Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра _____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

_____другий (магістерський)_____
(рівень вищої освіти)

Дослідження методів семантичного аналізу для пошукових механізмів
(тема)

Виконала: студент 2 курсу, групи ПЗСм-18-1_____

спеціальності 121- Інженерія програмного забезпечення____
(код і повна назва спеціальності)

Освітньо-професійної програми
_Програмне забезпечення систем_____

_____Хікматова Д. М._____
(прізвище, ініціали)

Керівник _____д. т. н. проф. Четвериков Г. Г._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

2019 р.

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)

освітньо-професійна програма _Програмне забезпеченнясистем_____

(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«_____»_____ 2019 р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Хікматовій Діані Марсіловні_____

(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження методів семантичного аналізу для пошукових механізмів_____

затверджена наказом по університету від "___" _____2019 р № _____

2. Термін подання студентом роботи до екзаменаційної комісії _____ 2019 р.

3. Вихідні дані до роботи *алгоритми векторизації, алгоритми тематичного моделювання, формули оцінки тексту для сприйняття, пояснювальна записка. Використовувати ОС Windows*

4. Перелік питань, що потрібно опрацювати в роботі *аналіз проблемної галузі і постановка задачі, огляд методів семантичного аналізу у пошуку даних, огляд методів кластеризації для тематичного моделювання, огляд методів визначення складності текстів для сприйняття*

5. Консультанти розділів роботи

Найменування Розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		Підпис	дата
Спецчастина	д.т.н. проф. Четвериков Г. Г.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Аналіз предметної галузі	20 вересня 2019р.	
2.	Огляд існуючих методів	7 жовтня 2019р.	
3.	Проведення дослідженння методів семантичного аналізу для семантичного пошуку	18 листопада 2019р.	
4.	Підготовка пояснювальної записки	27 листопада 2019р.	
5.	Спецчастина		
6.	Підготовка презентації та доповіді		
7.	Попередній захист		
8.	Нормоконтроль, рецензування		
9.	Занесення диплома в електронний архів		
10.	Допуск до захисту у зав. Кафедри		

Дата видачі завдання _____ _____2019 р.

Студент _____
(підпис)

Керівник роботи _____ __ д.т.н. проф. Четвериков Г. Г._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Атестаційна робота містить: 87 с., 19 рис., 19 формул, 2 додатки, 30 джерел.

СЕМАНТИЧНИЙ АНАЛІЗ, РОЗВІДУВАЛЬНИЙ ПОШУК, КЛАСТЕРИЗАЦІЯ, ВЕКТОРИ, ТЕМАТИЧНЕ МОДЕЛЮВАННЯ, PYTHON, BIGARTM.

Об’єкти дослідження – методи семантичного аналізу, тематичне моделювання та розвідувальний пошук.

Метою роботи дослідження є розробка моделі розвідувального пошуку, яка враховує складність сприйняття матеріалів.

Методи розробки базуються на використанні мови програмування Python та наступних бібліотек: NLTK для попередньої обробки тексту, TensorFlow для побудови нейронної мережі, BigARTM для тематичного моделювання та Pandas для аналізу даних. Щодо математичних методів, будуть використані такі підходи як векторизація, нечітка кластеризація, нейронні мережі.

У результаті дослідження проаналізовано предметну галузь та існуючі аналоги, поставлено задачу розробити систему розвідувального пошуку, що враховує складність тексту для сприйняття, досліджено вплив методів адитивної регуляризації на тематичну модель, підібрано оптимальну кількість тем та проходів по колекції документів для обраного датасету, а також порівняно роботу формул оцінки лекгості читання текстів (Фліша, Ганнінга та Дейла-Челла).

SEMANTIC ANALYSIS, EXPLORATORY SEARCH, CLUSTERING, VECTORS, TOPIC MODELING, PYTHON, BIGARTM.

The object of research is semantic analysis methods, topic modeling and exploratory search.

The aim is the development of the exploratory search model which considers the difficulty of the material for understanding.

Methods of developing are based on using of Python programing language and its following libraries: NLTK for the text preprocessing, TensorFlow for the creation of neural network, BigARTM for the topic modeling and Pandas for the data analysis. There also will be used such mathematic approaches as vectorization, fuzzy clustering and neural networks.

As a result of the research subject domains and existing analogues were analyzed, the goal was set to develop the exploratory search system which considers difficulty of the material for understanding, explored the influence of additive regularization to the topic model, matched the optimal amounts of topics and going over the document collection for the particular dataset and compared the work of readability formulas (Flesch, Ganning and Dale-Chell).

ЗМІСТ

ВСТУП

Технології усе ближче до того, щоб розуміти зміст тексту, і за цим стоїть один із розділів комп'ютерної лінгвістики – семантичний аналіз [1]. Незважаючи на постійне зростання даних, орієнтуватися у них людині все ще важко. Цікаво, що хоча традиційні повнотекстові пошукові системи і доволі успішно справляються із підбором матеріалу, якість цих результатів суттєво залежить від вдало сформованого запиту. Цікавою новою парадигмою є розвідувальний пошук, мета якого полегшити людині процес навчання та шукати по предметним галузям. Його головна мета – допомогти людям швидко розібратися із новою для неї предметною галуззю. Коли людина прагне ознайомитися із чимось новим для неї: часто вона не знає, як називається те, що вона хоче знайти, або взагалі не знає, що шукати, тому не може одразу добре сформувати запит, через що пошук відбувається ітеративно, доки вона зрештою не сформує необхідний запит. На відміну від повнотекстового пошуку, в його основі лежить тематичне моделювання. У рамках даної парадигми людина може знайти матеріали галузі, із якою хоче ознайомитися, використовуючи лише тему або документ, із якого система сама вилучить використані теми.

Мета – покращення підбору результатів розвідувального пошуку.

Об'єктом дослідження є застосування методів семантичного аналізу з метою покращення роботи пошукових механізмів. Предметом дослідження було обрано підбір оптимальних результатів розвідувального пошуку.

У межах роботи були використані як теоретичні, так і емпіричні методи дослідження. Теоретичні, зокрема, аналіз та синтез, були застосовані для аналізу предметної галузі та пошуку можливих рішень для покращення роботи пошукових механізмів. Емпіричні були використані для самого дослідження: експеримент та вимірювання для тематичного моделювання, а також експеримент, вимірювання та порівняння для вибору найкращого методу оцінки складності тексту.

У результаті дослідження була побудована тематична модель, яку було натреновано на датасеті статей Вікіпедії. Оцінка роботи тематичної моделі

відбувалася за допомогою таких параметрів як розрідженість матриць розподілу слів у темі та тем у документів, перплексія, інтерпретуємість тем та когерентіність. Завдяки застосуванню методів адитивної регуляризації були значно збільшені показники розрідженості матриць, що дозволяє зробити теми більш конкретними завдяки видаленню стоп-слів, а також оптимізувати роботу алгоритму. Окрім цього, вдалося виявити зв'язок між перплексією та когерентністю, що дозволяє визначити оптимальну кількість проходів по колекції з метою покращення інтепретування результатів тематичного моделювання. Застосування автоматичного виправлення помилок для текстів, згенерованих користувачами, було визначено ризикованим. Рекомендується або виконувати попереднє вилучення іменованих сутностей. Ще одним напрямком дослідження стало порівняння методів оцінки складності тексту (Фліша, Дейла-Челла та Ганнінга). Виявилося, метод Ганнінга найбільше корелює із результатами двох інших формул. Також цікавим виявилося те, що найскладніші тексти усіма трьома методами були визначені однозначно. Найкращі показники швидкості роботи показав метод Ганнінга. Таким чином, формули оцінки легкості читання можуть бути застосовані для підбору результатів пошуковими механізмами. Однак, усі три формули враховують довжину речень для визначення складності і при проведенні дослідження було виявлено, що при видаленні форматування списки були помилково сприйняті як довгі речення, тому при проведенні оцінки складності читання рекомендується враховувати оригінальне форматування текстів. Тож, завдяки проведеному дослідженню дістав подальшого розвитку напрямок розвідувального пошуку.

Результати проведеного дослідження можуть бути застосовані для розробки систем, у яких використовується тематичне моделювання, у тому числі для розвідувального пошуку, а також для систем, що надають користувачам текстовий контент, а особливо для пошукових систем для покращення досвіду користувача.

