МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет радіоелектроніки
Факультет Комп'ютерних наук
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

другий(магістерський)
(рівень вищої освіти)

Дослідження методів структурованого пошуку інформації в розподілених сховищах даних

Виконав:
студент 2 курсу групи ІПЗм-20-2
Погуляєв Ю. С.
(прізвище, ініціали)

Спеціальність	121 – Інженерія програмного забезпечення
Тип програми	Освітньо-наукова
Керівник	доц. Лановий О. Ф.

(посада, прізвище, ініціали)

Допускається до захисту

Зав. Кафедри _____ З.В. Дудар

2022

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва)

Тип програми освітньо-наукова програма

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«__»_____202_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента Погуляєва Юрія Сергійовича
(прізвище ім'я по батькові студента)

1. Тема роботи «Дослідження методів структурованого пошуку інформації в розподілених сховищах даних»

затверджена наказом університету від «__»_____202_ р. №___

2. Термін подання студентом роботи до екзаменаційної комісії «15» травня 2022 р.

3. Вихідні дані до роботи електронні ресурси за обраною тематикою, методичні вказівки до виконання кваліфікаційної роботи магістра, пояснювальна записка.

4. Перелік питань, що потрібно опрацювати в роботі аналіз предметної області, постановка задачі, огляд існуючих методів та патентів, опис розробленої системи, математичне моделювання, опис проектних рішень, опис та аналіз експериментів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Огляд наукової та патентної літератури	15.02.2022	виконано
2	Постановка задачі	20.02.2022	виконано
3	Математичне моделювання	04.03.2022	виконано
4	Збір тестових даних	10.03.2022	виконано
5	Проектування та реалізація програмної системи	01.04.2022	виконано
6	Розробка плану проведення експериментальних досліджень	06.04.2022	виконано
7	Проведення експериментальних досліджень	12.04.2022	виконано
8	Аналіз отриманих результатів	16.04.2022	виконано
9	Підготовка пояснювальної записки	06.05.2022	виконано
10	Підготовка презентації та доповіді	11.05.2022	виконано
11	Перевірка на плагіат	11.05.2022	виконано
12	Нормоконтроль	11.05.2022	виконано
13	Архівування	16.05.2022	виконано
14	Попередній захист	18.05.2022	виконано
15	Допуск до захисту у зав. кафедри	18.05.2022	виконано

Дата видачі завдання 17 січня 2022 р.

Студент _____

(підпис)

Керівник роботи _____ доц. Лановий О. Ф.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Кваліфікаційна робота магістра містить: 66 с., 33 рис., 8 табл., 19 джерел.

ПОШУК, СХОВИЩЕ, АВТОМАТИЧНЕ АНОТУВАННЯ, СТРУКТУРОВАНИЙ ПОШУК, КРИТЕРІЙ, УМОВА

В роботі в якості об’єкта дослідження розглядається процес організації структурованого інформаційного пошуку серед розподілених сховищ інформації.

Предметом дослідження є методи структурованого пошуку в розподілених сховищах даних.

Метою дослідження є аналіз існуючих методів пошуку інформації, їх порівняння, виділення їх переваг та недоліків з метою урахування усіх ключових моментів для проектування схеми пошуку, яка буде задовольняти наведеним системним критеріям.

Результатом дослідження є напрацьовані проектні та математичні дані, опис пошукової схеми, експериментальні дані та їх аналіз, а також рекомендації до використання.

SEARCH, REPOSITORY, AUTOMATIC ANNOTATION, STRUCTURED SEARCH, CRITERION, CONDITION

The process of organizing a structured information search among distributed information repositories is considered as an object of study.

The subject of research is the methods of structured search in distributed data storage facilities.

The purpose of the study is to analyze existing methods of finding information, their comparison, highlighting their advantages and disadvantages to take into account all key points for designing a search scheme that will satisfy the systemic criteria given.

The result of the study is the design and mathematical data, the description of the search scheme, experimental data and their analysis, as well as recommendations for use.

Я, Погуляєв Юрій Сергійович ,

(прізвище, ім’я, по-батькові)

студент(ка) групи ІПЗм-20-2 здобувач вищої освіти на другому (магістерському) рівні

кафедра програмної інженерії ,

(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему

Дослідження методів структурованого пошуку інформації в розподілених сховищах даних,

що буде представлена до ЕК для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(а) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ВСТУП

Незважаючи на досить молодий вік такого поняття як інформаційний пошук, проблема, яку він вирішує, була і залишається актуальною щонайменше останні 4500 років. Зважаючи на те, що перші письмові артефакти цивілізацій Близького Сходу та давнього Єгипту (скупчення шумерських глиняних табличок та єгипетських папірусів), які вважаються найбільш давніми відомими на сьогоднішній день попередниками бібліотек, датуються 27-25 сторіччями до нашої ери [1], то історія методів пошуку інформації в таких сховищах даних є значно молодшою – виникнення перших прототипів бібліотечних каталогів відносять до 8-7 сторіччя до нашої ери і пов’язують в першу чергу з легендарною бібліотекою ассирійського царя Ашурбанапала. Тому, починаючи з 8-7 сторіччя до нашої ери, проблема створення та розробки як нових сховищ даних, так і нових методів пошуку інформації в таких сховищах є актуальною [2].

Незважаючи на таку довгу історію існування як сховищ даних, так і методів пошуку інформації, саме наукове поняття «інформаційний пошук» є відносно молодим, оскільки в більшості випадків використовується саме в сфері пошуку інформації з використанням електронно-обчислювальних машин та персональних комп’ютерів. Ідея розробки такого апаратно-програмного забезпечення виникла ще в 1945 році та була викладена в статті «As We may think» американським вченим та розробником аналогових ЕОМ Веніваром Бушем.

Однак одним з перших наукових означень терміну «інформаційний пошук» є означення, наведене в 1948 році американським вченим та інженером Кельвіном Муерсом спочатку в рамках наукового видання, а потім і в рамках власної докторської дисертації.

Так, згідно його означення, «проблема спрямування користувача до інформації, що зберігається, частина з якої може бути йому невідома, є проблемою інформаційного пошуку».

З розвитком інформаційних технологій та обчислювальних систем зростали і обсяги даних, що зберігалися та\або оброблялися не власноруч, а за допомогою

аналогових чи цифрових ЕОМ. Найбільшу актуальність проблема інформаційного пошуку отримала, починаючи з початку 1990-х років. Цьому передувало декілька факторів.

По-перше, вже з другої половини 1970-х років частина населення (в першу чергу Сполучених Штатів) отримала можливість придбати ЕОМ персонально, що і стало однією з вагомих віх у перетворенні ЕОМ саме у ПК – персональний комп'ютер. Прикладами перших комерційно вдалих моделей персональних обчислювальних машин прийнято вважати Apple II та Apple Macintosh.

По-друге, в 1980-х роках вже були розповсюджені більш компактні моделі пристроїв для зв'язку та обміну інформацією – спочатку це були пейджери, а потім світ побачив і стільникові телефони.

По-третє, починаючи з 1989 року, ідея глобальної мережі (в подальшому відома як Інтернет) отримала свою практичну реалізацію у вигляді проекту британського вченого, який працював в швейцарському інституті ЦЕРНу Тіма Бернерса-Лі – всесвітня павутина (World Wide Web).

Саме сукупність наведених трьох чинників призвела до інформаційного вибуху – якщо в 1986 році загальний об'єм інформації, що зберігалася на електронних пристроях, оцінювався в $2.6 * 10^{18}$ байт, то вже в 2007 році ця цифра складала $295 * 10^{18}$ байт [3].

Таким чином, зі стрімким зростанням кількості інформації як на окремих електронних носіях, так і таких, що розповсюджуються мережею Інтернет, постала гостра необхідність створення як систем, так і нових методів пошуку інформації, причому тепер не лише для вузькоспеціалізованого доступу.

В роботі в якості об'єкта дослідження розглядається процес організації структурованого інформаційного пошуку серед розподілених сховищ інформації.

Предметом дослідження є методи структурованого пошуку в розподілених сховищах даних.

У якості методів дослідження будуть використані наступні:

– аналіз та синтез;

– індукція та дедукція;

– ідеалізація;

– математичні методи (методи моделювання).

Метою дослідження є аналіз існуючих методів пошуку інформації, їх порівняння, виділення їх переваг та недоліків з метою урахування усіх ключових моментів для проектування схеми пошуку, яка буде задовольняти наведеним системним критеріям.

Наукова новизна полягає у висвітленні та порівнянні різних методів організації структурованого пошуку, а також у висвітленні не лише методів пошуку, а й методів розподіленого зберігання документів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Аналіз предметної області дослідження

Задача інформаційного пошуку має декілька підходів до її розв’язання. Очевидно, що завжди є найбільш простий і найбільш зрозумілий інтуїтивно, проте найгірший з точки зору часу пошуку інформації. Саме необхідність якнайшвидше знайти документи, що задовольняють пошуковому запиту, і є основним стимулом розробки спеціальних методів пошуку інформації.

Окрім цього, з розвитком персональної обчислювальної техніки та мережі Інтернет для широких мас населення пошук має бути не надто складним процесом, оскільки це впливає на здатність без значної підготовки мати можливість користуватися пошуковим сервісом. В подальшому, для розгляду та порівняння різних існуючих пошукових методів та систем, введемо до розгляду такі терміни як запит, критерій запиту та об’єкт запиту.

Під запитом в подальшому будемо розуміти певну формалізовану структуру, яка містить в собі мінімальні дані про пошук. Це може бути як просто текст, який необхідно знайти, так і складна структура, яка буде містити додаткові дані про документи, які цікавлять користувача.

Під критерієм запиту будемо розуміти набір формальних відношень, які визначають, чи відповідає документ наданому запиту.

Під об’єктом запиту будемо розуміти документ (не обов’язково текстовий), який відноситься хоча б до одного запиту за наданими критеріями.

Методи інформаційного пошуку перш за все розподіляються на неструктуровані та структуровані [4]. Така класифікація виникає саме відповідно до ознаки зручності користування пошуковою системою для непідготовленого користувача.

Семантичний пошук – це процес пошуку неструктурованої інформації, яку використовують сучасні пошукові системи для надання найбільш релевантних результатів.

Термін походить від імені розділу «семантичне», який вивчає семантичні значення. «Семантичний пошук» – це визначення, яке часто використовується щодо зусиль, які здійснюються пошуковими системами для розуміння запитів природною мовою. Однак він набагато ширший, а також включає контекст, в якому користувач розташований під час введення пошуку.

Наприклад, якщо користувач набуває слово «готелі», а попередній запит був «Париж», тобто ймовірністю того, що він шукає інформацію про готелі в Парижі. Або, якщо раніше він налаштував багато запитів на тему дикої природи і входить у запит «Ягуар», то велика ймовірність, що його цікавить тварина, а не автомобіль.

Крім того, раніше пошукові системи не розуміли поняття суб'єктів, які полягають у тому, що властивості можуть бути пов'язані з людьми, подіями та місцями. Іншими словами, Емпайр-Стейт-Білдінг – це об’єкт, який має висоту, оригінальну архітектуру, дату завершення будівництва, пов'язані фотографії тощо. Тому, якщо користувач спочатку шукає за запитом «Ейфелева вежа», а потім – «Яка її висота», пошукова система покаже висоту Ейфелевої вежі.

Семантичний пошук полягає в тому, щоб забезпечити користувачеві значні результати, засновані на розумінні його нинішнього наміру. Для цього пошукові системи повинні знати не лише намір користувача, а й контекст запиту [5].

Пошукові системи значно прогресували в розумінні запитів. Зокрема, Google досяг великого успіху в цій галузі, завдяки досягненням у галузі аналізу та обробки природної мови. Функція голосового пошуку вперше з’явилася в програмі Google для iPhone у 2008 році. З тих пір компанія продовжує працювати над вдосконаленням своєї технології. Зрештою, одна з головних цілей Google – зрозуміти світ так само, як і люди.

Одним із перших кроків у цьому напрямку було створення мережі знань, концепція якої полягає в тому, щоб зрозуміти фактичні речі, а не рядки слів. Приклад такої пошукової структури наведено на рисунку 1.1 (див. с. 12).

