецька, який завантажується в списки жорстко закодованих даних і правил винятків. Приклад роботи токенізатора зображено на рисунку 4.1.

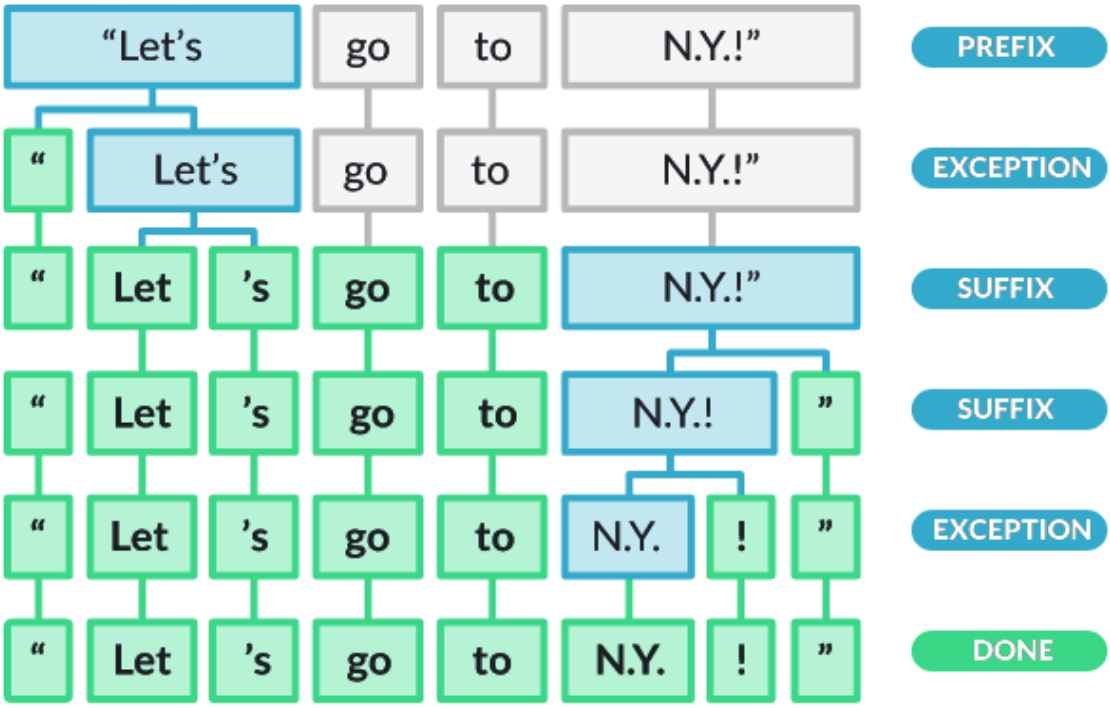

Рисунок 4.1 – Приклад роботи токенізатора spaCy

Після токенізації spaCy може аналізувати і тегувати поданий текст. Тут на допомогу приходить навчений конвеєр і його статистичні моделі, які дозволяють spaCy робити прогнози про те, який тег або мітка найбільш ймовірно застосовні в даному контексті. Навчений компонент включає в

себе виконавчі дані, які створюються шляхом демонстрації системи достатньої кількості прикладів, щоб вона могла робити прогнози, узагальнені для всієї мови – наприклад, слово, наступне за «the» в англійській мові, швидше за все, є іменником.

Також spaCy може розпізнавати різні типи іменованих сутностей в документі, запитуючи у моделі прогноз. Іменована сутність – це «реальний об'єкт», якому присвоєно ім'я, наприклад людина, країна, продукт або назва книги. Це дозволяє моделі віднести об’єкти реального світу до конкретного класу. spaCy зберігає велику кількість таких класів, а також додає нові. Оскільки моделі є статистичними і сильно залежать від прикладів, на яких вони були навчені, це не завжди працює ідеально. Приклади промаркованих токенів та іменованих сутностей зображено на рисунках 4.2 та 4.3.