Після виконання дослідження його результати були застосовані для розробки системи розвідувального пошуку, яка дозволяє підбирати матеріали для навчання за темами та документом, а також сортувати їх за легкістю для сприйняття.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Застосування семантичного аналізу та його методи

Сьогодні кількість доступних даних невпинно та дуже швидко зростає, у зв’язку із чим зростає навантаження на людину, зокрема на її когнітивні можливості. У зв’язку із цим усе більш актуальною стає автоматична обробка тексту. Цікавою та перспективною галуззю автоматичної обробки є семантичний аналіз, мета якої розуміння змісту тексту.

Серед підходів до семантичного аналізу виділяють два види: заснований на правилах та прихований семантичний аналіз.

Перший підхід включає у себе роботу із онтологіями (великими структурами даних, у який зберігаються співвідношення лексичних одиниць та їх значень, а також перелічені їх синоніми), синтаксичний аналіз (здебільшого за допомогою складання дерев), а також велика кількість прописаних правил. Усе це зазвичай потребує великого вкладу людських ресурсів, але натомість можна отримати систему, що має прозорий процес роботи та дає результати, що легко інтерпретувати.

Підхід прихованого семантичного аналізу базується на опрацюванні великих обсягів текстових даних за допомогою методів машинного навчання. Це дозволяє знаходити зв’язки між словами, за допомогою яких і здійснюється аналіз. Недоліком такого підходу є те, що його результати важко інтерпретувати через те, що процес роботи є занадто «прихованим», через що складно зрозуміти, що призвело до певних результатів. Однак, якщо йде мова про опрацювання великих обсягів даних, прихований семантичний аналіз може показувати доволі точні результати навіть із відносно невеликим вкладом у його розробку, оскільки відсутня потреба в онтологіях та правилах. Хоча варто відмітити, що комбіновані підходи також можуть бути використані для семантичного аналізу [1].

Семантичний аналіз має великий перелік сфер застосування. До них можна віднести наступні:

- класифікація та кластеризація тексту;
- тематичне моделювання;
- системи «запит-відповідь»;
- перевірка на плагіат;
- аналіз стилістики твору та виявлення авторства;
- фільтрація спаму;
- прогнозування подій на основі тексту;
- референсування тексту (зменшення обсягу тексту та вилучення його короткого змісту) та створення анотацій (перелік ключових тем або речень тексту) до нього;
- виявлення настрою автору;
- аналіз ставлення на емоційного забарвлення тексту;
- адаптація творів;
- машинний переклад;
- виділення сутностей та фактів із тексту;
- інформаційний пошук та інше.

1.2 Особливості інформаційного пошуку та пошукових механізмів

Інформаційний пошук – це процес пошуку неструктурованих даних, які задовольняють запиту, тобто ті, які містять необхідні факти, відомості чи дані. Для здійснення інформаційного пошуку користувач формує запит, який оброблює пошуковий механізм. Система шукає серед масивів інформації необхідні дані та представляє їх користувачеві (зазвичай попередньо ранжуючи відповідно до певних критеріїв), після чого користувач ознайомлюється із результатами та оцінює їх. Звісно, оцінка у більшості випадків відбувається неявно, тому

пошуковим механізмам доводиться шукати додаткові шляхи, як це можна дізнатися.

Існує кілька видів пошуку: повнотекстовий (часто для прискорення роботи використовується індексування), за метаданими (атрибутами документу) та по зображенням (коли користувач завантажує зображення, а отримує схожі). Тим не менше, є ще один вид пошуку – семантичний, який шукає документи за змістом, а не просто за ключовими словами, як це робить повнотекстовий пошук [2].

Пошук зазвичай складається із двох складових: запит та об’єкт пошуку.

При пошуку інформації вдалий запит є одним із ключових моментів. Саме від нього більшою мірою і буде залежати, знати користувач те, що хотів, чи ні, та скільки спроб на це піде.

Перш за все, система, яка працює із різними мовами, має визначити, якою мовою написано запит.

Наступним етапом при обробці запиту є перевірка на друкарські помилки. Добре спроектована система пропонує автоматичні пропозиції на основі введеної частини тексту, таким чином допомагаючи людини одночасно і уникнути помилок, і краще сформувати запит. Якщо ж помилка все-таки трапилася, краще запропонувати автоматично скоригувати слово або запропонувати пошук за запитом із вже виправленими помилками. Цікаво, що у деяких системах, що базуються на навчанні, можлива випадки, коли система вважає правильне написання невірним. Це відбувається тому, що більшість людей вводить його неправильно. Таких ситуацій варто уникати.

При аналізі запиту пошуковий механізм має вирішувати проблему синонімії та омонімії. У першому випадку система має пропонувати результати, які мають схожий зміст, хоча ключові слова і відрізняються від запиту. У випадку омонімії механізм має розпізнати, що саме має на увазі користувач, якщо слово може мати кілька значень. Зазвичай це вирішується за допомогою інших слів, що знаходяться у запиті та створюють контекст. Однак, можливі й інші підходи, які можуть покращити вгадування. Наприклад, можна враховувати попередні запити

користувача (скоріше за все він шукатиме інформацію, що відноситься до однієї предметної галузі).

Ще однією проблемою, яка виникає при пошуку за запитом, є виявлення зав’язків між словами у запиті, тобто які із слів із більшою вірогідністю можуть бути використані як словосполучення та взагалі які відношення можуть мати слова один до одного.

Інколи може виникнути ситуація, коли з’являються нові слова, яких ще немає у морфологічному словнику. У такому випадку система має «вивчити» це нове слово, самостійно скануючи ряд відомих їй словників (інших ресурсів). Іншим варіантом є пошук контекстів, у яких використовується слово, щоб спробувати зрозуміти його семантику. Для цього зазвичай використовують метод скіп-грам.

Важливе значення для пошукових механізмів відіграє також унікальність тексту. Таким чином, при підборі результатів пошуку алгоритм має враховувати оригінальність, визначення якої може займати надзвичайно багато часу та ресурсів, якщо порівнювати абсолютно усі тексти. З цієї причини механізми використовують спеціальні прийоми. Наприклад, може бути використаний метод n-грами, який працює із ймовірностями та завдяки цьому може сканувати не усі слова, а пропускати деякі з певним кроком. Ще один підхід полягає у наданні більшої ваги рідкісним словам (які не використовуються всюди), якщо вони наявні на сайті.

Окрім вищезгаданого, системи веб-пошуку можуть враховувати метадані сторінок. Втім, застосунки, які працюють із іншими форматами даних, також можуть використовувати схожий підхід, впровадивши метадані або теги.

У випадку із веб пошуком, аспектів, які можуть аналізуватися, є дещо більше. Наприклад, люди можуть шукати не тільки визначення чи якусь інформацію, але й прагнути виконати якусь дію (читати, придбати, слухати та інше). Таким чином, пошуковий механізм має знайти, яке слово у записі є інтентом, тобто наміром або задачею, яка може бути пов’язана із словом. Складність даної задачі полягає у тому, що не кожне дієслово (і взагалі це може бути й інша частина мови або словосполучення) у запиті може означати намір. Цікаво, що зазвичай намір люди ставлять у кінець запиту. У певній мірі це пов’язано із використанням пошуковими

механізмами автоматичних пропозицій. Веб-механізми аналізують також багато інших факторів, таких як структура HTML сторінки, навігація, форматування тексту, якість тексту (за SEO показниками), наявність посилань та їх кількість, а також багато іншого, що пов’язано із використанням веб-елементів.

Окрім лінгвістичних аспектів можуть бути враховані ще й такі фактори, як скільки часу користувач перебуває на певному ресурсі (наприклад, коли він клікнув на посилання, але одразу закрив, значить, що даний результат не підійшов) або чи переходить користувач за результатами пошуку взагалі (зазвичай це означає, що жоден із результатів пошуку не підійшов і у таких випадках користувач зазвичай намагається використати інший варіант запиту).

Таким чином, для отримання точних та вдалих результатів необхідний добре сформований запит. Проблема виникає тоді, коли користувач не може вдало сформувати запит через те, що він ще не знайомий із предметною галуззю, а отже не може вірно сформувати запит через брак ключових слів. У таких випадках людина зазвичай має ітераційно працювати із пошуковим механізмом, тобто багаторазово повторювати запити, постійно вдосконалюючи їх, базуючись на знаннях, отриманих із результатів попередніх запитів. А бувають і випадки, коли людина взагалі не знає, що вона прагне зайти, у неї нема конкретної мети, вона просто хоче ознайомитися із новою для неї галуззю. Обидва випадки потребують величезної кількості часу, інколи марно витраченої, оскільки користувач має власноруч переглядати на аналізувати текстові дані, серед яких чимало будуть дублюватися або виявлятимуться непотрібними.