Неструктуровані пошукові системи орієнтовані на знаходження документів, які за обраною ознакою близькі до тексту, який користувач намагається шукати. Тобто, запит в таких системах має вигляд простого тексту, який набирається

користувачем. Критерієм може виступати як точне співпадіння слів (або всього тексту), так і семантична близькість документу до наданого тексту в запиті. В першому випадку зазвичай говорять про повнотекстовий пошук (в разі, якщо потрібно знайти повноцінне входження тексту в документ), в другому – про семантичний пошук.

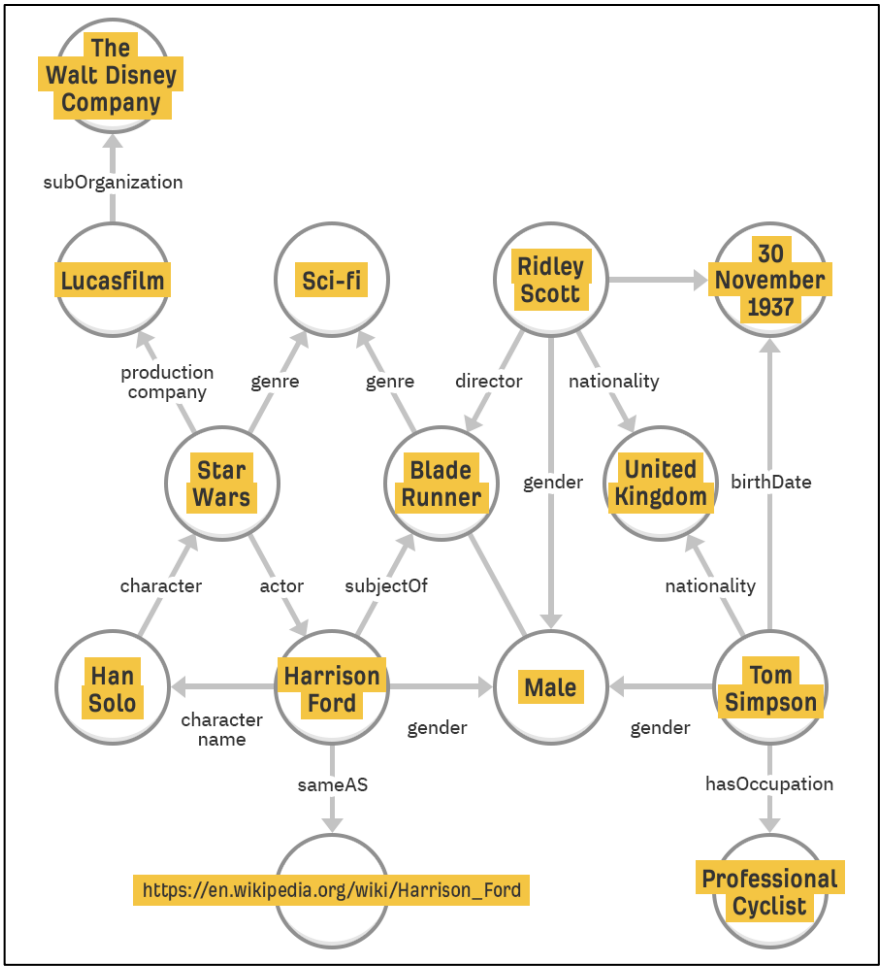

Рисунок 1.1 – опис пошукової структури вузлового типу

Неструктурований пошук має наступні переваги та недоліки. Серед переваг можна виділити відносну простоту для користувача та інтуїтивну зрозумілість: набрав пошуковий запит – отримав відповідь.

Серед недоліків можна виділити наступні:

– для семантичного пошуку – технічна складність розробки, необхідність використання або специфічних алгоритмів, або технологій машинного навчання;

– для повнотекстового пошуку – необхідність дослівно знати частину тексту, по якій планується знайти документ, що не завжди є можливим.

В якості альтернативи методам неструктурованого пошуку пропонуються методи структурованого пошуку. Вони є досить різноманітними, оскільки основні відмінності в більшості методів саме в структурі пошукового запиту. З одного боку очевидно, що чим більш розгалуженою є структура запиту – тим більша точність і тим кращий результат пошуку.

З іншого боку, зростання структури запиту призводить до двох негативних наслідків – ускладнення самого процесу пошуку (що не підходить для більшості користувачів) та зростання обсягу метаданих, які потрібно зберігати про кожен документ, адже зрозуміло, що якщо існує структура запиту – то повинна бути і відповідна структура даних про самі документи, що зберігаються в сховищі [6].

Серед методів структурованого пошуку виділяють наступні:

– адресний пошук, тобто коли в результаті оформлення структурованого запиту повертається адреса розташування шуканого документа чи документів – посилання на фізичний файл, веб-сайт, чи інший ресурс;

– документальний пошук, тобто коли об’єкт запиту – це або сам документ, або додаткові метадані про нього;

– фактографічний пошук.

Приклад структури пошукової системи можна побачити на рисунку 1.2 (див. с. 14).

Іншим аспектом розробки пошукової системи є сама організація сховища з документами. В найпростішому випадку сховище являє собою фізичний носій з даними, серед яких може проводитися пошуковий запит. Проте, зазвичай використовують деякі методи збереження даних, що дозволяють покращити продуктивність. До них відносять:

– автоматичне сортування даних всередині фізичного носія за найбільш важливими критеріями;

– горизонтальне масштабування сховища для забезпечення рівномірного навантаження на пошукову систему;

– розподілення сховища на декілька частин заздалегідь з урахуванням певних регіональних критеріїв.

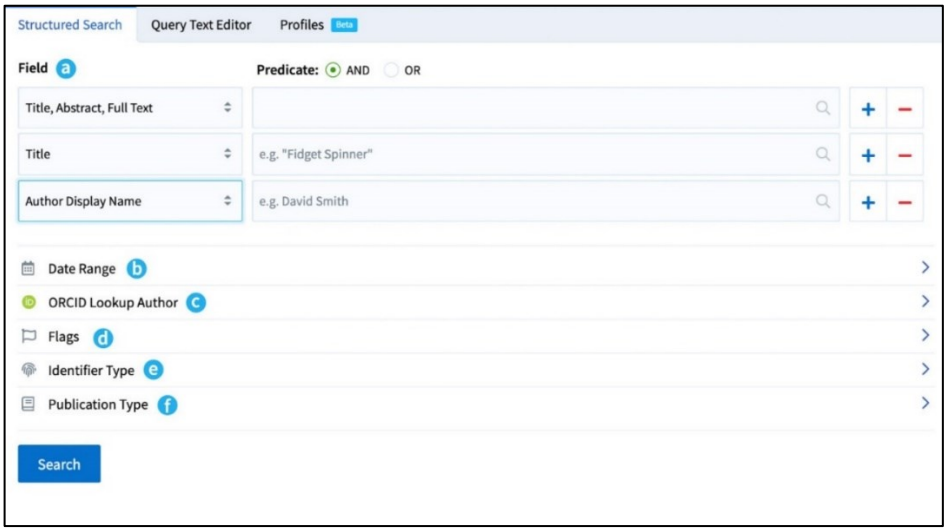

Рисунок 1.2 – приклад системи структурованого пошуку

Остання стратегія організації сховищ документів є найбільш доцільною в тому разі, коли проведено аналіз щодо використання пошукової системи і виявлено кореляцію між певною групою документів в сховищі (згруповану за деякою ознакою) та регіоном доступу. Проте ця стратегія є доцільною лише в тому випадку, коли така кореляція дійсно має місце, в інакшому випадку це може призвести до невиправданого збільшення часу відгуку пошукової системи [7].

Інформаційний пошук виконується з допомогою системи пошуку інформації (СПІ). СПІ – це набір окремих частин, пов’язаних одна з одною, розроблених для визначення будь-яких елементів інформації, яка відповідає поданому інформаційному запиту. Масив інформаційних елементів, в яких зроблена СПІ, називається пошукова маса.

СПІ поділяються на документальні та фактографічні. Документальні СПІ у відповідь на введені в них запити на інформації видаються оригінали, копії або адреси зберігання, що містять необхідну інформацію. Підклас документальних СПІ, що видають лише бібліографічні описи (БО) бажаних документів, іноді називають бібліографічними СПІ [8].

На відміну від документальних СПІ, фактичні пошукові системи розроблені для отримання безпосередньо необхідної інформації (наприклад, температура кипіння будь-якої рідини; структурні або молекулярні формули хімічних сполук з певними властивостями тощо).

Немає основних відмінностей між документальною та фактографічною СПІ. Якщо ми проводимо аналогію, то документальна СПІ відрізняється від фактичних не більше, ніж первинні наукові документи від референтних книг. Основна особливість, об'єднання документальних та фактичних пошукових систем в один загальний клас, полягає в тому, що і перший, і останній можуть видати таку та єдину інформацію, яка раніше була введена в них на запит.

Будь-яка документальна СПІ – від посібника до автоматизованого – включає такі елементи: інформаційно-дослідницька мова (ІДМ), правила перекладу текстів документів та запитів з природної мови до ІДМ, формальних правил (алгоритмів) пошукових, технічних пристроїв, що реалізують пошук, алгоритми, документи фонду (або їх адреси), записані на будь-яких пристроях інформації.

Інформаційний пошук проводиться відповідно до певних правил, що визначають стратегію пошуку, тобто. Способи досягнення оптимального результату. Стратегія інформаційного пошуку залежить від типу пошукового завдання, критеріїв видачі та характеру діалогу між споживачами інформації та СПІ. Загалом, процедура інформаційного пошуку складається з чотирьох етапів:

– роз'яснення потреби інформації та формулювання запиту;

– визначення сукупності власників інформаційних масивів;

– вилучення інформації з інформаційних масивів;

– ознайомлення користувача з отриманою інформацією та оцінкою результатів пошуку.

Найефективнішим методом пошуку документів, що містять наукову інформацію, є перегляд усіх документів. Але цей метод практично неможливий, оскільки кількість документів зазвичай занадто велика, щоб усі вони могли бути прочитані з кожним запитом на інформацію. Тому потрібно використовувати інший, менш ефективний метод, в якому інформаційний пошук складається не

відповідно до текстів самих документів, а відповідно до коротких характеристик вмісту або певних зовнішніх ознак документів. Подібна схема зображена на рисунку 1.3.

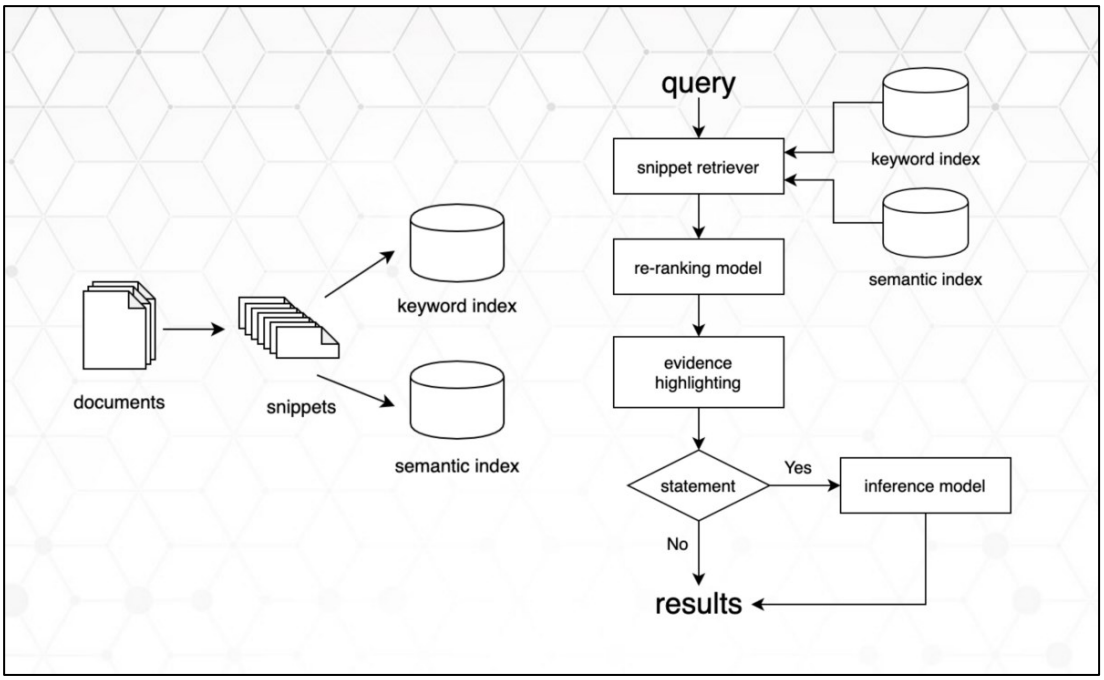

Рисунок 1.3 – схема пошукової системи з вилученням пошукових даних

Для цього кожен документ оснащений пошуковим зображенням документа (ПЗД) – характеристикою, в якій коротко виражається основний семантичний зміст документа. У формі тієї ж короткої характеристики – рецепту пошуку або пошуковому зображенню запиту (ПЗЗ) – також повинен бути сформульований запит на інформацію. Завдяки цьому процедура інформаційного пошуку може бути зведена до простого порівняння із заданою ПЗЗ. Якщо за необхідним та достатнім критерієм є збіжність з ПЗЗ, вважається, що цей документ відповідає запиту на інформацію. Порядок сформульований з точки зору однієї мови, і, крім того, в якій кожна фраза дозволяє одне і лише одне тлумачення.