TEXT	LEMMA	POS	TAG	DEP
is	be	AUX	VBZ	aux
looking	look	VERB	VBG	ROOT
at	at	ADP	IN	prep
buying	buy	VERB	VBG	pcomp

Рисунок 4.2 – Приклад результатів тегування відносин та POS

TEXT	START	END	LABEL	DESCRIPTION
Apple	0	5	ORG	Companies, agencies, institutions.
U.K.	27	31	GPE	Geopolitical entity, i.e. countries, cities, states.
$1 billion	44	54	MONEY	Monetary values, including unit.

Рисунок 4.3 – Приклад результатів NER

Конвеєр, що використовуються під час обробки тексту, зазвичай включає в себе тегер, лемматізатор, синтаксичний аналізатор і распознавач об'єктів. Кожен компонент конвеєра повертає оброблений документ, який потім передається наступному компоненту (рис. 4.4). Можливості конвеєра обробки завжди залежать від компонентів, їх моделей і того, як вони були навчені. Наприклад, конвеєр для розпізнавання іменованих сутностей повинен включати навчений компонент розпізнавання іменованих сутностей зі статистичною моделлю і вагами, які дозволяють йому робити прогнози міток сутностей.

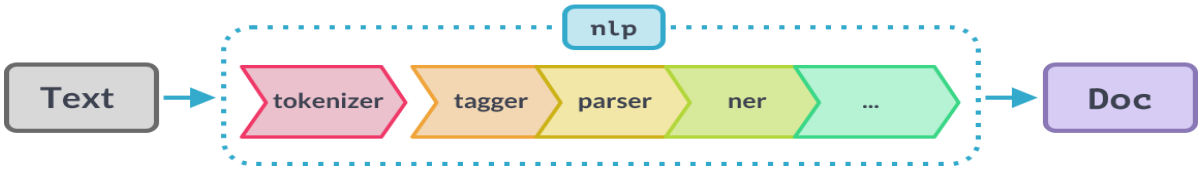

Рисунок 4.4 – Процес роботи конвеєра у spaCy

Окрім основної бібліотеки для обробки природної мови були також використані бібліотеки для візуалізації результатів роботи програмного додатку, а саме бібліотека networkX, шо призначена для роботи з графами та іншими мережевими структурами, та бібліотека matplotlib – комплексна бібліотека для створення статичних, анімованих та інтерактивних візуалізацій в Python.

Також сам програмний додаток було запущено та протестовано за допомогою засобів GoogleColab. Colaboratory, або скорочено Colab, – продукт компанії Google Research. Colab дозволяє будь-кому писати і виконувати довільний код Python через браузер і особливо добре підходить для машинного навчання, аналізу даних і дослідницької діяльності. Це безкоштовний хмарний сервіс на основі Jupyter Notebook, що надає все необхідне для машинного навчання прямо в браузері, дає безкоштовний доступ до неймовірно швидких GPU і TPU.

4.2 Опис алгоритму та демонстрація роботи програмного додатку

Першим чином на початку роботи додатку потрібно завантажити текстове джерело, звідки далі будут видобуватися сутності та відносини для побудови вузлів та ребер графу знань, відповідно. У роботі буде використано набір речень, що містить інформацію зі статей у Вікіпедії, пов’язаних з фільмами. У наборі міститься близько 4300 речень з більш ніж 500 статей у Вікіпедії.. Приклад речень, що були видобуті зі статей на Вікіпедії зазначено на рисунку 4.5.

```
256                  prometheus  and alien: covenant  address extraterrestrial themes.
3849    the entire compilation is not bad, yet there is nothing unique about the sound.
3664                         as if hell erupted through the pavement and kept on going.
649                                actors usually have their own separate call times.
3049                                                 i wanted to dive to the shipwreck.
Name: sentence, dtype: object
```

Рисунок 4.5 – Приклад текстового джерела

Наступним чином виглядають залежності у реченні у одному з наведених речень (рис. 4.6). З цього видно, що наведений приклад речення містить один суб’єкт (actors) та один об’єкт (times).

```
actors ... nsubj
usually ... advmod
have ... ROOT
their ... poss
own ... amod
separate ... amod
call ... compound
times ... dobj
```

Рисунок 4.6 – Приклад маркування залежностей в реченні

Для видобування елементів (сутностей, та відносин) з речень буде використано підхід, заснований на граматичних зв’язках між ними.