Цікаво, що через постійне збільшення даних користувачі почали сканувати текст замість його послідовного читання. І дійсно, повне перечитування тексту зайняло б просто неймовірну кількість часу. Люди зазвичай сканують текст результатів пошуку, і, якщо результати поточного запиту не дали бажаної відповіді, вони часто намагаються покращити наступний за рахунок отриманої із цих результатів інформації. На жаль, незрячі люди не можуть так просто просканувати дані. Звісно, існують спеціальні засоби браузерів, що озвучують контент (screenreaders), але читання або займе багато часу, або значно підвищить ризики

пропустити важливі дані, оскільки багато буде залежати від інформаційної структури сторінки та реалізації. У зв’язку із перерахованим вище, усе більше дослідників починають цікавитися розвідувальним пошуком.

Даний підхід є доволі новою парадигмою інформаційного пошуку, яка направлена на отримання та систематизацію знань, самоосвіту. Він може бути використаний для пошуку наукової інформації, соціологічних досліджень, аналізу новин та, звісно, ознайомлення із зовсім новою інформацією у повсякденному житті. Потенційними користувачами є спеціалісти різноманітних галузей, дослідники, викладачі, студенти та пересічні люди, що прагнуть опанувати нову галузь. Враховую те, що у наш час, чимало професій потребують знань із суміжних галузей, та і взагалі люди стають більш схильними змінювати сфери діяльності протягом життя, дана парадигма стає усе більш актуальною.

Розвідувальний пошук дозволяє легко аналізувати нові предметні галузі, розуміти їх структуру та відношення, можливо, навіть відображення розвитку у часі. Зрозуміло, що результати такого пошуку будуть відрізнятися від тих, до як ми зараз звикли, тому проблема візуалізації таких даних є зараз актуальною. Іншими аспектами, на які також варто звернути увагу, є моніторинг нових даних та розробка рекомендацій на основі дій користувача (із якими документами він уже ознайомився).

Один із варіантів розвідувального пошуку – це пошук за документом або навіть колекцією документів. Припустимо, людина має документ, із яким вона має розібратися. Як відомо, один документ може містити кілька тем. Людині буде важко розібратися зі змістом документу, якщо вона погано орієнтується в темах документу. Фактично, у цьому разі пошук все ще відбувається за темами, оскільки ключові теми документу буде визначено за допомогою частоти, із якою слова зустрічаються у документі, а також семантичного аналізу. У випадку із колекцією документів, вилучення тем відбуватиметься одразу із усіх документів. Таким чином, задача розвідувального пошуку у даному випадку полягає у вилученні тем із документу та підбору матеріалів по цим темам. Базуючись на отриманих результатах, людині буде легше відійти від парадигми ітераційного пошуку, що

може суттєво зменшити час на пошуки необхідної інформації, а також знизити вірогідність упущення важливих даних.

1.3 Огляд аналогів та патентів

Для розуміння поточної ситуації у галузі інформаційного пошуку були проаналізовані кілька найбільш відомих систем-аналогів, а також розглянуто наукові публікації щодо розвідувального пошуку та тематичного моделювання, як його важливої складової.

Google Scholar – це пошукова система по текстам наукових публікацій для усіх предметних галузей та форматів (див. рис. 1.1).

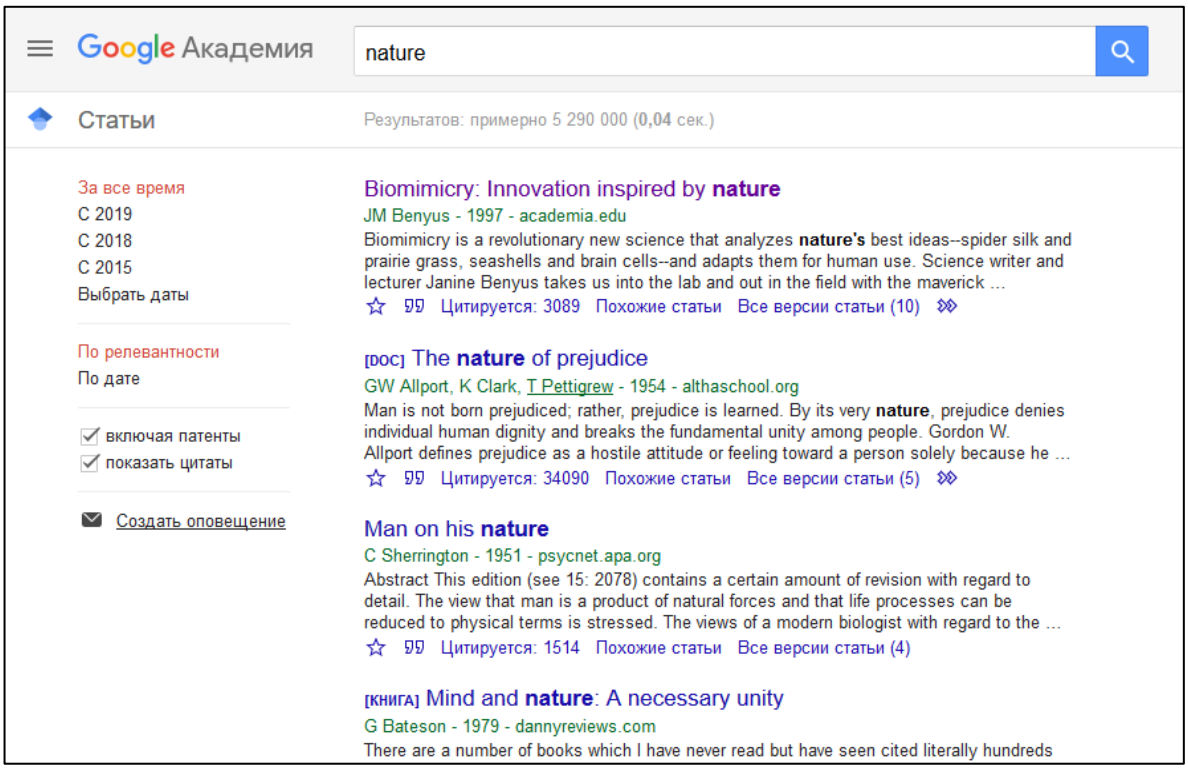

Рисунок 1.1 – Приклад роботи платформи Google Scholar

Результати сортуються, параметр обирає користувач, можна ранжирувати комбіновано. Система дає більшу вагу публікаціям, які часто цитуються.

Окрім простого пошуку, платформа передбачає і розширений для пошуку у конкретних статтях та журналах.

Ще однією системою, яка сьогодні доволі популярна, є Semantic Scholar (див. рис. 1.2), яка являє собою платформу для пошуку наукових публікацій.