У короткій формі виражається лише головний семантичний зміст документа. Тому цей метод не може гарантувати, що всі документи, що містять необхідну інформацію в бібліотеці, не можуть бути знайдені в бібліотеці. Крім того, серед

знайдених документів можуть бути ті, які насправді не відповідають на цей запит на інформацію. Ці документи утворюють так званий «пошук шуму» [8].

Важливо пам’ятати, що інформація, що міститься в наукових документах, об'єктивно підкоряється закону розсіювання. Повна та точність пошуку – це конкуруючі показники: збільшення одного з них призводить до зменшення іншого. Збільшуючи повноту пошуку, ми неминуче знижуємо його точність і навпаки, підвищуючи точність пошуку, зменшують його повноту.

Іншим напрямком розвитку пошукових систем в цілому стала концепція розподілення даних, або розподілених сховищ даних. Дана тема отримала розвиток спочатку в площині файлових систем, а з початку 1990-х – і в площині баз даних. Такий підхід до проектування файлових систем та баз даних має за мету при мінімальних змінах з точки зору користувача максимально оптимізувати структуру збереження даних відповідно до тієї задачі, що потребує розв’язання [9, 10]. Вперше подібні концепції файлових систем виникли ще в 1960-х, одним з перших представників є проект MIT Incompatible Timesharing System. Подібні системи можуть бути класифіковані за різними параметрами – узгодженістю даних (відбувається реплікація даних між різними вузлами чи ні), протоколом обміну (використовується файловий чи мережевий протокол), параметрами виділення окремих вузлів (за географічним, функціональним чи іншими критеріями).

Основними перевагами використання саме розподілених систем зберігання даних є:

– можливість автоматичної реплікації даних – це забезпечує відмовостійкість пошукової системи в разі виходу з ладу одного чи декількох вузлів збереження даних;

– можливість створення моделі збереження даних на базі існуючого підходу з організації бібліотечних каталогів – це спрощує роботу з документами, оскільки в такому випадку сховища даних ще є індикаторами того, яку спільну характеристику мають усі документи, що зберігаються у них.

1.2. Постановка задачі

Метою роботи є дослідження методів структурованого пошуку в розподілених сховищах даних.

Для проведення дослідження слід розробити схему системи пошуку, збереження та завантаження документів з урахуванням усіх переваг та недоліків, що присутні різним методам структурованого пошуку та зберігання документів. Також необхідно визначити набір метрик, згідно яких буде проводитися аналіз результатів експериментального дослідження та зробити порівняльний багатокритеріальний аналіз.

Для досягнення поставленої мети необхідно виконати наступні задачі:

– провести аналіз існуючих структурованих пошукових систем;

– розробити власну структуру пошукової системи, відштовхуючись від результатів аналітичного етапу;

– спроектувати та реалізувати програмні рішення для проведення дослідження;

– спланувати та провести експериментальне дослідження.

Результатом проведеного дослідження буде порівняння методів структурованого пошуку в розподілених сховищах даних та висновки щодо результативності методів пошуку, що порівнюються, а також опис та розробка гібридної структури пошуку документів, оцінка її ефективності та рекомендації. Для оцінки ефективності та аналізу найбільш доречно використовувати математичний апарат, що притаманний оцінці бінарних класифікаторів, оскільки пошукова система планується до проектування з використанням технологій машинного навчання, а також такі метрики дозволять досить швидко та точно оцінити основні показники системи, виявити своєчасні недоліки та надати рекомендації: які зміни можна внести задля кращої працездатності системи.

2 ОГЛЯД ІСНУЮЧИХ МЕТОДІВ ТА ПРИЙНЯТИХ ПРОЕКТНИХ РІШЕНЬ

2.1. Аналіз закордонних наукових публікацій

В результаті огляду існуючої патентної літератури для аналізу було обрано 3 запатентовані методи структурованого пошуку та збереження документів. У роботі [11] наведено принцип проведення структурованого пошуку за так званими «ключовими об'єктами» – автори пропонують таку програмну систему, де документи (в винаході розглядаються веб-сторінки) містять заздалегідь узгоджені ключові об'єкти, які власно містять інформацію, яка потрібна пошуковій системі для проведення пошуку. Структуру таких ключових об'єктів можна конфігурувати – від цього залежить точність пошукової системи. Схема роботи наведена на рисунку 2.1.

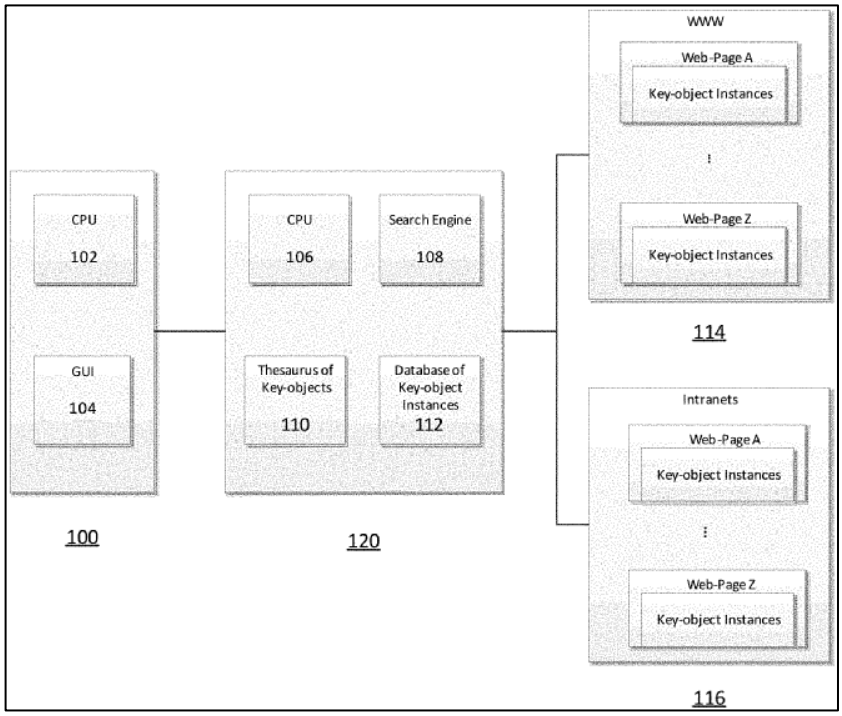

Рисунок 2.1 – схема роботи винаходу на базі ключових об'єктів

Даний патент описує більше саме структуру зберігання документів, а не сам процес структурованого пошуку. Незважаючи на це, патент досить просто описує принцип взаємодії усіх частин системи – основна складність даної схеми це

автоматична генерація або автоматичне додавання подібних ключових об’єктів до веб-сторінок, очевидним є і те, що така схема працює лише для тих документів, до яких адміністратор має доступ і якщо наприклад для централізованої системи зберігання документів такий підхід є виправданий, для мережі Інтернет (для якої і розроблялася схема) цей шлях є неоптимальним, оскільки неможливо централізовано додати такі об’єкти для кожної веб-сторінки.

Також до недоліків патентної схеми можна віднести і те, що сучасні пошукові системи більшою частиною розраховані на методи неструктурованого пошуку, тому використання довільної системи пошуку, як описується в винаході, є сумнівним та невиправданим з точки зору точності та повноти результату пошуку, що очікується.

В роботі [12] наведено принцип семантичного структурованого пошуку. Його принцип полягає в наступному: система аналізує пошуковий запит (в даному патенті лише простий текст) та перетворює його після морфемного та семантичного аналізу в відповідний числовий ключ. Далі проводиться пошук по всім документам, аналізуються лексеми та проводиться аналіз збігу тексту. Результатом є документ з найбільшою близькістю до пошукового запиту. Для оцінки близькості текстів використовується міра близькості TF-IDF. Схема роботи наведена на рисунку 2.2 (див. с. 21).

Незважаючи на те, що даний патент акцентує увагу саме на методі пошуку, в ньому не висвітлено один нюанс – даний метод не включає в себе певної структури пошукового запиту (тобто є насамперед семантичним неструктурованим пошуковим методом). Окрім цього, алгоритм отримання числового значення, що базується на словнику, потребує значних витрат для його збереження в ОЗП чи ПЗП.

В роботі [13] розглядається найбільш досконалий з усіх варіантів метод – метод семантичного структурованого пошуку, заснований на перетворенні неструктурованого пошукового запиту в структурований.

Згідно з патентним описом, спочатку з неструктурованого запиту проводиться спроба виділити елементи пошукової структури. В разі успіху проводиться структурований пошуковий запит і видаються результати.

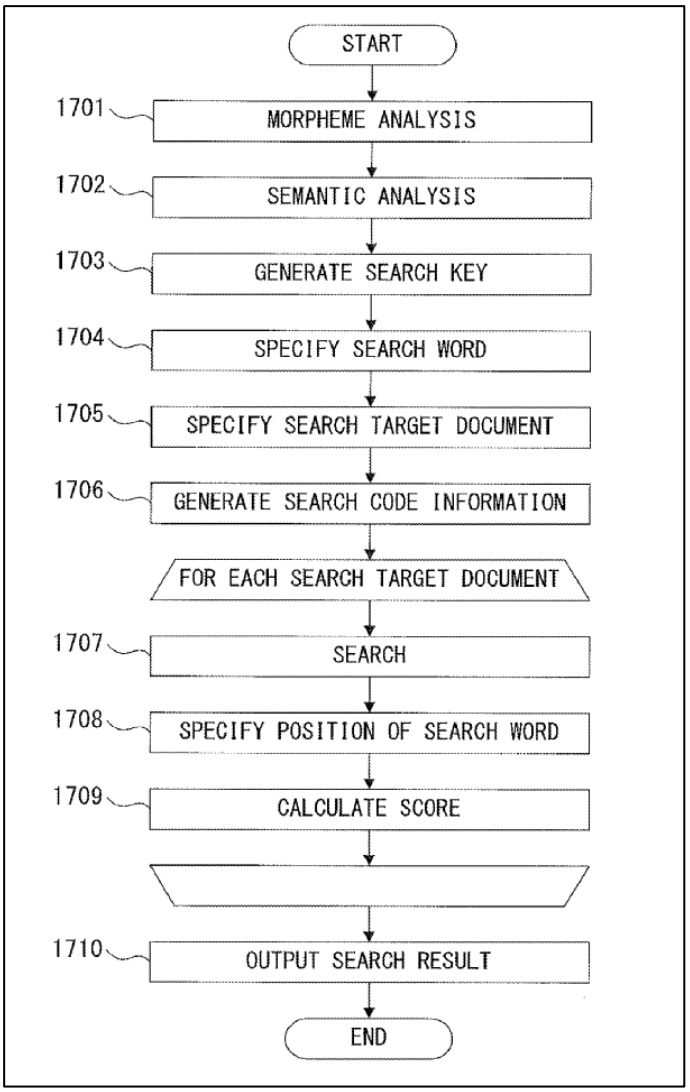

Рисунок 2.2. – схема роботи винаходу на базі семантичного структурованого пошуку

Цей метод є досить досконалим з точки зору зручності використання, проте має недолік з точки зору вхідних даних – патент не описує випадок, коли пошуковий запит не містить елементів пошукової структури (або система не визначила такі лексеми). Принцип його роботи наведений на рисунку 2.3 (див. с. 22).