Основна ідея буде полягати в тому, щоб пройти речення і виділити суб’єкт і об'єкт, коли вони зустрічаються. Однак є кілька проблем – сутність може охоплювати кілька слів, наприклад, «червоне вино», і парсери залежностей позначають тільки окремі слова як суб'єкти або об'єкти. Отже для подолання цієї проблеми було створено функцію, що працює в декілька етапів.

Першим етапом роботи функції буде перебор токенів в реченні. Спочатку перевіряється, чи є токен знаком пунктуації. Якщо так, то цей токен будет проігноровано та процес обробки перейде до наступного токену. Якщо токен є частиною складеного слова (тег залежності = «складений»), то його буде збережено у відповідній змінній, що зберігає текст, пов'язаний з суб'єктом або об'єктом. Складене слово – це комбінація кількох слів, поєднаних в слово з новим значенням (наприклад, «Футбольний стадіон»). Видобування складених слів є можливим завдяки тегам залежностей, які додає до слів spaCy. Коли в реченні буде зустрічатися суб'єкт або об'єкт, до нього будуть додаватися токени, що є частями цього складеного слова. Те ж саме буде відбуватися зі словами-модифікаторами, такими як «красива сорочка», «великий будинок» тощо.

Після того як були витягнути частини складеного слова та модифікатори починається видобування суб’єкту та об’єкту. Коли функція знаходить один з цих елементів, то додає до них слова модифікатори та частини складеного слова, якщо їх має знайдений елемент. Після цього функція отримає кортеж, що містить суб’єкт та об’єкт, та продовжує роботу з наступними реченнями. У результаті буде отримано пари сутностей для кожного речення (рис. 4.7).

```
[['he', 'real  passion'],
 ['directors', 'other'],
 ['production designer', 'eight  films'],
 ['so  we', 'substantial  piece'],
 ['harry potter series', 'four  composers'],
 ['desplat', 'harry potter'],
 ['', '2  2011'],
 ['they', 'trumps'],
 ['shouts', 'last  time'],
 ['harry potter', 'orphaned unkind aunt'],
 ['however  part', '2  2d'],
 ['2017  johns', 'january'],
 ['principal  photography', 'august'],
 ['david ayer', 'director'],
 ['', 'bruce wayne barry allen'],
 ['justice league', 'worldwide  november'],
 ['aquaman', 'october'],
 ['production', 'august'],
 ['villain mister mind', 'credits  scene'],
 ['month  sequel', 'san diego comic con']]
```

Рисунок 4.7 – Результати видобування сутностей

Після видобування пар сутностей приходить черга відносин між ними. Основна ідея роботи функції для видобування відносин полягає в тому, що присудок насправді є головним дієсловом в реченні. Наприклад, у реченні «Шістдесят голлівудських мюзиклів було випущено в 1929 році» дієслово «випущено в», і це те, що буде використовуватися в якості предиката для трійки, отриманої з цього речення.

Під час роботи функції використовується певний патерн. Патерн, визначений у функції, намагається знайти ROOT-слово або головне дієслово в реченні. Після визначення ROOT паттерн перевіряє, чи слідує за ним прийменник («prep») або агентне слово. Якщо так слово дійсно присутнє, то воно додається до ROOT слова. Нижче приведено результати роботи функції, на яких показано 20 відносин, що найчастіше зустрічаються у текстовому корпусі речень з Вікіпедії (рис. 4.8).

is	370
was	297
released on	87
include	73
are	71
were	71
released	40
's	38
composed by	35
have	31
has	31
became	31
become	29
released in	27
included	26
produced	22
called	22
considered	20
made	20
had	20

Рисунок 4.8 – Видобуті з текстового корпусу відносини

Як видно на рисунку вище, найчастіше зустрічаються відносини типу «А є Б» і «А було Б». Однак було видобуто чимало відносин, які буле більше пов'язані із загальною темою текстового корпусу речень Вікіпедії – екосистема навколо фільмів. Деякі з прикладів таких відносин: «скомпазовано», «випущено», «створено», «написано кимось» та інші.