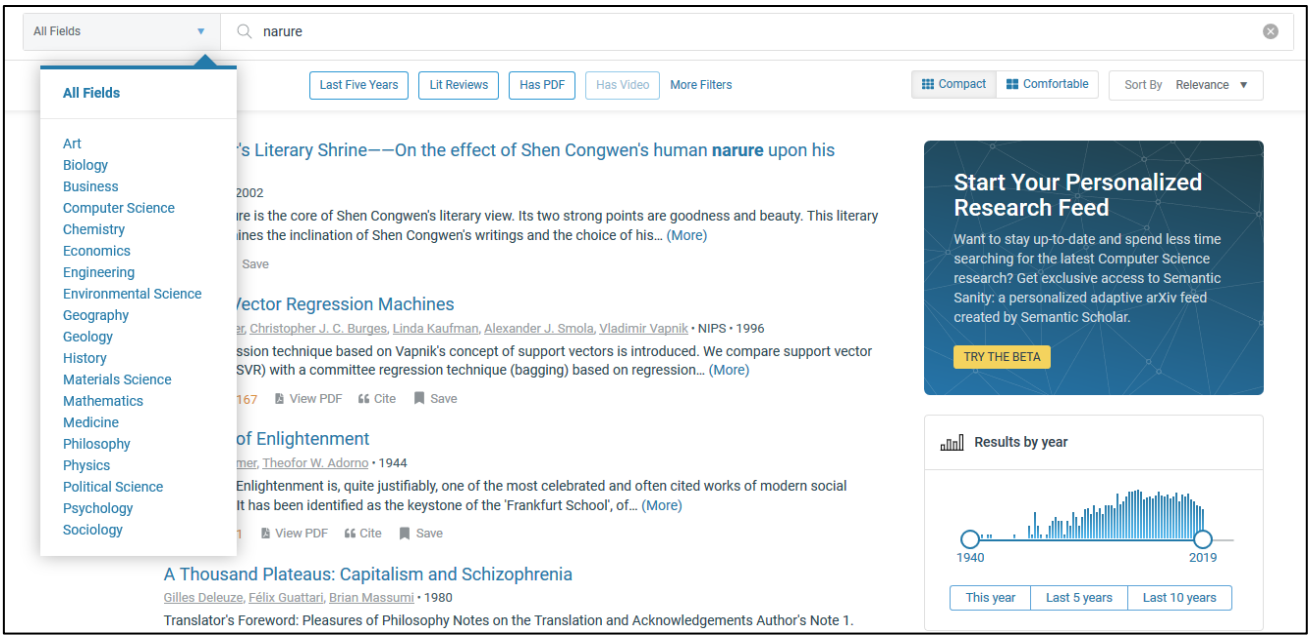

Рисунок 1.2 – Приклад роботи платформи Semantic Scholar

Пошук забезпечується цитуванням, а також семантичним аналізом, що працює за допомогою машинного навчання, обробки натуральної мови та машинного зору. Дана платформа є цікавою з точки зору того, що враховує цитування авторів. Тобто чим більше разів публікації автору цитувалися, тим вищим буде його рейтинг. Також система дозволяє більш детально налаштовувати сортування статей, може показувати деякі цифр та діаграми, виділяє ключові слова, що допомагають краще аналізувати інформацію. На додачу до звичайного пошуку, користувач також має змогу відразу фільтрувати за категорією. Можливість розвідувального пошуку у даній системі відсутня.

PubMed – це архів медичних та біологічних публікацій. Звичайний режим пошуку є простим та зручним у використанні – лише поле пошуку та кнопка. Окрім звичайного пошуку, дана платформа також дає можливість прогресивного пошуку: використання булевих операторів, категоризація, робота із синонімами, робота із

індексами. Система використовує MeSH (Medical Subject Headings), який являє собою великий словник, що індексує статті та книги природничого характеру. Також він має змогу зберігати історію пошуку та проміжних результатів, дає можливість переглядати деталі щодо того, як пошуковий механізм «зрозумів» запит, і змінювати формати виводу документів (Abstract, Free full та інші). Тож, платформа дозволяє доволі ефективно шукати конкретні пошукові запити за допомогою великої кількості налаштувань. Тим не менше, користувачеві, який тільки хоче ознайомитися із предметною галуззю та ще не може точно сформувати запит (або навіть не знає, з чого почати), буде доволі складно користуватися системою. Однак, це вузькоспеціалізована платформа (що у певній мірі є її недоліком), тому більшість користувачів мають конкретну мету пошуку.

Microsoft Academic – це ще одна безкоштовна веб-платформа для пошуку наукових публікації та літератури (див. рис. 1.3).

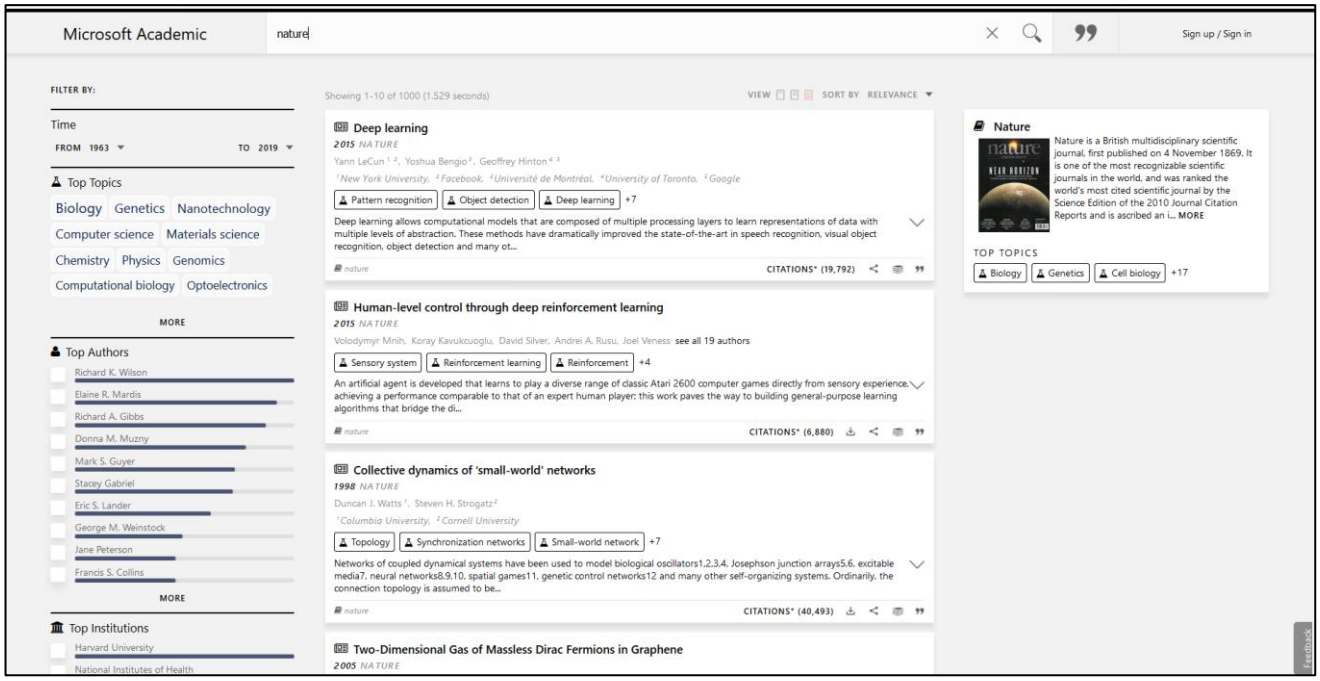

Рисунок 1.3 - Приклад роботи платформи Microsoft Academic

Дана система використовує семантичний пошук та індексує понад 220 мільйонів публікацій. Microsoft Academic дозволяє уточнити запит відповідно до

домену, а також підбирає рекомендації для публікацій, але не зберігає історію. Існує можливість використання фільтрів, сортування та роботи із цитатами.

Інший аналог – ResearchGate (див. рис. 1.4), який являє собою соціальну мережу, яка забезпечує співробітництво між вченими різний предметних галузей.

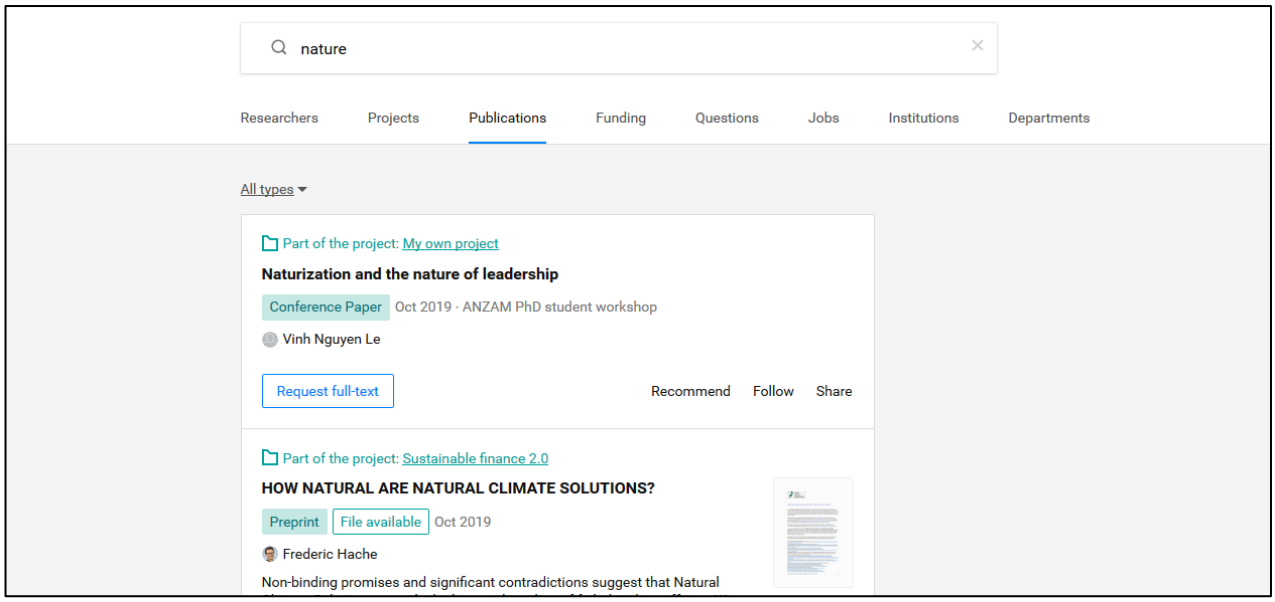

Рисунок 1.4 - Приклад роботи платформи ResearchGate

Серед наявних функцій: семантичний пошук (у тому числі по анотації), форуми, можливість спільної роботи над файлами, обмін публікаціями та інше. Семантичний пошук працює за допомогою індексації як публікацій внутрішніх ресурсів, так і сторонніх баз, окрім ключових слів для пошуку використовується ще й семантичне значення слів, що покращує результат, особливо при пошуку за анотацією. Можна шукати по різним категоріям (наприклад, люди, проекти, публікації). Окрім цього, система зберігає історію використання.