Таким чином, кожен з проаналізованих патентів містить як свої переваги, так і свої недоліки. В той час, як перший проаналізований патент більше стосувався способу зберігання документів та пошукової інформації, останні два були спрямовані саме на методу структурованого пошуку такої інформації. В той же час, жоден з патентів, на жаль, не описував схему вилучення пошукової інформації з документу під час його завантаження до сховища. Окрім цього, не було приділено значної уваги можливій оптимізації організації сховищ для більш ефективної організації пошукового процесу, як було зазначено в розділі 1.1.

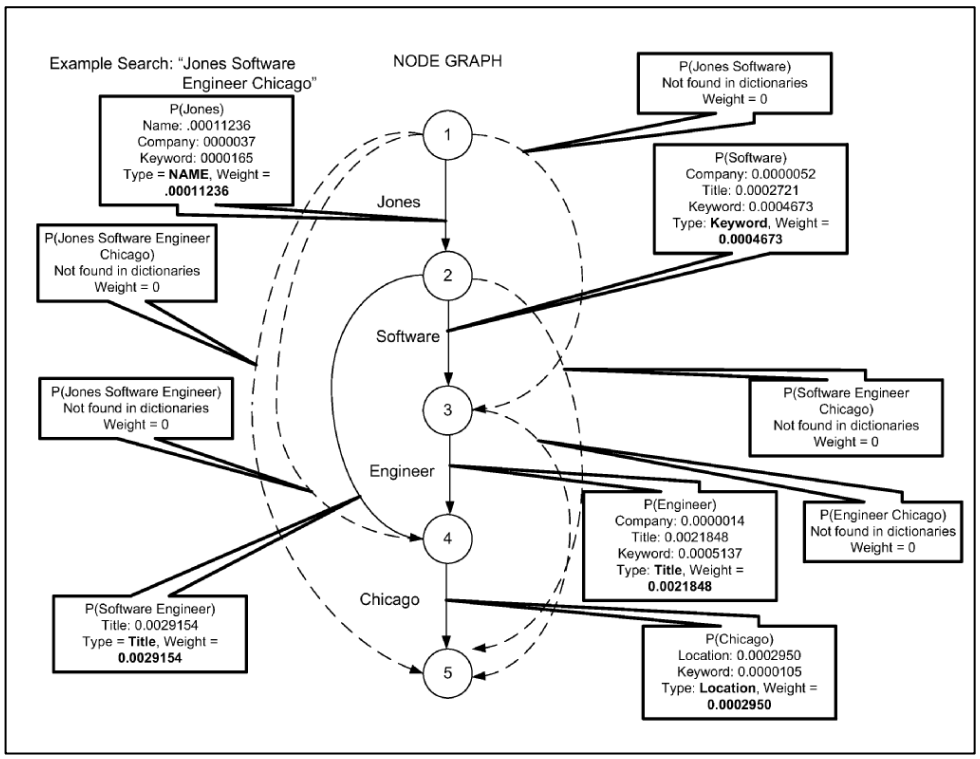

Рисунок 2.3 – схема роботи винаходу на базі відображення неструктурованої моделі

Найбільш досконалим з усіх патентів виявився останній розглянутий патент, проте і він має певні недоліки, які можуть критично впливати на інформаційний пошук.

2.2. Огляд вітчизняних наукових публікацій

В роботах вітчизняних дослідників тема дослідження була висвітлена в роботах [14] – [16]. Зокрема, в роботі [14] було проаналізовано ефективність підходу семантичного пошуку взагалі та проведено експеримент щодо виміру ефективності пошуку існуючих систем як неструктурованого, так і структурованого пошуку. Було визначено, що за інших еквівалентних обставин в більшості випадків результати структурованого пошуку є більш релевантними, аніж неструктурованого. В роботі [15] було розглянуто прикладні моделі (щоправда, для української мови) систем, що мають змогу проводити розбиття запиту по категоріях з допомогою BERT-моделі, що була навчена на корпусі текстів української мови. Щоправда, в дослідженні було визначено, що існуючі BERT-моделі (mBERT, roBERTa та ін.) мають досить невелику метрику ефективності. Частково це пов’язано з особливостями формування моделей саме для корпусу української мови.

В роботі [16] проаналізовано можливості трансформації пошукових запитів на прикладі натуральної людської мови за допомогою побудови семантичної інтелектуальної системи, що спирається на алгебру скінчених предикатів. Незважаючи на те, що такий підхід є більш універсальним та дозволяє розв’язати широкий спектр задач, така семантична модель є більш складною у прикладному використанні, оскільки використовує АСП-структури, реалізація яких відсутня у більшості програмних засобів на базовому рівні.

Слід зазначити, що в наведених публікаціях не наводилася проблема дослідження саме змішаної системи – використання семантичного структурованого пошуку для неструктурованих пошукових запитів. Це робить тему поточного дослідження актуальною.

3 РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ

Для розгляду в подальшому введемо наступні простори – нехай $D = \{D_1, D_2, \dots, D_n\}$ – простір документів, де кожний документ представляє собою текст, а $M = \{M_1, M_2, \dots, M_n\}$ – простір метаданих, причому кожен з об'єктів складається з множини властивостей $P_1, P_2, \dots, P_n$ – де кожна властивість задається конфігурацією пошукової системи. В даному дослідженні структура метаданих складається з 6 властивостей, описаних в розділі 4.

Тоді систему можна описати як n операторів, кожен з яких виконує певне математичне перетворення, що зображено на формулі 3.1.

$$S = (O_1, O_2, \dots, O_n) \tag{3.1}$$

де S – система, що описується;

$O(X, Y): X \to Y$ – оператор перетворення множини X в множину Y.

Для розглянутої схеми система структурно буде складатися з двох таких операторів. Перший оператор перетворює простір необроблених документів на вході в простір метаданих, тобто (див. формулу 3.2.):

$$O_1(D, M): D \to M \tag{3.2}$$

причому кожен об'єкт простору метаданих має структуру, зображену на формулі 3.3.

$$M = (P_1, P_2, P_3, P_4, P_5, P_6) \tag{3.3}$$

де $P_1 - P_6$ – критерії, описані в розділі 4.

Таким чином, результатом перетворення, що здійснюється оператором O_1 є колекція метаданих, описаних структурою, зображеною в формулі 3.3.

Другий оператор (або оператор пошуку) відображує пошуковий запит в простір метаданих, які задовольняють умовам пошуку. Нехай пошуковий запит має структуру, що зображена на формулі 3.4:

$$Q = \big((C_1, P_1), (C_2, P_2), (C_3, P_3), (C_4, P_4), (C_5, P_5), (C_6, P_6)\big) \tag{3.4}$$

де P_n – властивість з структури метаданих;

C_n – критеріальна умова, що накладається на дану властивість.

Простір критеріальних умов включає в себе дві умовні групи: числові умови (умова еквівалентності та порівняння) та текстові умови (умова еквівалентності та близькості). Кожна критеріальна умова також є бінарним логічним оператором – її результатом є булевий простір.

Таким чином, оператор пошуку можна представити у наступному вигляді (див. формулу 3.5):

$$O_2(Q, M): Q \to M : \exists q \in Q, m \in M: \forall p, c \in Q: p(q)\ c\ p(m) = 1 \tag{3.5}$$

де q – об’єкт пошукового запиту;

m – об’єкт метаданих;

c та p – критеріальна умова та властивість відповідного пошукового запиту.

Тобто оператор пошуку визначає таку відповідність пошукових запитів до метаданих, коли усі бінарні оператори критеріальних умов є дійсними (результат їх застосування до властивості в пошуковому запиті і властивості в метаданих – істина).

В подальшому при формуванні критеріальних умов буде використано такі необхідні оператори над простором текстів, як оператор відстані між текстами та оператор визначення наявності фрагменту в тексті.

Серед метрик відстані між текстами виділяють наступні:

– відстань Хеммінга – показує, в якій кількості позицій текст А не співпадає з текстом Б;

– відстань Джаро-Вінклера – модифікація відстані Джаро, яка показує кількість односимвольних перетворень, з допомогою яких текст А перетворюється в текст Б;

– відстань Левенштейна та її модифікація – відстань Дамерау-Левенштейна – визначає кількість односимвольних перетворень, ціна яких неоднакова, за допомогою яких текст А перетворюється в текст Б.

Також існують метрики, засновані не на символьному, а на векторному представлені рядків. Серед таких найбільш популярною є метрика TF-IDF.

Зважаючи на запроваджені параметри зберігання документів, можна виділити наступні критеріальні умови:

– умова збіжності авторів: $c_1(a,b)$: $1 - d(a,b) > 0.7$, де d – оператор відстані між рядками;

– умова збіжності дат: $c_2(a,b)$: $year(a) = year(b)$, де year – оператор отримання року від дати, тобто дати вважатимуться еквівалентними хоча б якщо їх роки співпадають;

– семантична умова збіжності: $c_3(a,b)$: $\frac{1}{3}\big(1 - d(a,b)\big) + \frac{1}{3}\big(kmp(a,b)\big) + \frac{1}{3}\big(kmp_a(a,b)\big) > 0.6$, де d – оператор відстані між рядками, kmp – оператор визначення наявності підрядка в рядку, kmp_a – модифікацію оператору kmp, що при пошуку підрядку виконує пошук не цілком, а у лексемах, враховуючи можливі перестановки слів, які семантично не змінюють значення тексту.

Таким чином, поєднання наведених трьох операторів визначає булевий результат відповідності документа пошуковому запиту. В подальшому, при виведенні результатів, відповідні документи сортуються згідно наступного правила. Нехай маємо багатокритеріальну систему, що має критерії, відповідні до наведених критеріальних умов пошуку (збіжність авторів, дат, та семантична збіжність). Тоді остаточну оцінку документа можна провести згідно з формулою 3.6.

$$Score(D,s) = \sum_{i=1}^{3} a_i(D,s) * q_i \,, 0 \le Score \le 1, \sum_{i=1}^{3} q_i = 1 \qquad (3.6)$$

де Score – фінальна оцінка релевантності документа;

$a_i(D,s)$ – числове значення збіжності за відповідною критеріальною умовою;

q_i – відносна вага критерія збіжності.

Тоді результатом роботи математичних перетворень буде множина документів, отримана згідно з формулою 3.7.

$$D_{result}(s) = D \cap D'(s) \colon \forall i,j : i < j : Score\left(D_{result,i}, s\right) > Score\left(D_{result,j}, s\right) \qquad (3.7)$$

де $D_{result}(s)$ – остаточний результат математичних перетворень;

D – вихідна колекція документів;

$D'(s)$ – результат фільтрації документів оригінальної колекції.

Результати пошукового запиту також мають відповідати умові релевантності – тобто, результат пошуку не має містити документи, які отримали досить низьку оцінку згідно до формули 3.6. Даний крок описано в формулі 3.8.

$$\forall i,s \colon Score\left(D_{result,i}(s)\right) \ge 0.5 \qquad (3.8)$$

де i – індекс документа в результуючій множині;

s – пошуковий запит;

$Score(D_{result,\,i}(s))$ – числове значення релевантності, підраховане за формулою 3.6.

Для надання інформації щодо правильності було використано показники, що використовуються для аналізу бінарних класифікаторів, а саме наступні метрики:

– True Positive – кількість документів, що коректно було видані пошуковою системою;

– False Positive – кількість документів, що некоректно були видані пошуковою системою;

– False Negative – кількість документів, які пошукова система не видала, незважаючи на їх релевантність;

– True Negative – кількість нерелевантних документів, що не було видано пошуковою системою).