Але все ж таки в результаті роботи функцій для витягу сутностей та відносин було витягнуто чимало малозначущих, для основної теми текстового джерела, структур. У сутностях це були різноманітні нерелевантні слов, як «воно», «він», «вони» тощо. У відносинах у свою чергу з’являлися загальні відносини, просто через те, що це найпоширеніші слова. Такими словами виступали структури «є», «був», «були» та інші. Однак у графі знань нам потрібні більш значущі та конкретні поняття, що будут добре відображати основний сенса витягнутого з тексту знання.

Для отримання більш корисних сутностей та відносин потрібно вжити наступні заходи для передобробки тексту. Першим чином з тексту будуть витягнуті відносини, що не несуть ніякої конкретної інформаційної цінності. Це буде реалізовано шляхом витягу з тексту так званих шумових слів. Шумові слова – це англійські слова, які не додають особливого сенсу реченню . Їх можна спокійно ігнорувати, не жертвуючи сенсом речення. Результати видобування відносин з передоброблених речень, що допомагають усунути з них шум зображено на рисунку 4.9. На рисунку представлено 15 відносин, що найчастіше зустрічаються у тексті.

released	174
directed	50
produced	48
film	44
include	38
made	37
began	33
used	33
composed	31
shot	26
introduced	24
written	22
included	21
set	20
films	18

Рисунок 4.9 – Відносини, що найчастіше зустрічаються в тексті.

Далі для отримання більш значущих сутностей буде використано алгоритми розпізнавання іменованих сутностей бібліотеки spaCy. Процес видобування буде організовано за допомогою класів, що зберігаються у навченій моделі бібліотеки. Це дозволить залишити тільки ті сутності, що несуть у собі інформацію про реально існуючі об’єкти та поняття. Після видобування пари іменованих сутностей було розділено для подальшої побудови графа на 2 групи («source» і «target»). На рисунку 4.10 зображено по 10 сутностей кожної з груп, що найбільш часто зустрічаються у тексті.

Source entities		Target entities	
film	64	warner bros	7
khan	10	national film awards	5
first film	9	surviving studio india	4
movie	6	film	4
indian film	6	october	3
two certificate awards	5	decades	3
series	5	positive box office success	3
music	5	undisclosed roles	3
films	4	2 may	3
kanchivaram	4	international film festival	3

Рисунок 4.10 – Видобуті з передобробленого тексту сутності.

Останнім кроком у роботі буде безпосередньо побудова графа знань. В цій задачі дуже допоможе побудова датафрейму, який буде включати в себе усі сутності та відносини графу. Далі для побудови графу використовується бібліотека network для створення мережі з цього фрейму даних. У результаті будет побудовано орієнтований граф, тобто граф, де зв'язок між будь-якою пов'язаною парою вузлів не є двостороннім, а тільки від одного вузла до іншого.

У якості прикладу буде візуалізовано граф знань для речення, що містить у собі згадку американского актора, режисера та сценариста Девіда Аєра (рис. 4.11). Згадку імені актора у реченні виглядає так «… , with david ayer confirmed as director.»

Рисунок 4.11 – Приклад вихідного графу знань для сутності «Девід Аєр»

Також у якості прикладу було побудовано граф знань для цільової сутності, якою є таке поняття як «індійський фільм» (рис. 4.12).

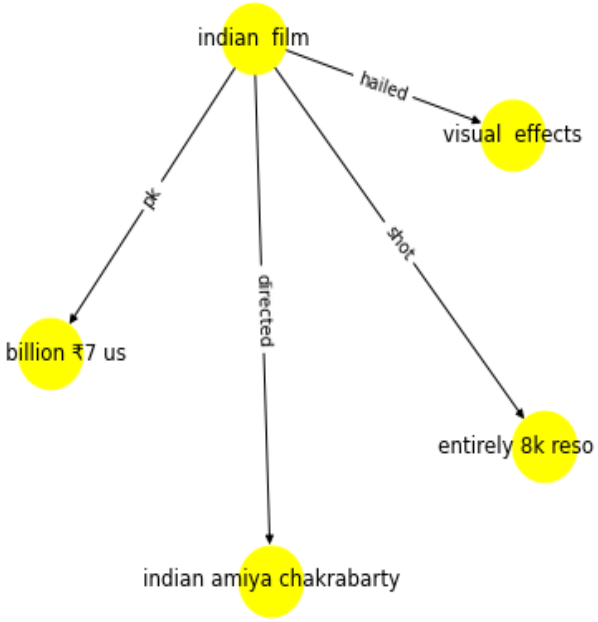

Рисунок 4.12 – Приклад вихідного графу знань для сутності «індійський фільм»