Загалом, Research Gate – це не єдина відома пошукова платформа, акцент якої зроблений на соціальній взаємодії. Серед подібних систем aMiner, Academia.edu та інші.

У одному із знайдених патентів були запропоновані наступні покращення до існуючої ідеї для пошуку, що базується на багатьох запитах (що, по суті, дуже схоже на розвідувальний пошук): збереження та поновлення контексту пошуку, збереження історії запитів, можливість збереження користувачем корисних

результатів, усунення надлишкових результатів, інтеграцію пошуку із навігацією, співпраця із кількома пошуковими системами та автоматичне підведення підсумків пошуку [3].

Досі залишається невирішеним питання зручного та інформативного відображення результатів розвідувального пошуку, у зв'язку з чим деякі дослідники пропонують свої варіанти.

Цікавим є підхід, обраний веб-застосунком Aemoo. Його ідея полягає у тому, що користувач може запросити у даної пошукової системи інформацію про будь-яку сутність, а результати будуть агреговані із різних ресурсів, таких як Вікіпедія, Твіттер, Google новини [4]. Окрім цього, дана система також візуально відображає зв'язки між темами та дає короткі відомості щодо них (див. рис. 1.5).

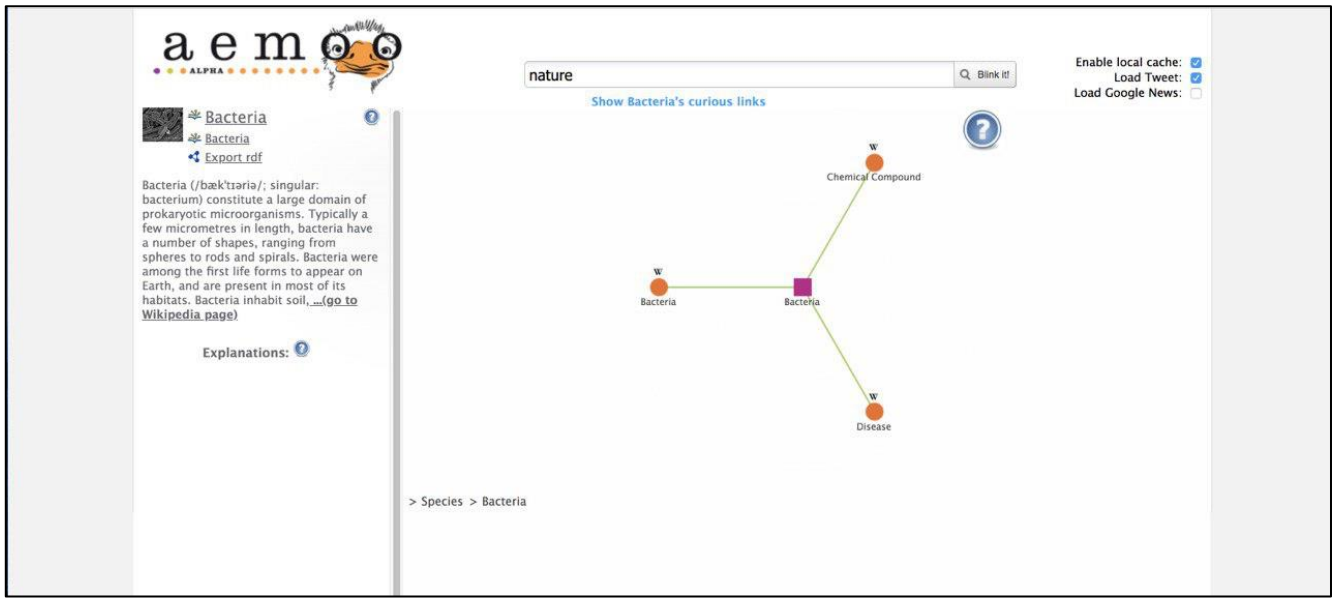

Рисунок 1.5 – Приклад роботи Aemoo

Доволі багато знайдених публікацій присвячені проблемі колективного пошуку, тобто коли одразу кілька людей намагаються розібратися у певній галузі. Така можливість є особливістю системи CollabSearch. Для покращення спільної роботи вони використовують чати. В одному із досліджень зазначається, що чат часто є джерелом ідей для запитів [5].

Вартим уваги є пошуковий застосунок TweetMotif для Twitter [6]. Його ідея полягає у групуванні повідомлень за термінами, які часто зустрічаються. Таким

чином, отримуються підтеми результуючого набору, які полегшують пошук, навігацію та сприйняття інформації.

Деякі дослідження направлені на оцінку результатів пошуку, а також на персоналізацію пошуку. Чимало дослідників пропонують оцінювати якість результатів самим користувачам (аналізуючи їх поведінку, наприклад). Персоналізація пошуку може здійснюватися як за допомогою аналізу дій користувача, так і завдяки виконання спеціальних налаштувань в особистому кабінеті. У результаті одного с досліджень згадується про проблематичність оцінки результатів у випадку пошуку без специфічної мети [7].

В одній із публікацій [8] була висунута доволі цікава ідея щодо використання гіперпосилань для зв'язків документів між собою, таким чином об'єднуючи схожі теми та покращуючи процес навчання.

Деякі дослідження направлені на пошук за науковими публікаціями. В одній із них пропонується використовувати метадані [9]. Очікується, що вони не будуть показуватися користувачеві у результатах пошуку, але будуть враховуватися пошуковими механізмами. Окрім того, дана робота пропонує також генерацію гіперпосилань для іменованих сутностей з метою надання можливості їх подальшого дослідження.

Пропонуються і підходи дещо відмінні від пошуку за матеріалами. Наприклад, одним із варіантів є об'єднання семантичних метаданих у базу знань, яка являтиме собою графічне представлення [10]. Планується, що у такій системі користувач переходить по кліку до веб-ресурсу, документу, тощо.

Чимало досліджень направлені на розробку систем розвідувального пошуку, які б враховували лінію часу. Більшість із них пропонують різноманітні рішення для інтерфейсу користувача. У роботі «Exploratory search using timelines» висвітлюється пропозиція відображати матеріали на лінії часу аби користувачам було легше знаходити актуальні дані [11], а також ставало зрозуміло, як розвивалася та чи інша галузь. Окрім безпосередніх назв матеріалів пропонується також відображати основні метадані до них.

Дещо незвичний спосіб представлення пошуку запропонували розробники інтерфейсу NameSieve [12]. Ідея відображення полягає у відображенні тем у вигляді хмар слів та демонстрації семантичних анотацій користувачам.

Серед розглянутих аналогів найближчою до розвідувального пошуку найбільш популярною є Research Gate, оскільки вона реалізує можливість розширеного пошуку, його налаштувань, а також пошуку за анотаціями, що дуже схоже на пошук за документом у розвідувальному пошуку, хоча у даній системі акцент зроблено не на навчанні та ознайомленні, а скоріше саме на знаходженні публікацій, що найбільше відповідають короткому опису. Серед вже існуючих систем розвідувального пошуку варто назвати Aemoo та CollabSearch.