Для аналізу даних використаємо такі дискримінативні показники як точність (Accuracy), влучність (Precision), повнота (Recall), їх метрику-згортку, відому як F-міра, або міра ван Рейсбергена. Опис метрик наведено в формулах 3.9 – 3.12.

$$Acc = \frac{TP + TN}{D} \tag{3.9}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.10}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.11}$$

$$F_\alpha = \frac{(1 + \alpha^2)(Precision * Recall)}{\alpha^2 Precision + Recall}, F_1 = 2Precision * \frac{Recall}{Precision + Recall} \tag{3.12}$$

де D – розмір колекції;

α – параметр F-міри, який задає пропорцію впливу двох метрик.

4 ОПИС ПРИЙНЯТИХ ПРОЕКТНИХ РІШЕНЬ

Спираючись на проаналізовані наукові публікації, а також відповідно до завдання дослідницької роботи, було вирішено розробити схему пошукової системи з урахуванням всіх переваг та недоліків розглянутих аналогів. Структурно система складається з 3 основних частин – модуль завантаження, модуль пошуку та модуль збереження. Проте для опису усіх частин детально необхідно визначити структуру метаданих пошукової системи, яка буде використана усіма її частинами.

Як найбільш оптимальні, було обрано наступні метадані:

– дата створення;

– дата завантаження;

– анотація;

– автор\автори;

– розмір документу;

– адреса документу.

Наведена структура забезпечує бібліографічний варіант документального пошуку, найближчим аналогом якого є картотека – коли пошук проводиться не серед документів, а серед карток з інформацією про місце знаходження документа. Таке рішення прийняте з оглядом на можливий великий розмір документів, що значно навантажить роботу системи, якщо документи зберігати разом з метаданими.

4.1. Огляд системи завантаження документів

Завантаження документу, що виступає у якості вхідних даних, в пошукову систему має проходити в 5 етапів.

Першим етапом є обробка документу модулем генерації реферату, що базується на технології автоматичного створення реферату з області обробки природних мов. Результатом цього етапу є невеликий реферат, який відображає суть даного документу.

На другому етапі реферат обробляється модулем, що проводить класифікацію тексту та співвідносить реферат з однією з категорій. Категорія визначає сховище, де має зберігатися документ.

На третьому етапі автоматично формуються метадані про документ. Метадані документу – це усі дані, що автоматично вилучаються з властивостей документу, окрім анотації та адреси документа. Анотацією вважається результат попереднього кроку.

На четвертому етапі документ завантажується у сховище документів. У результаті чого сховище повертає адресу зберігання документу.

На останньому етапі адреса зберігання документу додається до метаданих, після чого вони вважаються повністю сформованими та зберігаються в основному пошуковому сховищі.

Діаграму послідовностей, що відображає наведений алгоритм завантаження документу, наведено на рисунку 4.1.

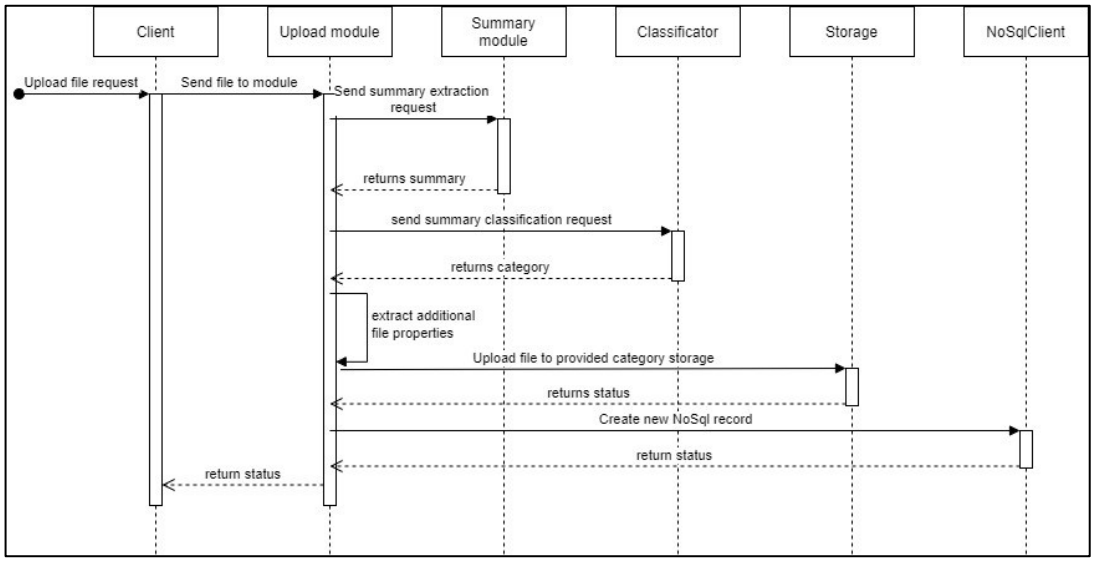

Рисунок 4.1 – діаграма послідовностей для процесу завантаження документа

Результатом виконання алгоритму є лише позитивна відповідь системи, що відповідний файл оброблено та створено відповідний запис в СКБД.

4.2. Огляд системи структурованого пошуку документів

У якості пошукового запиту для системи було обрано неструктурований запит. Метод пошуку передбачає трансформацію неструктурованого запиту в структурований, проте, на відміну від патенту 3, трансформація буде відбуватися за допомогою NER-системи, яка буде виділяти окремі пошукові категорії. В якості анотації буде виступати увесь пошуковий запит на той випадок, якщо розпізнання сутностей не дало бажаного ефекту.

Далі буде проведено пошук зі збігом значень метаданих (або відповідності наведеним пошуковим критеріям) та визначено близькість анотацій метаданих сховища до анотації пошукового запиту. Для потрапляння до списку результатів достатньо хоча б одного позитивного збігу. Після висвітлення результатів користувач має можливість перейти за вказаною адресою, щоб передивитися увесь документ, який він шукав.

Таким чином, схема пошуку по запиту буде мати наступний вигляд:

– вилучення з пошукового запиту іменованих сутностей у разі їх наявності. Слід зазначити, що перелік іменованих сутностей, що вилучаються, може бути змінним;

– фільтрація документів за вилученими атрибутами метаданих;

– оцінка кожного з фільтрованих документів;

– сортування за оцінкою та повернення результату користувачу.

4.3. Огляд системи зберігання документів

Як було зазначено в пунктах 4.1 та 4.2 система зберігання складається структурно з двох сховищ – сховища метаданих та сховища документів. Сховище документів представляє собою на вибір – або фізичний носій, або веб-каталог для зберігання документів. Сховище метаданих являє собою нереляційну СКБД, яка зберігає усі наведені структурні параметри, серед яких проводиться пошук. Таким чином, досягається дві основні мети:

– потенційно великий обсяг документів не заважає ефективному пошуковому процесу;

– нереляційна СКБД для зберігання метаданих не потребує значних ресурсів для зберігання та пошуку серед метаданих, що позитивно впливає на продуктивність системи в цілому.

Система зберігання документів розподіляється на декілька директорій (відповідно до категорій, які може видати модуль, що проводить класифікацію реферату). Відповідно, окрім імені файлу в результаті пошуку користувач повинен також знати і відповідну адресу директорії, де цей файл зберігається. На рисунку 4.2 наведено діаграму послідовностей для процесу пошуку та завантаження документу.

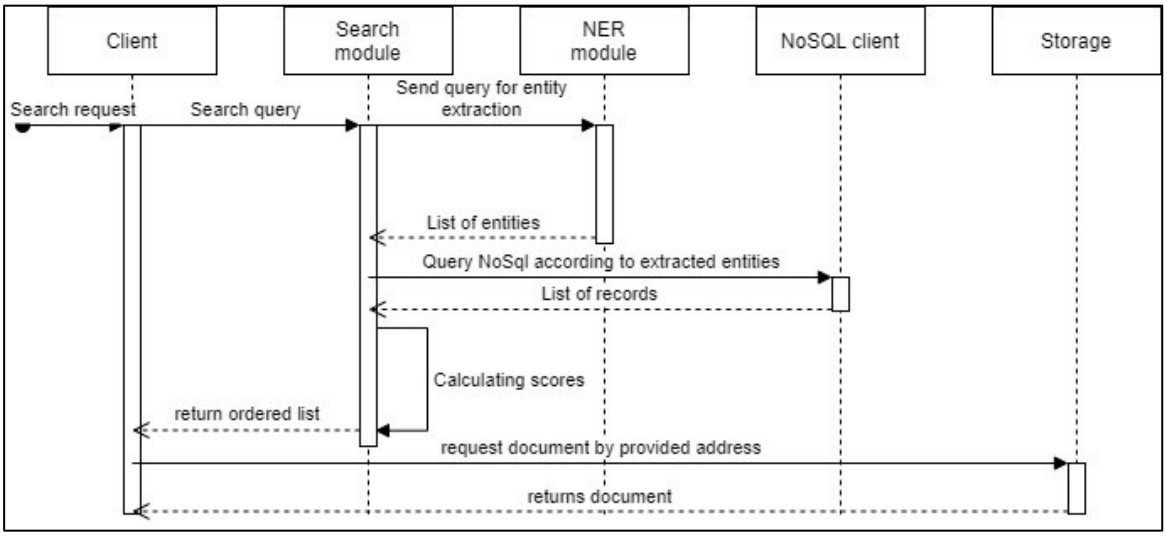

Рисунок 4.2 – діаграма послідовностей для процесу пошуку та завантаження документа

Тобто пошук та завантаження документу відбувається наступним чином:

– клієнт надсилає пошуковий запит;

– модуль пошуку документів намагається розпізнати іменовані сутності у наданому неструктурованому запиті;

– після того, як були виявлені іменовані сутності у неструктурованому запиті, відбувається запит до нереляційної бази даних для пошуку даних за ідентифікованими сутностями у запиті;

– після отримання результатів з бази даних про наявні файли за запитом оцінюється релевантність кожного окремого знайденого документу за рахунок виміру близькості реферату тексту до пошукового запиту;

– знайдені документи сортуються у порядку зменшення оцінки, отриманої на попередньому етапі;

– клієнт отримує список адрес документів, що є релевантними для його запиту.

Таким чином, запропонована схема завантаження, зберігання та пошуку документів покриває повний цикл обертання документу та відповідає усім користувацьким вимогам, які найбільш часто висуваються користувачами до систем електронного зберігання та пошуку інформації.

5 ОПИС ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

Для оцінки адекватності та точності роботи запропонованої системи було побудовано наступний план проведення експериментів.

В експерименті 1 робота пошукової системи перевірялася при роботі з документами приблизно схожого змісту (класифікатор співвідносив такі реферати до однієї і тої самої категорії). Самі документи – невеликі (до 10 сторінок) наукові статті, присвячені різним темам, споріднені лише напрямком дослідження. Такий вибір вхідних даних до експерименту надає можливість перевірити, наскільки точно пошукова система виокремлює деталі пошукового запиту. Експеримент було проведено на 25 документах (тобто загальний обсяг текстової інформації не перевищує 250 сторінок) з використанням 4 різних пошукових запитів, створених довільним чином. В таблиці 5.1 наведено результати, що очікувалися від системи (якщо б пошук проводився людиною), а також результати роботи пошукової системи. В таблиці 5.2. наведено метрики ефективності та їх середнє значення щодо експерименту №1.

Таблиця 5.1 – актуальні та очікувані дані щодо експерименту №1

	True Positive	False Positive	False Negative	True Negative
1	12	10	2	1
2	10	5	8	2
3	10	3	7	5
4	9	6	6	4

Таблиця 5.2 – оцінки ефективності моделі в ході експерименту №1

	Accuracy	Precision	Recall	F1
1	0,52	0,545455	0,857143	0,666667
2	0,48	0,666667	0,555556	0,606061
3	0,6	0,769231	0,588235	0,666667
4	0,52	0,6	0,6	0,6

Кінець таблиці 5.2

	Accuracy	Precision	Recall	F1
Середнє	0,53	0,645338	0,650233	0,634848

Таким чином, можемо зробити висновок, що в рамках поставленого експерименту створена модель продемонструвала середню якість пошуку (невисокі показники точності та F-міри).