Через те що граф, який візуалізує усі вузли та ребра, буде виглядати як цілковитий безлад, було прийнято рішення візуалізувати тільки декілька найважливіших відносин, що напряму відносяться до тематики текстового корпусу, тобто до фільмів. Далі будуть візуалізовано графи для таких ключових відносин як «скомпазовано кимось», «написано кимось», «випущено у», бо саме ці відносини добре відображають тематичне наповнення речень (рис. 4.13, 4.14, 4.15).

Рисунок 4.13 – Відносина «скомпазовано (кимось) »

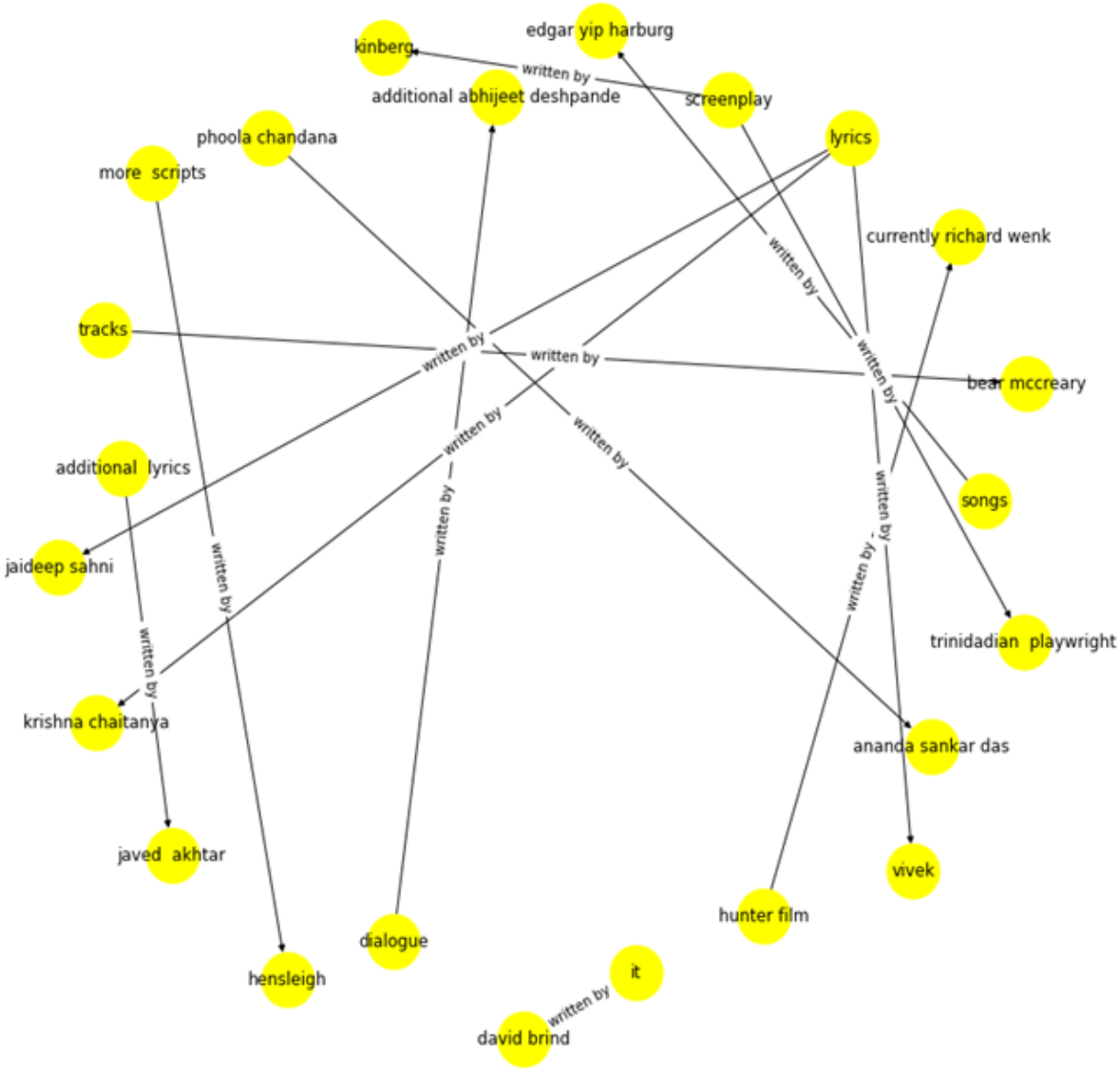

Рисунок 4.14 – Відносина «написано (кимось) »

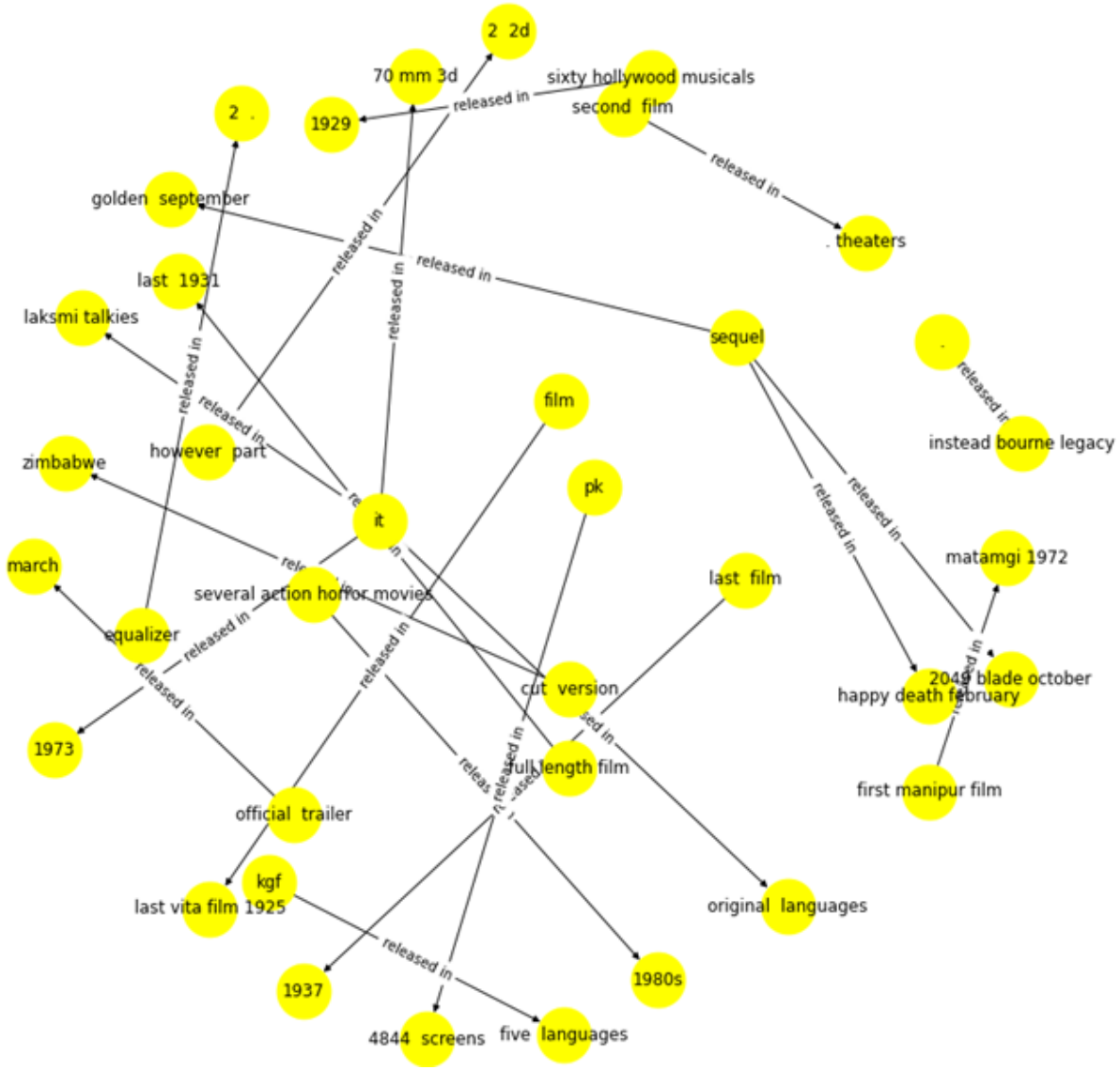

Рисунок 4.15 – Відносина «випущено у »