Цікаво, що жодна із розглянутих систем чи патентів не враховує ранжування за складністю документів для розуміння. Але це також важливо для користувачів, оскільки деякі спеціалісти можуть доволі гарно орієнтуватися у предметній галузі і прагнути поглибити свої знання. У такому випадку вони скоріше за все забажають ознайомитися із більш складними матеріалами. Актуально це може бути і для студентів чи наукових співробітників, які бажають не просто зрозуміти матеріал, але ще й використати набуті знання у наукових цілях. У випадку із користувачем, який є новачком у галузі, складні наукові статті можуть практично не нести користі. Такі люди зазвичай прагнуть зрозуміти саму ідею, а для цього краще підходять матеріали, написані простою мовою, без поглиблення у складні поняття.

На даний момент практично відсутні відомі платформи для розвідувального пошуку. Цей напрямок є доволі новим, тому зараз ведуться активні дослідження та розробки у цій галузі. Кілька років тому компанія Яндекс також заявила про найближчий випуск подібного продукту, втім на сьогоднішній день система ще не почала працювати. Відомо, що їх ідеї лягли в основу розробки бібліотеки для тематичного моделювання BigARTM [13]. Вона заслуговує особливої уваги.

BigARTM – доволі нова та потужна бібліотека із відкритим кодом, у якій реалізовано багато методів із можливістю їх налаштувань. Їй передували бібліотеки PLSA (на основі імовірнісного латентного семантичного аналізу) та LDA (базується на латентному розміщенні Дириклє). Особливістю BigARTM є те, що її

принцип полягає у адитивної регуляризації, що дає змогу створювати моделі за допомогою готових модулів. До функцій бібліотеки можна віднести:

- регуляризатори (оптимізація критеріїв), які можна використовувати у різних комбінаціях, що допомагає покращити якість класифікації текстів та підвищує точність та повноту пошуку;
- опис об’єктів у документах, які не є текстом (наприклад, метадані, посилання та інші), за допомогою модальностей;
- можливість працювати із тематичними ієрархіями;
- обробка тексту у якості послідовності векторів слів;
- використання даних щодо частоти словосполучень, що зустрічаються;
- велика кількість метрик якості.

Ще однією особливістю BigARTM є можливість працювати із великими даними, що забезпечується розпаралелюванням на ядрах центрального процесору, пакетною обробкою даних, лінійною обчислювальною складністю, кешуванням та слідуванням сучасним стандартам програмування. Однак перед застосуванням даної бібліотеки необхідно попередньо обробити текст відповідно до цілей.

1.4 Постановка задачі

Після ознайомлення із підходами та методами, що застосовуються для пошуку та тематичного моделювання, наявними інструментами, а також існуючими аналогами, стало зрозуміло, що напрямок розвідувального пошуку є доволі новим та комерційних аналогів практично немає. Натомість існує чимало аналогів, в основі яких лежить розширений пошук, який включає у себе додаткові параметри. Найближчим за функціоналом виявилася платформа Research Gate, яка може виконувати семантичний пошук та шукати за анотаціями. Що стосується досліджень, патентів та публікацій, більшість із них направлена на рішення щодо відображення подібних нетипових результатів пошуку з точки зору інтерфейсу

користувача. У деяких публікаціях була висвітлена ідея застосування методів покращення результатів, які все успішно застосовуються для повнотекстового пошуку. Цікавою є бібліотека BigARTM, яка скоріше за все стане основою розвідувального пошуку компанії Яндекс, що вже займається його розробкою.

Також під час дослідження було виявлено, що жоден із розглянутих аналогів чи патентів не враховує рівень складності тексту, або іншими словами складність тексту для сприйняття, хоча це може бути важливим параметром для користувачів, що мають різний рівень знань у певній предметній галузі.

Тож, були поставлені наступні задачі:

- проаналізовані методи тематичного моделювання та обрати найкращий;
- підібрати оптимальні параметри для оптимізації роботи системи та підвищення якості виділених тем;
- дослідити вплив перплексіїї на інтерпретуємість тематичних моделей;
- проаналізувати роботу бібліотек для автоматичного виправлення помилок у текстах;
- порівняти роботу методів оцінки складності текстів для сприйняття та обрати найкращий;
- розробити прототип системи розвідувального пошуку, яка при підборі результатів сортуватиме матеріали за складністю сприйняття.

Планується, що система буде працювати із такими текстовими форматами як txt, docx та pdf.

Імплементація системи відбуватиметься мовою програмування Python із використанням наступних бібліотек: bigARTM, Pandas, textStat та nltk.

2 МЕТОДИ СЕМАНТИЧНОГО АНАЛІЗУ ТА КЛАСТЕРИЗАЦІЇ У ПОШУЦІ ДАНИХ

У розвідувальному пошуку часто використовується тематичне моделювання. Окрім розвідувального пошуку, тематичне моделювання застосовується ще й для класифікації за жанрами та категоризації документів, анотування одного чи кількох документів або навіть зображень (виявивши тему легше виділити ключові фрази), сегментації документів (виявлення у якій саме частині документа знаходиться потрібна інформація), пошуку зображення по тексту та тексту по зображенням, виявлення подій у потоках новин та тематичних спільнот у соціальних мережах [14], побудови профілів інтересів користувачів та для управління діалогом у системах «питань-відповідей».

Тема являє собою набір термінів, які часто зустрічаються у документах. Ідея полягає у тому, що зазвичай людина може зрозуміти із якої галузі документ, тобто із фізики він чи із історії, базуючись на словах, які зустрічаються. Кожна тема має свої словники (набори слів, що її характеризують). Основою тематичного моделювання є теорія вірогідності та статистики. Для визначення тематики тексту аналізується частота із якою слова зустрічаються у документі та колекції документів. Кожна тема має свою частоту певних слів та словосполучень. Варто також враховувати, що один документ може відноситися одразу до кількох тем.

2.1 Методи попередньої обробки тексту

Перед тим, як застосовувати формальні методи тематичного моделювання, варто врахувати необхідність попередньої обробки тексту, адже в залежності від формату документу у тексті можуть бути елементи форматування, переноси слів, залишки від графічних елементів на інше. Такі елементі мають бути усунуті з

тексту. Особливої попередньої обробки також потребують дуже короткі тексти (у таких випадках можна зливати кілька коротких фрагментів в один, щоб отримати кращі результати тематичного моделювання). Наступним кроком у роботі із текстом, в залежності від мови, є стемінг (знаходження основи для заданого початкового слова) та лематизація (процес приведення словоформи до нормальної мови), тобто вилучення основи слова. Третій крок – вилучення стоп-слів (які зустрічаються скрізь, прийменники, сполучники, займенники), слів із високою частотністю, а також дуже рідкісні слова з метою зменшення словника, а значить, оптимізації роботи моделі.

Далі потрібно сформувати базові допущення. Перше полягає у тому, що текст – це згенерована за допомогою імовірнісного розподілення послідовність слів, довільна перестановка слів не впливає на тему тексту. Друге – кожне слово у документі пов'язане із певною темою, причому саме тема генерує слова, а не документ. Третє – тема складається із відносно невеликої кількості ключових слів, а сам документ містить невелику кількість тем. Останнє допущення дозволяє оптимізувати роботу та отримати більш якісні результати.

2.2 Методи векторизаціїї

Після попередньої обробки тексту необхідно виділити у ньому ознаки (векторизація). Ідея даного методу полягає у тому, що для кожного слова знаходиться його векторне представлення у вигляді вектору чисел певної довжини. Довжина вектору означає кількість вимірів. Особливістю векторизації є те, що над векторами можна виконувати арифметичні дії для отримання семантичного значення.

Дуже поширеним підходом у семантичному аналізі є метод «мішка слів». Працює він наступним чином: незалежно від порядку слів у реченні (тексті, документі) будується вектор, довжина якого дорівнює кількості слів. Також

вказується або бінарна ознака (наявне дане слово чи ні) слова або кількість разів, скільки воно використовувалося.

Наступним кроком після «мішка слів» може бути використано Tf-Idf. Цей метод полягає у тому, що окрім простого підрахунку кількості слів у документі чи колекції документів, підраховується ще й відносна частота слів. Tf – це відношення кількості разів, скільки слово зустрічається у документі до сумарної кількості слів у документі. Idf – це відношення загальної кількості документів до кількості документів, у яких зустрічається конкретне слово. Tf велике, коло слово часто використовується у документі. Idf буде великим тоді, коли слово рідко зустрічається у колекції. Отож, за допомогою даного методу із більшою вірогідністю будуть отримані слова, які мають велике змістовне навантаження. Це пов'язано із тим, що у результаті будуть вилучені слова, які рідко зустрічаються взагалі, але часто використовуються у конкретному документі, а отже, не є фоновими. Таким чином, добуток Tf та Idf допомагає отримати більш точні відомості відносно ваги кожного слова, аніж звичайний мішок слів. Відповідно, чим більші значення Tf та Idf, тим більшим буде їх добуток.