В експерименті 2 робота пошукової системи перевірялася при роботі з документами різного змісту з різних предметних областей, причому розподіл було взято приблизно однаковий по категоріях (було визначено документи 5 категорій: математика та наука, медицина, економіка, соціальні науки та будівництво).

Результати роботи пошукової системи для експерименту 2 приведено в таблиці 5.3.

Таблиця 5.3 – актуальні та очікувані дані щодо експерименту №2

	True Positive	False Positive	False Negative	True Negative
1	3	2	1	19
2	4	1	2	18
3	2	0	1	22
4	2	2	1	20

Обчислимо наведені вище метрики для результатів експерименту №2. Дані метрики наведено в таблиці 5.4.

Таблиця 5.4 – оцінки ефективності моделі в ході експерименту №2

	Accuracy	Precision	Recall	F1
1	0,88	0,6	0,75	0,666667
2	0,88	0,8	0,666667	0,727273
3	0,96	1	0,666667	0,8

Кінець таблиці 5.4

	Accuracy	Precision	Recall	F1
4	0,88	0,5	0,666667	0,571429
Середнє	0,9	0,725	0,6875	0,691342

Результати експерименту №2 показують, що запропонована модель значно краще працює в умовах змішаної колекції документів (що більш наближено до реального сценарію користувача), причому незважаючи на незначний ріст F-міри (в середньому лише в 1,09 разів), показник точності зріс в середньому в 1,7 разів.

В експериментах №3 та №4 хід експерименту, набір даних, що збиралися, а також набір метрик, за якими проводилася оцінка, залишався ідентичним, проте на відміну від експериментів №1 та №2 була використана модель неструктурованого пошуку (без етапу виокремлення іменованих сутностей). Задача експерименту полягала у визначенні впливу структурованості пошуку на результат роботи моделі.

Хід експерименту 3 – провести 4 пошукові запити на колекції з 25 документів спорідненого змісту з використанням моделі неструктурованого пошуку.

Хід експерименту 4 – провести 4 пошукові запити на колекції з 25 документів різноманітного змісту з використанням моделі неструктурованого пошуку.

Дані експериментів №3 та №4 наведено в таблицях 5.5 – 5.8.

Таблиця 5.5 – актуальні та очікувані дані щодо експерименту №3

	True Positive	False Positive	False Negative	True Negative
1	8	10	6	1
2	7	8	5	5
3	8	8	6	3
4	7	10	2	6

Таблиця 5.6 – оцінки ефективності моделі в ході експерименту №3

	Accuracy	Precision	Recall	F1
1	0,36	0,444444	0,571429	0,5
2	0,48	0,466667	0,583333	0,518519
3	0,44	0,5	0,571429	0,533333
4	0,52	0,411765	0,777778	0,538462
Середнє	0,45	0,455719	0,625992	0,522578

Таблиця 5.7. – актуальні та очікувані дані щодо експерименту №4

	True Positive	False Positive	False Negative	True Negative
1	9	9	3	4
2	9	6	3	7
3	10	6	4	5
4	12	6	1	6

Таблиця 5.8 – оцінки ефективності моделі в ході експерименту №4

	Accuracy	Precision	Recall	F1
1	0,52	0,5	0,75	0,6
2	0,64	0,6	0,75	0,666667
3	0,6	0,625	0,714286	0,666667
4	0,72	0,666667	0,923077	0,774194
Середнє	0,62	0,597917	0,784341	0,676882

В результаті проведення експериментів 3 та 4 визначено відповідну тенденцію, що й в експериментах 1 та 2 (показники точності та F-міра вище у випадку випадково розподілених за змістом документів), причому в поточному випадку метрика точності зросла в середньому в 1,38 разів, а F-міра в 1,295 разів. Проте, в обидвох експериментах числові значення метрик є меншими, аніж у відповідних експериментах 1 та 2.

6 АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Відповідно до проведених експериментів та отриманих показників результативності моделей, можна зробити наступні висновки:

По-перше, характер пошукової колекції має суттєвий вплив на результативність роботи пошукової системи. В ході експериментів 1 та 3 було визначено, що порівняно з відносно спорідненими документами пошукова система видає більш релевантні результати у випадку, коли документи не є спорідненими за змістом і співвідносяться модулем класифікації рефератів до різних класів.

Дана тенденція пов’язана з характером отримання оцінки семантичної збіжності – оскільки документи самі по собі є більш близькими у разі використання метрики TF-IDF, то і пошуковий запит порівняно з документами буде більш близьким, що збільшує кількість нерелевантних документів в результатах запиту (False Positive). У випадку, коли колекція документів має в середньому більшу попарну відстань один від одного, пошукова модель видає менше нерелевантних документів.

По-друге, в ході експериментів було визначено, що структурована модель пошуку є більш оптимальною для пошуку як у випадку споріднених, так і неспоріднених документів в колекції. Так, показники точності для експериментів 3 та 4 гірші в 1,18 та 1,45 разів, а показники F-міри гірші в 1,2 та 1,02 рази порівняно з відповідними метриками для експериментів 1 та 2.

Така тенденція показує, що додавання структурованих запитів до колекції має рацію, оскільки таким чином зменшується кількість потенційно нерелевантних документів, які можуть потрапити до результатів пошуку після підрахування оцінки семантичної збіжності за рахунок фільтрації колекції документів на предмет атрибутів, що розпізнаються модулем категоризації іменованих сутностей.

По-третє, обидві моделі (як структурованого, так і неструктурованого пошуку) мають відносно низьке значення метрики F-міри. В усіх експериментах середнє значення не перевищує відмітку 0,7. Така негативна тенденція пов’язана з відносно високою кількістю нерелевантних документів та таких, що не були

визначені, як релевантні. Це потребує внесення відповідних коректив до алгоритму підрахунку метрики семантичної збіжності (наприклад, заміна алгоритму підрахунку відстані між текстами, коригування відносних критеріальних ваг та ін.).

Незважаючи на це, структурована пошукова система у випадку неспоріднених документів в колекції має високий показник точності, що частково компенсує не такий високий показник F-міри. Також, для покращення значення F-міри можливе розгалуження формату зберігання метаданих щодо документу та збільшення кількості атрибутів, що можна розпізнати як іменовану сутність. Даний крок позитивно вплине на процес фільтрації нерелевантних документів в разі коректного вилучення та фільтрування метаданих.

7 ОПИС ТЕХНОЛОГІЇ ТА ПРОГРАМНОГО ДОДАТКУ

7.1 Опис технологій

В ході проектування та реалізації тестового програмного додатку, що реалізовує описану пошукову систему, було вирішено використати рішення компанії Amazon, а саме Amazon Web Services. Вибір даної технології пов’язаний з наявністю більшості функціоналу, необхідного в ході дослідження. Так, AWS надає автоматично можливість використання розподіленого сховища даних, безсерверних обчислень та інші позитивні особливості, що було використано в ході дослідження.

Amazon Web Services (в подальшому AWS) надає потужний інструментарій для розміщення неважкого програмного коду на готовій інфраструктурі [17]. Зокрема, AWS надає інструментарій в наступних площинах:

– створення та розміщення RESTful серверів;

– створення віртуальних машин під будь-яку цільову платформу;

– створення обчислювальних модулів за технологією Serverless;

– створення та обслуговування NoSql баз даних;

– зберігання будь-яких файлів та розміщення статичних веб-сайтів та ін.

Основна мета подібного інструментарію – відносно просте розміщення програмних модулів в хмарі. На рисунку 7.1 (див с. 41) наведено приклад взаємодії модулів, створених з допомогою інструментарію AWS.

Технологія AWS була обрана відповідно до наступних програмних вимог:

– максимальна простота демонстраційного програмного коду;

– наявність усіх необхідних ресурсів для пошукової системи;

– швидкість розміщення коду в хмарі;

– наявність демонстраційної безкоштовної версії.

Для автоматизованого контролю за AWS ресурсами, їх створенням, модифікацією та видаленням було використано фреймворк Terraform.

Інфраструктура як інструменти коду (IaaS) дозволяє керувати інфраструктурою з файлами конфігурації, а не через графічний інтерфейс

користувача. IaaS дозволяє будувати, змінювати та керувати інфраструктурою безпечним, послідовним та повторюваним, визначаючи конфігурації ресурсів, які ви можете версію, повторно використовувати та ділитися.

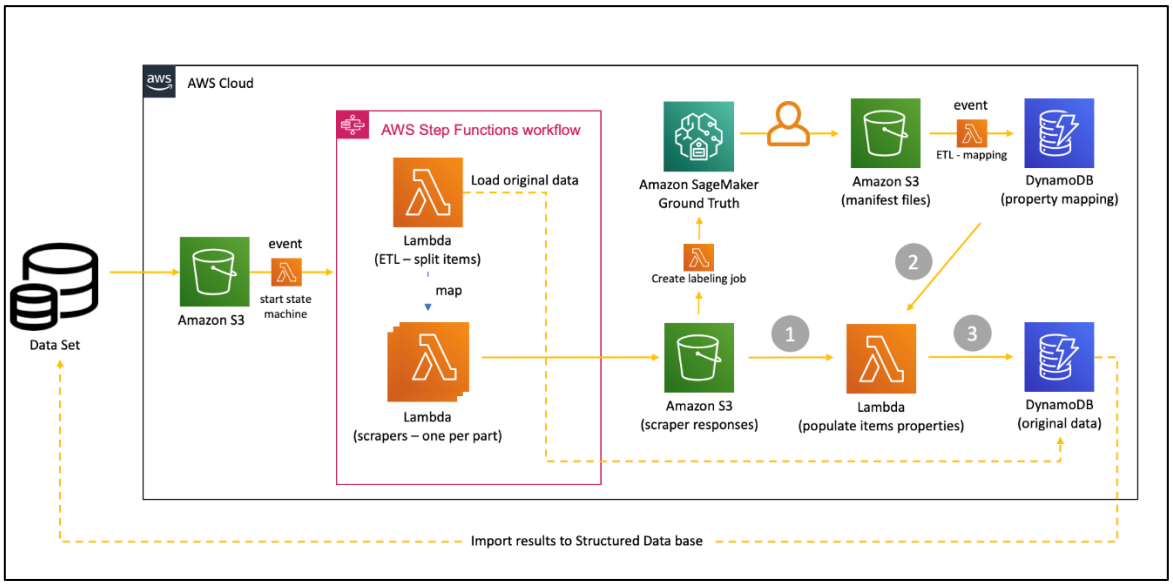

Рисунок 7.1 – схема розміщення програмних модулів з допомогою AWS

Terraform – це інфраструктура Hashicorp в якості інструменту коду. Це дозволяє визначити ресурси та інфраструктуру у файлах, що читаються, декларативних файлах конфігурації та керувати життєвим циклом інфраструктури [18]. Використання Terraform має кілька переваг перед управлінням інфраструктурою вручну:

– Terraform може керувати інфраструктурою на декількох хмарних платформах;

– мова конфігурації, що читається людиною, допомагає швидко писати код інфраструктури;

– Terraform дозволяє відстежувати зміни ресурсів протягом ваших розгортань.

Плагіни Terraform під назвою провайдери дозволяють Terraform взаємодіяти з хмарними платформами та іншими послугами через інтерфейси програмування додатків (API). Hashicorp та спільнота Terraform написали понад 1000 провайдерів

для управління ресурсами на веб-службах Amazon (AWS), Azure, Google Cloud Platform (GCP), Kubernetes, Helm, Github, Splunk, DataDog та ін.