Як видно на представлених вище рисунках з текстового джерела було витягнуто велику кількість різноманітних триплетів, що представляють собою деякі знання з області кінематографу. Наприклад триплет «60 голлівудських мюзиклів», «випущено у», «1929» чітко відображає знання про конкретну подію. У зв'язку з цим можна зробити висновок, що програмний додаток коректно відображає знання.

ВИСНОВКИ

В даній кваліфікаційній роботі було показано, як використання програмування може покращити видобування знань з природно-мовних текстів.

Під час виконання кваліфікаційної роботи практики було проаналізовано технічну літературу та інтернет джерела на тему видобування знань, актуальності використання моделі графів знань та методів обробки природної мови. На основі проведеного дослідження були встановлені методи обробки природної мови, за допомогою яких можна організувати процес вилучення сутностей, відношень та семантичних залежностей з природно-мовних текстів, а також було проведено дослідження моделі подання знань, такої як графи знань.

Для подальшої розробки програмного рішення було проведено огляд існуючих додатків, що реалізують витяг знань з природно-мовних текстів та бібліотек для побудови графів знань. Після чого було виявлено переваги та недоліки кожного з них. На основі проведених досліджень було сформовано постановку задачі, а саме розробку програмного додатку для видобування знань та побудові на їх основі графу знань.

Були розроблені компоненти додатку для витягу знань з природно-мовних текстів та формування на їх основі графу знань. За допомогою даної системи, можна вилучати інформацію про об’єкти реального світу, факти та події, які у результаті роботи додатку будуть представлені у вигляді структурованої моделі знань.

Також був розроблений зручний графічний інтерфейс програми, який наглядно відображає усі дії, що виконувалися у процесі роботи програмного додатку.

Розроблене програмне забезпечення буде актуальним, в першу чергу, в інформаційних системах, що виконують функції семантичного пошуку та рекомендацій, тобто систем основною метою яких є пошук та надання

користувачу найбільш коректної відповіді або рекомендації на його запит. Саме знання в цій концепції стають основною будівельною одиницею, що дозволяє системі робити висновки грунтуючись на фактах, як це робить людина. Але можливості використання додатку не закінчуються лише на задачах надання конкретної відповіді на запит. Також розроблений додаток може використовуваится у різних аналітичних система, задачах текстового аналізу та побудові інтелектуального контенту, бо модель графу знань надає найбільш природній підхід для роботи зі знаннями та представляє собою зручне рішення для їх візуалізації і звітності. Програмний додаток було реалізовано на мові програмування Python за допомогою бібліотеки для обробки природної мови spaCy та бібліотек для візуалізації графу знань networkX (бібліотека для побудови графів) та matplotlib (бібліотека для створення статичних, анімованих та інтерактивних візуалізацій) .

Розроблений веб-додаток відповідає сформульованим вимогам завдання, таким чином, завдання на кваліфікаційну роботу виконано в повному обсязі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Brachman R. J., Levesque H. J. Knowledge Representation and Reasoning. Elsevier, 2004. 413 p.

2. Gomez-Perez J. M., Garcia S. A. A Practical Guide to Hybrid Natural Language Processing. Springer, 2020. 281 p.

3. Люгер Д. Ф. Искусственный интеллект: стратегии и методы решения сложных проблем. 4-е издание.: Пер. с англ. М.: Издательский дом «Вильямс», 2003. 864 с.

4. Рассел С., Норвиг П. Искусственный интеллект: современный подход, 2-е изд.: Пер. с англ. М.: Издательский дом «Вильямс», 2006. 1408 с.