Покращити результати Tf-Idf підходу можна, якщо враховувати не тільки слова, а послідовності слів (колокації). Це дає змогу зробити результати точнішими, але недоліком даного методу є суттєве збільшення словника (експоненціально в залежності від обраної кількості слів у послідовності), тому таку матрицю об'єктів-ознак становиться важко зберігати. Матриці являють собою відображення того, скільки разів слово зустрілося у документі. Одним із варіантів вирішення цієї проблеми є зберігання розріджених матриць (тобто вилучення нулів, коли слово чи словосполучення відсутнє).

Як «мішок слів», так і Tf-Idf допомагають вирішити задачу категоризації та класифікації документів.

Далі необхідно вирішувати безпосередню задачу класифікації чи кластеризації.

2.3 Методи кластеризації тексту

Класифікація – це розподілення деяких об’єктів по відомим класами за ознаками, які вибрані для визначення схожості чи відмінності між цими об’єктами. Кластеризація – це розподілення множини об’єктів на кластери (групи). Відмінність цих методів полягає у тому, що при кластеризації групи (класи) завчасно відомі. У випадку з кластеризацією перелік груп завчасно невідомий, він визначається під час роботи алгоритму. Об’єкти різних груп мають бути максимально схожими один із одним та максимально відрізнятися від об’єктів інших груп.

Одними із найбільш простих та популярних алгоритмів класифікації є лінійний класифікатор та наївний байєсівський класифікатор. Ідея першого полягає у тому, що існує деяка площина, яка ділить простір на півпростори. Даний алгоритм погано працює у випадку великої кількості ознак та погано перенавчається у цьому випадку.

Байєсівський класифікатор заснований на теоремі Байєса. Варто зазначити, що цей алгоритм передбачає, що наявність однієї ознаки не буде пов’язано із наявністю якоїсь іншої ознаки. Іншими словами, ознаки є незалежними одна від одної. Робота даного алгоритму полягає у наступних кроках:

- перетворити набір даних у частотну таблицю;
- створити таблицю правдоподібності;
- розрахувати вірогідність для класів за допомогою теореми Байєса.

Даний алгоритм можна використовувати для розбиття об’єктів на базі множинних ознак, тож може бути успішно застосований для багатокласової класифікації текстів. Цей метод дає доволі гарні результати, а також легко та швидко виконується. Серед його недоліків потрібно зазначити неможливість зробити прогноз, якщо у навчаючій вибірці не було певної ознаки, та допущення щодо незалежності ознак, яке у реальності трапляється доволі рідко.

Щодо кластеризації варто відразу зазначити, алгоритми поділяють на ієрархічні та плоскі, а також на чіткі та нечіткі. Плоскі алгоритми просто розбивають вибірку на кластери, що не перетинаються. Ієрархічні алгоритми будують цілу систему вкладених кластерів. У випадку чітких алгоритмів кожний об'єкт належить лише одному кластеру. Нечіткі алгоритми допускають можливість того, що кожен об'єкт може відноситися до кожного класу із деякою вірогідністю.

Прикладом алгоритму кластеризації тексту може слугувати метод LSA/LSI. У його основі лежать принципи факторного аналізу. Використовуючи статистичну інформацію, LSA допомагає впоратися із проблеми синонімії (кілька слів можуть мати одне й те ж значення) та омонімії (одне слово може мати кілька значень). Ідея алгоритму полягає у сингулярному розкладанні (декомпозиція матриці з метою її приведення до канонічного вигляду) tf-idf матриці. Таке розподілення допомагає краще відсіювати шуми та знаходити кластери.

До переваг цього методу можна віднести роботу із матрицями tf-idf, відсутність потреби у навчанні та попередньому налаштуванні відповідно до набору документів, гарно працює із прихованими залежностями. Мінусами методу є повільність роботи алгоритму через велику кількість обчислень, відсутність назв для отриманих факторів, а також відсутність можливості працювати із кластерами, що перетинаються.

Доволі цікавими прикладами кластеризації тексту є алгоритм Buckshot та Fractionation. У Buckshot береться випадкова вибірка документів, після чого на ній проводиться процедура пошуку кластерів, яка відбувається шляхом «склеювання» розташованих найближче один до одного документів. Так відбувається до тих пір, доки не буде знайдено задане число центрів. Після цього документам присвоюються виявлені центри, за принципом присвоєння до найближчого центру. Назва кластеру формується із слів, що найчастіше зустрічаються у ньому. Перевагами такого методу є висока швидкість роботи, відсутність необхідності у навчанні, використання матриці близькості документів; серед недоліків – невисока точність (або зниження швидкості при покращенні алгоритму), потреба у заданні

кількості кластерів та відсутність можливості працювати із кластерами, що перетинаються [15].

Одним із найбільш популярних алгоритмів кластеризації є метод k-means. У його основі лежить ітеративний процес стабілізації центроїдів кластерів. Його перевагами є лінійна швидкість роботи, а також те, що метод не потребує навчання і при необхідності може накопичувати відомості для збільшення точності у подальшому, серед недоліків – на вхід потрібна кількість кластерів, а також те, що кластери не можуть перетинатися, тобто алгоритм є чітким.

Альтернативою k-means є c-means [16]. Даний алгоритм дозволяє розбивати множину об’єктів на задану кількість нечітких множин. Він вважається удосконаленою версією k-means, оскільки надає змогу для кожного об’єкту розрахувати ступінь його належності до кожного із кластерів.

Є й інші моделі для роботи із нечіткою класифікацією: метод Монте-Карло із використанням ланцюгів Маркова (працює із вибірками розподілення вірогідності); MaxMax, який працює із графовими структурами даних та дозволяє одній вершині знаходитися у кількох кластерах; WatSet, в основі якого також лежить робота із графами та групування синонімічних слів (вершин). Однак, перший алгоритм є доволі складним, другий та третій – застосовуються тільки із графовими структурами даних.

Стає усе більш поширеним підхід нейронний мереж. Вони гарно працюють у випадках, коли багато текстів та є доволі гнучкими, оскільки підходять для різноманітних архітектур, але їх недолік у перенавчанні.

У випадку простих використання підходів (байєсівський класифікатор, k-means) моделі можуть бути не дуже гарно «навчені», але вони легко інтерпретуються, що полегшує розуміння коригування недоліків. Нейронні мережі хоч часто і працюють дуже добре, важко інтерпретуються, оскільки являють собою «чорний ящик», та потребують перенавчання.

Окрім згаданих методів, бувають і випадки розробки спеціальних методів для рішення конкретної задачі.

2.4 Аналіз складності сприйняття тексту

Легкість сприйняття письмового тексту (англ. readability) – це легкість, із якою користувач може сприймати та розуміти написаний текст. Зрозуміло, що під час вивчання нового матеріалу легкість сприйняття є важливою. Відомо, що тексти, які легше читаються, сприяють більш глибокому та повному засвоєнню матеріалу. Отже, для системи розвідувального пошуку можливість сортувати матеріал зі складністю була б корисною [17].

Існують різні фактори, які впливають на легкість сприйняття. Наприклад, одним із них є візуальне оформлення тексту. Відомо, що текст, у якому всі літери прописні, сприймається важко. Так само погіршують процес читання декоративні шрифти. Це відбувається тому, що люди під час читання зазвичай впізнають слова за формою, а не читають літери послідовно, а згадані вище стилі спотворюють форму слова [18]. Також важливо, щоб текст мав достатню контрастність із фоном. Якщо їх кольори недостатньо контрастні, це погіршує сприйняття, особливо при яскравому денному освітленні та для людей із вадами зору [19]. Втім, дані фактори не залежать від самого тексту, це лише особливості стилів, які варто враховувати при відображенні матеріалів.

Одним із найбільш суттєвих факторів, які впливають на сприйняття є концентрація специфічних термінів, так зване «лексичне навантаження», - чим воно вище, тим складніше сприймати текст. Іншою важливою складовою є довжина речень: довгі речення складніше сприймати.