Провайдери визначають окремі одиниці інфраструктури, наприклад, обчислювальні екземпляри або приватні мережі як ресурси. Приклад оформлення terraform наведено на рисунку 7.2.

```
resource "aws_api_gateway_rest_api" "api" {
  name = "${var.bucket_name_prefix}-api"
}

resource "aws_api_gateway_account" "api_log_account" {
  cloudwatch_role_arn = aws_iam_role.aws_apigateway_log_role.arn
}

resource "aws_api_gateway_resource" "regions" {
  rest_api_id = aws_api_gateway_rest_api.api.id
  parent_id   = aws_api_gateway_rest_api.api.root_resource_id
  path_part   = "regions"
}

resource "aws_api_gateway_resource" "region" {
  rest_api_id = aws_api_gateway_rest_api.api.id
  parent_id   = aws_api_gateway_rest_api.api.root_resource_id
  path_part   = "region"
}

resource "aws_api_gateway_resource" "summary" {
  rest_api_id = aws_api_gateway_rest_api.api.id
  parent_id   = aws_api_gateway_rest_api.api.root_resource_id
  path_part   = "summary"
}

resource "aws_api_gateway_resource" "category" {
  rest_api_id = aws_api_gateway_rest_api.api.id
  parent_id   = aws_api_gateway_rest_api.api.root_resource_id
  path_part   = "category"
}

resource "aws_api_gateway_resource" "upload" {
  rest_api_id = aws_api_gateway_rest_api.api.id
  parent_id   = aws_api_gateway_rest_api.api.root_resource_id
  path_part   = "upload"
}
```

Рисунок 7.2 – приклад коду на Terraform

Мова конфігурації Terraform є декларативною, це означає, що вона описує бажаний кінцевий стан для інфраструктури, на відміну від процедурних мов програмування, які потребують покрокових інструкцій для виконання завдань. Провайдери Terraform автоматично обчислюють залежності між ресурсами для створення або знищення їх у правильному порядку.

7.2 Загальна структура системи

Описані в розділі 4 модулі пошукової системи було створено з допомогою ресурсів AWS наступним чином:

– модулі класифікації, реферування, завантаження, підрахунку семантичної близькості та пошуку було виділено як окремі ресурси типу AWS Lambda, які реалізовано з допомогою мови програмування Python 3.9 лише з тими залежностями, що необхідні для роботи алгоритмів обробки натуральних мов. AWS Lambda – це служба розрахунку, яка запускає програму у відповідь на певні події, і відповідає за автоматичне розподіл необхідних обчислювальних ресурсів. Список подій включає зміни в стані або оновлення, наприклад, коли користувач розміщує товари в кошик на веб-сайті Інтернет-комерції. AWS Lambda може використовуватися для розширення можливостей інших служб AWS за допомогою спеціальної логіки або для створення власних серверів, використовуючи можливості масштабування, продуктивності та безпеки AWS. AWS Lambda автоматично запускає програмний код у відповідь на різні події, такі як HTTP-чеки через шлюз Amazon API, зміна об'єктів в Amazon Simple Service Service (Amazon S3), оновлення таблиць в Amazon DynamoDB або зміна умов у функції AWS. Lambda запускає код у високоефективному обчислювальному середовищі та повністю виконує адміністрування обчислювальних ресурсів. Включаючи послугу сервера та операційну систему, розподіл ресурсів та автоматичне масштабування, розгортання виправлень коду та системи безпеки, а також моніторинг кодів та журналів.

– модуль зберігання документів було реалізовано з використанням ресурсів AWS S3 Bucket. Даний ресурс дозволяє створювати різні віртуальні контейнери лише з однією метою – збереження файлів. Amazon Simple Service (Amazon S3) – це послуга для зберігання об'єктів, яка пропонує найкращі показники продуктивності в галузях, масштабованість, доступність та безпеку даних. Клієнти будь-якого розміру та будь-якої промислової галузі можуть зберігати та захищати необхідну кількість даних майже для будь-якого прикладу використання.

Наприклад, для озер даних, хмарних додатків та мобільних додатків. Сприятливі класи зберігання та прості засоби для використання інструментів адміністрування дозволяють оптимізувати витрати, організовувати дані та точно налаштувати обмеження доступу відповідно до потреб бізнесу чи законодавчих вимог.

– збереження метаданих про завантажені документи виконується з допомогою Amazon DynamoDb. Даний ресурс являє собою NoSql СКБД типу key-value. Amazon DynamoDB – це повністю керована бездоганна база даних NOSQL на основі пари ключових значень, створена для запуску високопродуктивних програм у будь-якій шкалі. DynamoDB пропонує вбудований захист, безперервну підтримку, автоматичну реплікацію в декількох регіонах, кешування пам'яті та інструменти експорту даних.

Координацію усіх наведених вище модулів було організовано з використанням ресурсу AWS API Gateway. Даний ресурс надає можливість створення незалежного RESTful API, сполучаючи відповідні кінцеві точки з Serverless модулями, створеними за допомогою технології AWS Lambda. Amazon API Gateway – це повністю контрольована послуга для розробників, розроблена для створення, публікації, технічного обслуговування, моніторингу та забезпечення безпеки API в будь-якому масштабі. Через API програми отримують доступ до даних, логіки бізнесу або функціональність ваших серверів. API шлюзу дозволяє створити API RESTFUL та WebSocket, які є основним компонентом подвійних програм у режимі реального часу. API Gateway підтримує навантаження в контейнерах та без підтримки робочих навантажень, а також онлайн-додатки.

API шлюзу передбачає всі завдання, пов'язані з прийом та обробкою сотень тисяч одночасних дзвінків, включаючи управління дорожнім рухом, підтримку CORS, авторизацію та контроль доступу, регулювання кількості запитів, моніторингу та управління версіями API.

Для демонстраційного програмного додатку було використано готові моделі машинного навчання для автоматичного реферування, класифікації текстів та вилучення іменованих сутностей. Їх робота в програмному коді забезпечується з допомогою бібліотеки для Python Spacy3.

RESTful сервер буд створений за допомогою API Gateway. Структура організації додатку зображена на рисунку 7.3.

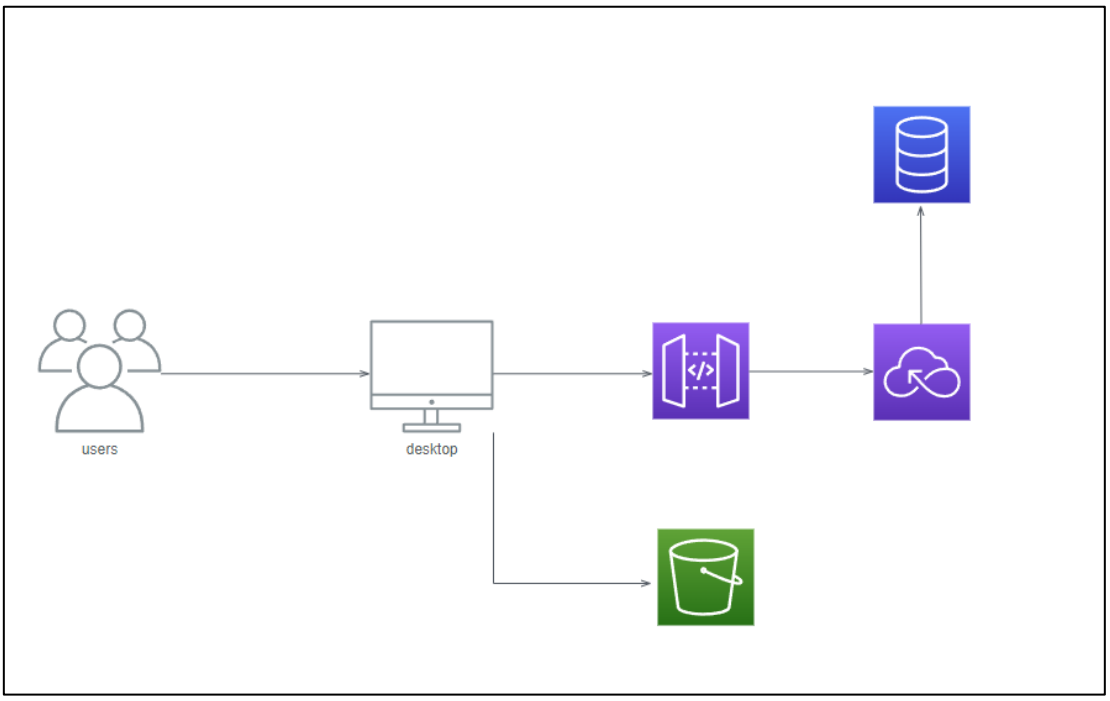

Рисунок 7.3 – структура AWS ресурсів

Усі описані ресурси було створено за допомогою terraform-скриптів. Усі скрипти було створено відповідно до типів ресурсів, що вони створюють. Також скрипти було розподілено на модулі, задля більшої гнучкості у конфігурації ресурсів.

Структурно, terraform-додаток має наступні складові:

– конфігурація API Gateway: створення API, визначення кінцевих точок, визначення HTTP методів та їх конфігурування, поєднання кінцевих точок з кодом з ресурсів Lambda та розгортання;

– конфігурація загальних S3 корзин – створення з публічними правами доступу до них та до усіх файлів, що вони містять;

– конфігурація ресурсів Lambda через модулі. Кожен модуль містить декларацію ресурсу Lambda, а також додаткові декларації (наприклад, визначення

ролей та налаштувань безпечності, визначення таблиць в DynamoDB та їх структури, визначення файлів з вихідним кодом, які мають бути заархівовані та ін.)

7.3 Реалізація модулю завантаження документів

Схема роботи модуля завантаження документів зображена у розділі 4 (див. рис. 4.1, с. 30).

Модуль завантаження документів складається з 3 підмодулів: category, summary та storage. Спочатку користувач, використовуючи клієнт, обирає файл (або купу файлів), які він бажає завантажити до системи. Далі, використовуючи заздалегідь визначену S3 директорію (де зберігаються тимчасово файли), клієнт завантажує туди файл та посилає запит до модулю upload. Структура http запиту містить лише один параметр – ім’я файлу, який було завантажено.

Модуль upload приймає запит, перевіряє наявність поточного файлу в S3 корзині з тимчасовими файлами. Далі модуль визначає тип файлу (наразі підтримуються лише документи формату docx та pdf) і відповідно до розширення обирає файловий процесор. Частину коду модуля upload наведено на рисунку 7.4 (див. с. 47).

Файловий процесор відповідає за вилучення тексту з документу, а також отримання його метаданих, таких як автор, дата створення, розмір та ін. Для вилучення тексту з PDF-файлів використано бібліотеку PyPDF2, а для вилучення з Word файлів – python2docx.

Далі, після отримання тексту та метаданих про файл, модуль upload надсилає запит до підмодуля summary, мета якого – створити короткий реферат до наведеного тексту. За замовчування реферат складається з 10 речень, що в середньому еквівалентно 200-250 словам. Отриманий реферат повертається як результат до модуля upload.

Наступним кроком модуль upload надсилає отриманий реферат до підмодуля category, який відповідає за семантичну класифікацію тексту. Підмодуль категоризує надісланий реферат за допомогою заздалегідь натренованої моделі

класифікатора, що співвідносить текст до однієї з 19 категорій відповідно до семантичного змісту. Результат – обрана категорія – повертається до модуля upload.

```
class PdfTextExtractor(BaseTextExtractor):
    def parse_text(self, file_path: str) -> str:
        with open(f"{self.temp_folder}/{file_path}", "rb") as f:
            reader = PyPDF2.PdfFileReader(f)
            result = ""
            for i in range(reader.numPages):
                result += reader.getPage(i).extractText()
        return result

    def get_metadata(self, file_path: str) -> dict:
        doc_size = str(os.path.getsize(f'{self.temp_folder}/{file_path}'))
        with open(f"{self.temp_folder}/{file_path}", "rb") as f:
            reader = PyPDF2.PdfFileReader(f)
            info = reader.getDocumentInfo()
            return {
                "Author": info.author,
                "Creation-Date": self.parse_datetime(info['/CreationDate']),
                "Upload-Date": datetime.utcnow().strftime("%Y-%m-%dT%H:%M:%SZ"),
                "Size": doc_size
            }

    def parse_datetime(self, datetime: str) -> str:
        try:
            return parse(datetime).strftime("%Y-%m-%dT%H:%M:%SZ")
        except ParserError:
            return parse(datetime[2:16]).strftime("%Y-%m-%dT%H:%M:%SZ")
```

Рисунок 7.4 – приклад коду модуля upload

Після цього модуль перевіряє наявність корзини, що відповідає обчисленій категорії реферату. В разі відсутності такої – надсилається запит на її створення, після чого в корзину з ім’ям, що відповідає категорії, переміщується вхідний файл.