5. Randall D., Howard S., Peter S. What Is a Knowledge Representation?. AI Magazine. 1993. January 14. P 17-33.

6. Frank V. H., Lifschitz V., Porter B. Handbook of Knowledge Represantation. The Netherlands, AE Amsterdam, Elsevier, 2008. 1035 p.

7. Chein M., Muginer M.L. Graph-based Knowledge Representation. Computational Foundations of Conceptual Graphs. New York, NY: Springer, 2009. 425 p.

8. Рябова Н.В. Онтологічний підхід до моделювання інтелектуальних інформаційних середовищ. Матеріали міжнар. наук. конф. «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» ISDMCI’2017, 22-26 травня 2017 р., Залізний Порт, Україна, Херсон, ХНТУ, 2017. 2 с.

9. Рябова Н.В., Золотухін О.В. Побудова онтологічних баз знань із застосуванням методів інтелектуальної обробки текстів Матеріали XIV Міжнар. наук. конф. «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту» ISDMCI’2018, 21-27 травня 2018 р., Залізний Порт, Україна, Херсон, ХНТУ, 2018. С. 200-201.

10. Рябова Н.В., Золотухін О.В. Обучение онтологий на основе методов интеллектуального анализа текстовых документов. Сучасні проблеми та

перспективи соціально-економічного, інформаційного та науково-технічного розвитку підприємств України: Всеукр. науково– практ. конф. / Маріуполь (2 січня 2018 р.): тези доп. / редкол.: М.Г. Белопольський [та ін.]; ДВНЗ «ПДТУ». Маріуполь: ПДТУ, 2018. С.306-309.

11. Kendal S.L., Creen M. An Introduction to Knowledge Engineering. Springer, 2007. 294 p.

12. Janev V., Graux D.,Jabeen H., Sallinger E. Knowledge Graphs and Big Data Processing. Springer,2020. 212 p.

13. Heiko P. Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web Journal, IOS Press. 2016. November 28. 23 p.

14. Yan J., Wang C., Cheng W., Gao M. A retrospective of knowledge graphs. Frontiers of Computer Science. 2016. September 26. 21 p.

15. Knowledge Graph and Semantic Computing: Knowledge Computing and Language Understanding / Zhao J. et al. 4th China Conference, CCKS 2019, Hangzhou, China, 2019, 160 p.

16. Wu J., Xie R., Liu Z., Sun M.. Knowledge Representation via Joint Learning of Sequential Text and Knowledge Graphs. 4th China Conference, CCKS 2019, Hangzhou, China, 2019. 10 p.

17. Ehrlinger L., Wöß W. Towards a Definition of Knowledge Graphs. Leipzig, Germany, Joint Proceedings of the Posters and Demos Track of 12th International Conference on Semantic Systems – SEMANTiCS2016 and 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS16), 2016. 5 p.

18. Hogan A., Blomqvist E., Cochez M. Knowledge graphs. ACM Computing Surveys Journal, March 2020. 136 p.

19. Text to Knowledge Graph: Knowledge Extraction Pipeline with Transformers. URL: https://medium.com/swlh/ text-to-knowledge-graph-683002cde6e0 (дата звернення: 14.04.2021)

20. Дмитриев Д.Ю. Исследование текстологических методов построения графов знаний.: тез. докл. 25-го Міжнародного молодіжного

форуму «Радіоелектроніка і молодь у XXI столітті». Харків: ХНУРЕ, 2021. Том 6. С. 29-30.

21. Eftimov T., Koroušić B. S., Korošec P. A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations. 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, 2016. 23 p.

22. Jie T., Mingcai H., Duo Z. Information Extraction: Methodologies and Applications. Emerging Technologies of Text Mining: Techniques and Applications / Ed. Prado A. H., Idea Group Inc., Hershey, USA, 2007. P.1-33.

23. Aoran L., Xinmeng W., Wenhuan W. A Survey of Relation Extraction of Knowledge Graphs. Web and Big Data / Ed. S. Jensen, Cyrus S., Xiaochun Y. Chengdu, China, August 3, 2019. P.52-66.

24. spaCy 101: Everything you need to know. URL: https://spacy.io/usage/spacy-101 (дата звернення: 17.04.2021)