Останнім кроком роботи є створення відповідного допису в таблиці Dynamo. Для цього модуль upload додає до метаданих, вилучених файловим процесором, згенеровану анотацію, а також категорію, як місце знаходження файлу у сховищі S3. Після цього клієнту повертається сповіщення, що файл успішно оброблено та завантажено у сховище.

7.4 Реалізація модулю зберігання документів

Схема роботи модулів зберігання та пошуку документів описана у розділі 4 (див рис. 4.2, с. 32).

Розподілене сховище створюється засобами AWS. Для цього визначено 19 категорій, до яких потенційно можна віднести будь який документ. Відповідно до цих категорій, модуль upload час від часу створює корзину, яка відповідає імені категорії та містить лише файли, чий реферат було співвіднесено до такої категорії. Це додає системі гнучкості, оскільки засобами S3 така структура корзин може бути розміщена в різних куточках планети Земля (відповідно до тих регіонів, що надає AWS) і мінімізувати час завантаження файлу, зважаючи на певні користувацьки тенденції в запитах.

Кожному файлу в корзині S3 відповідає відповідний допис в таблиці Dynamo – з посиланням на S3 корзину, де зберігається файл. Така схема збереження даних мінімізує активність самих документів, а також мінімізує програмну активність, оскільки в подальшому модулям не потрібно працювати з самими документами – їм достатньо зібраних метаданих.

7.5 Реалізація модулю пошуку документів

Модуль пошуку складається з 2 підмодулів: similarity та query. Спочатку користувач надсилає неструктурований пошуковий запит до модуля через HTTP запит. Далі, відповідно до конфігурації, модуль пошуку може або використати сирий запит (лише в цілях експерименту), або ж надсилає отриманий пошуковий запит до модулю query.

Основна задача підмодуля query – засобами NER вилучити з неструктурованого запиту певні іменовані сутності. Зважаючи на структуру метаданих, яка збирається модулем upload при завантаженні файлів, було вирішено вилучати лише сутності з тегами PERSON та DATE, які в подальшому будуть відповідати автору документу та даті його написання. Підмодуль повертає запит без вилучених сутностей, а також набір розпізнаних параметрів.

Далі модуль пошуку виконує фільтрацію дописів в таблиці Dynamo згідно з вилученими сутностями і проводить поштучний підрахунок семантичної міри збіжності. Для цього він надсилає до підмодуля similarity запит і ідентифікатор файлу. Підмодуль similarity в свою чергу за допомогою transformer-моделі рахує

відстань між анотацією та запитом, також між усіма лексемами та перевіряє входження слів пошукового запиту в анотацію і повертає число від 0 до 1 – семантичну міру збіжності. Найбільша міра збіжності буде отримана при повному збіганні анотації з пошуковим запитом.

Останнім кроком модуля пошуку є відсіювання нерелевантних даних – для досліджень було встановлено поріг 0.5, тобто усі документи, що отримали семантичну міру збіжності менше 0.5 не потрапляють у результати пошуку. Ті дописи, що залишилися, надаються користувачеві у відповідь, після чого він має змогу ознайомитися з документом відповідно до даних про ім’я файлу та його корзину S3.

8 ПОДАЛЬШИЙ РОЗВИТОК ТА ВПРОВАДЖЕННЯ ДОСЛІДЖЕНЬ

Система, що буда запропонована у рамках даного дослідження, має потенціал для розвитку як у науковій, так і в практичній площині.

Зокрема, проблематика поточного дослідження була описана у статті «Дослідження методів структурованого пошуку інформації в розподілених сховищах даних» [19], у якій було описано проблематику структурованого та неструктурованого пошуку даних.

Запропонована розроблена система у подальшому може використовуватися у наступних системах:

– системи обліку та зберігання архівних документів;

– електронні бібліотеки;

– системи для пошуку інформації в інтернеті;

– для систем в області документообігу та юриспруденції;

– для заміни старих каталогів.

Запропонований API для взаємодії спрощує зміну та впровадження рішення для пошуку та зберігання документів у системи клієнтів.

При цьому враховуючи результати експериментальних досліджень, можна зробити висновок, що існує потенціал для наукового розвитку даного дослідження. Було виявлено, що аспектами для подальшого дослідження з метою покращення запропонованої системи є наступні:

– дослідження методів та засобів для вимірювання близькості рефератів текстів за ключовими словами;

– дослідження можливості розширення та кастомізації атрибутів, за якими має проводитися пошук текстів.

ВИСНОВКИ

У ході проведення дослідження щодо структурованих методів пошуку інформації в розподілених сховищах даних було вивчено існуючі методи та підходи до побудови систем як структурованого, так і не структурованого пошуку. Було визначено основні переваги та недоліки підходів.

В ході дослідження було проаналізовано закордонні патентні публікації, а також вітчизняну літературу з приводу теми дослідження. Було проведено короткий опис кожної з публікацій з висвітленням як переваг, так і недоліків. Знайдені недоліки було враховано при описі, проектуванні, та моделюванні системи.

У рамках дослідження було проведено проектну роботу, результатом якої стала загальна схема пошукової системи, заснованої на принципах як семантичного, так і структурованого пошуку. Основні етапи пошуку було описано за допомогою математичних моделей, виокремлено основні формули та показники, необхідні для пошукового модуля загалом.

Для проведення експерименту було створено невеликий програмний додаток з використанням сервісів Amazon Web Services та мови програмування Python з урахуванням створеної в ході дослідження проектної схеми.

В ході експериментальної частини дослідження було сплановано та проведено 4 послідовні експерименти, основна мета яких – виявити поточні результати спроектованої схеми, порівняти поведінку пошукової системи в умовах різної наповненості колекції документів, а також підтвердити чи спростувати гіпотезу щодо позитивного впливу структуризації пошуку в рамках наведеної проектної схеми. Було отримано результати, що по-перше підтвердили гіпотезу щодо позитивного впливу використання структур в пошуковій системі на точність системи, по-друге було визначено, що в умовах, наближених до реальних, система працює значно краще, аніж в умовах споріднених за змістом документів. Для проведення аналізу роботи моделі було використано математичні метрики, притаманні бінарним класифікаторам. Для оцінки моделі було використано

метрики точності та F-міри, які в найкращому випадку склали 0,9 та 0,691 відповідно.

Спираючись на отримані теоретичні та експериментальні дані було виявлено потенційні причини відносно невисокого значення F-міри та надано відповідні рекомендації щодо того, які чинники можуть позитивно впливати на роботу спроектованої схеми. Окрім цього, було визначено подальші прикладні сфери застосування подібної схеми, наприклад, спроектована схема може підходити як для покращення роботи існуючих семантичних неструктурованих систем, так і для переведення до електронного вигляду існуючих бібліографічних систем пошуку та бібліотечних каталогів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Stuart A.P. Murray. The Library: An Illustrated History. — New York: W.W.Norton, 2012.

2. Fred Lerner. The Story of Libraries: From the Invention of Writing to the Computer Age. — London: Bloomsbury, 2009.

3. Martin Hilbert, Priscila Lopez. The World's Technological Capacity to Store, Communicate, and Compute Information. *Science.* 2011. №332. URL: https://bblfish.net/tmp/2012/06/18/Science-2011-Hilbert-60-5.pdf (дата звернення 01.05.2022).

4. Manning C., Raghavan P., Schütze H. Introduction to Information Retrieval. — Cambridge University Press, 2008.

5. Л. В. Грекова. Информационный поиск в информатике и библиотековедении. *Научные ведомости*. 2013. №1. URL: https://bit.ly/3vXILph (дата звернення 01.05.2022).

6. Ланкастер Ф. У. Информационно-поисковые системы. М, 1972.

7. Никитин П. И. Автоматизированные системы обработки и поиска документальной информации. М, 1977.

8. Соколов А. В. Информационно-поисковые системы. М, 1981.

9. Yaniv Pessach. Distributed Storage: Concepts, Algorithms and Implementations. NY: Amazon, 2013.

10. Cristopher J. Date. Introduction to database systems. Boston: Addison-Wesley, 2003.

11. Structured search via key-objects: патент US20200026741 США : МПК G06F 16/9535, G06F 16/9536, G06F 16/955. № 16587935, заявл. 23.01.2020, опубл. 09.11.2021. URL: https://patentscope.wipo.int/search/en/detail.jsf?docId=US282300810&_cid=P21-L0ZX33-73011-2 (дата звернення 20.03.2022).

12. Semantic structure search device and semantic structure search method: патент US20160217207 США : МПК G06F 17/30. №14995775, заявл. 14.01.2016, опубл.

28.07.2016. URL: https://patentscope.wipo.int/search/en/detail.jsf?docId=US175462817&_cid=P21-L0ZXKG-74985-5 (дата звернення 20.03.2022).

13. Method and system for semantic search against a document collection: патент US20130232171 США: МПК G06F 17/30, G06F 7/00, G06Q 50/00. №13869327, заявл. 24.04.2013, опубл.05.09.2013. URL: https://patentscope.wipo.int/search/en/detail.jsf?docId=US91072316&_cid=P21-L0ZWY9-72194-1 (дата звернення 20.03.2022).

14. Є. В. Левус, Н. І. Нечипір, Ю. В. Полиняк. Аналіз алгоритму Apriori для структурованих та неструктурованих даних. *Інформаційні системи та мережі*. 2017. №6, с.62-68. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2018/jun/13003/ilovepdfcom-62-68.pdf (дата звернення 01.05.2022).

15. Дашенков Д. С. Дослідження методів автоматичної відповіді на запитання. URL: https://openarchive.nure.ua/bitstream/document/18861/1/2021_M_PI_Dashenkov_DS.pdf (дата звернення 01.05.2022)

16. Універсальний багатозначний функціональний перетворювач як базова комірка семантичної мережі інтелектуальної системи / Четвериков Г.Г., Пузик О. С., Курасова В. В., Божко І. К. *Біоніка інтелекту*. 2016, №1, с.164-168. URL: https://openarchive.nure.ua/bitstream/document/4854/1/Chetverikov_GG_164-168.pdf (дата звернення 01.05.2022).

17. Amazon Web Services. Про продукти. URL: https://aws.amazon.com/ru/ (дата звернення 01.05.2022).

18. Terraform by Hashicorp – головна сторінка. URL: https://learn.hashicorp.com/ (дата звернення 01.05.2022).

19. Погуляєв Ю. С., Лановий О. Ф. Дослідження методів структурованого пошуку інформації в розподілених сховищах даних // Сучасні напрями розвитку інформаційно-коммунікаційних технологій та засобів управління. Тези доповідей дванадцатої міжнародної науково-технічної конференції 27 – 28 квітня 2022 року. – Баку – Харків – Жиліна, 2022, т. 2. с. 137.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

15. Дашенков Д. С. Дослідження методів автоматичної відповіді на запитання. URL: https://openarchive.nure.ua/bitstream/document/18861/1/2021_M_PI_Dashenkov_DS.pdf (дата звернення 01.05.2022)

16. Універсальний багатозначний функціональний перетворювач як базова комірка семантичної мережі інтелектуальної системи / Четвериков Г.Г., Пузик О. С., Курасова В. В., Божко І. К. *Біоніка інтелекту*. 2016, №1, с.164-168. URL: https://openarchive.nure.ua/bitstream/document/4854/1/Chetverikov_GG_164-168.pdf (дата звернення 01.05.2022).

19. Погуляєв Ю. С., Лановий О. Ф. Дослідження методів структурованого пошуку інформації в розподілених сховищах даних // Сучасні напрями розвитку інформаційно-коммунікаційних технологій та засобів управління. Тези доповідей дванадцатої міжнародної науково-технічної конференції 27 – 28 квітня 2022 року. – Баку – Харків – Жиліна, 2022, т. 2. с. 137.