Одна із метрик легкість читання була запропоновала Флішем. Відповідно до неї бал читання залежить від таких аспектів як середня довжина речення (розраховується як кількість слів поділена за кількість речень) та середня довжина слів (рахується у складах та дорівнює загальній кількості складів у тексті, поділеній на кількість слів). Формула для розрахунку включає у себе також коефіцієнти, підібрані дослідником.

Існує також і спрощена формула (її спростили Дженкінс та Патерсон). Вона враховує кількість односкладових слів на 100 слів та середню довжину речень (яка вимірюється у словах). Варто зазначити, що спрощена формула більше корелює із розумінням написаних текстів.

Дещо інший підхід було запропоновано Дейлом та Челлом. Їх метод використовує словник із 3000 слів, які були визначені простими. Для того щоб скористатися запропонованим підходом, необхідно вибрати кілька фрагментів тексту по 100 слів, підрахувати середню довжину речення (розділивши кількість слів на кількість речень) та обчислити проценте відношення слів, які не входять до словника. Застосувавши певні коефіцієнти та виконавши прості математичні операції, можна виконувати аналіз складності тексту, результати якого корелюють із ступенем розуміння майже так само, як і покращена формула Фліша.

Ще одну формулу було запропоновано Р. Ганнінгом. Згідно до цієї формули рівень складності розраховується як сума середньої довжини речення помноженої на коефіцієнт 0,4 та відсотку складних слів (у яких більше ніж два склади). Згодом Г. Маклафліном було запропоновано використовувати квадратний корінь із слів, які мають більше, ніж два склади, у фрагменті тексту із тридцяти речень. Цю формулу часто рекомендують використовувати для текстів зі сфери охорони здоров’я.

Існують і такі формули, які використовують тільки кількість складів у словах (наприклад, формула FORCAST). На жаль, використання таких формул обмежений та кореляція із складністю читання є не дуже високою. Втім, вони можуть бути застосовані для текстів, у яких речення є неповними.

Цікавим підходом до оцінки складності є «оцінка синтаксичної щільності Голуба». Цей метод приділяє багато уваги синтаксичному аналізу речення, зокрема, наявності складнопідрядних частин, середній довжині слів головної та підрядної частин речення, кількості модальних дієслів (метод запропоновано для англійської мови) та форм із «be» та «have», кількості прийменників, місцевих, прислівників часу, герундію та часток. Кожний із цих параметрів помножується на певний коефіцієнт. Кінцевий результат порівнюється із табличними даними,

завдяки чому відбувається оцінка тесту. Звісно, такий підхід є більш складним, оскільки потребує синтаксичного аналізу та більшої кількості окремих обчислень. Ще однією проблемою даного підходу є його залежність від мови, якою написаний текст, оскільки часто мови мають різний синтаксис.

Окрім згаданих параметрів, у багатьох інших дослідженнях також пропонувалося використовувати такі параметри як використання активного та пасивного станів, наявність різноманітних вставок у реченні, зв'язність тексту, вік документу, наявність зображень, діаграм та графіків, наявність структури документу та багато іншого.

Варто зазначити, що призначені для оцінки легкості читання формули, дійсно мають високий коефіцієнт кореляції із справжньою легкістю сприйняття, але гарні показники при оцінці за формулами ще означають, що текст справді добре читати. По-перше, зменшення довжини слів та речень не завжди покращує сприйняття, а по-друге, існують й інші характеристики, які формули зазвичай не враховують, наприклад, оформлення та форматування тексту, зацікавленість та рівень освіченості читача та багато іншого. Однак, перераховані вище формули добре зарекомендували себе протягом тривалого періоду часу та довели, що їх використання раціональне використання дійсно дозволяє проаналізувати текст на складність та спростити допомагає його.

3 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ СИСТЕМИ

3.1 Моделювання роботи системи

Схема роботи на рисунок 3.1 відображає основні етапи роботи, а також вхідні та вихідні дані.

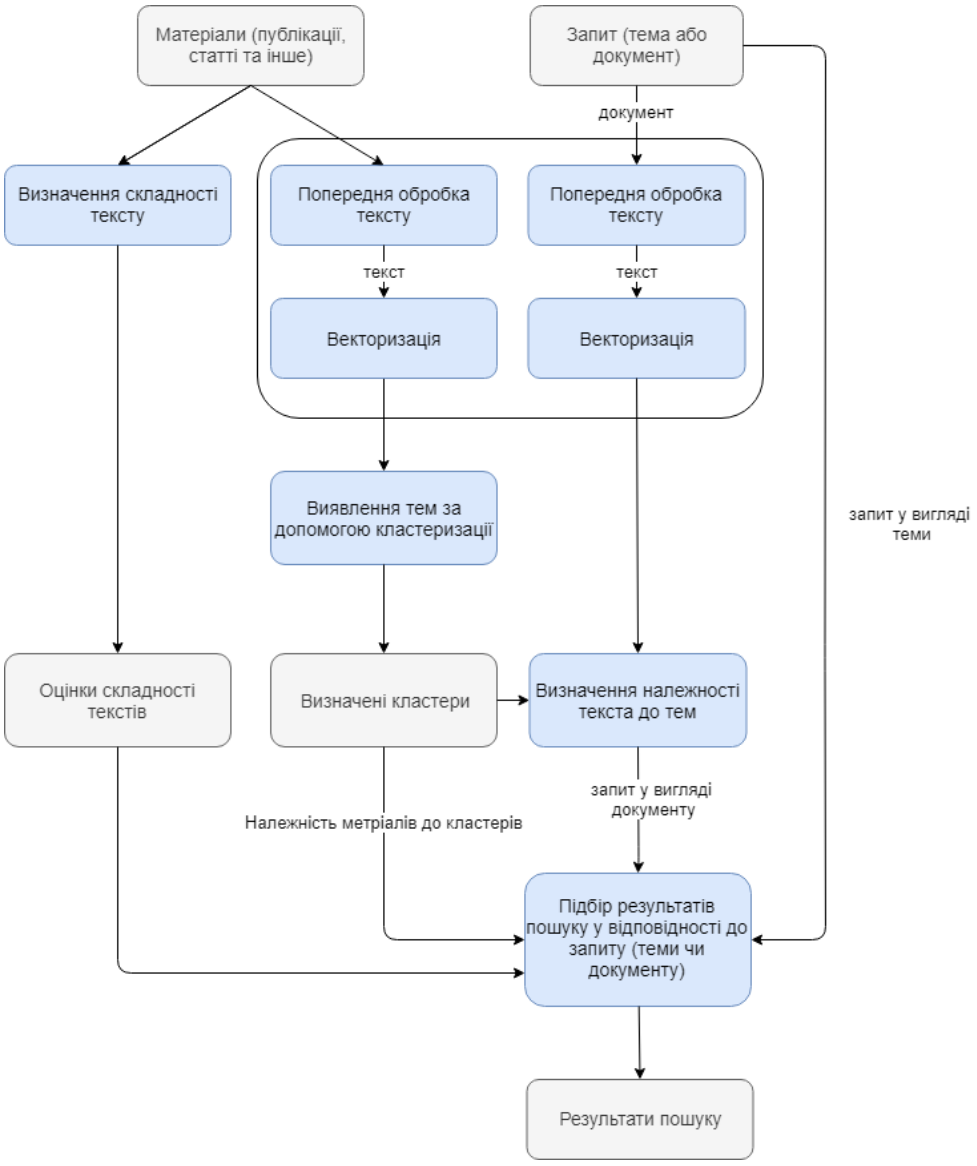

Рисунок 3.1 – Основні етапи роботи системи

Усього є два типи вхідних даних: запит у вигляді теми або документу, матеріали, по яким відбуватиметься пошук.

Перш за все і матеріали для навчання, і розмічені приклади матеріалів, і запит у вигляді документу мають пройти попередню обробку, яка складається із конвертації вхідних даних у зручний формат роботи (із вилученням різноманітного сміття, що може зберігатися у файлах), усунення переносів, видалення стоп-слів та стематизацію чи лематизацію. Після отримання чистих матеріалів необхідно виконати векторизацію. Для даної моделі було обрано формат «мішок слів».

Для опису тематичної моделі будуть використовуватись наступні поняття: D – множина текстових документів (колекція), W – множина усіх термінів, які використовуються (словник).

При побудові тематичної моделі потрібно мати на увазі дві гіпотези: гіпотезу по існування тем та гіпотезу умовної незалежності. Згідно з гіпотезою існування тем, кожне входження терміна w у документ d пов'язано із деякою темою t із заданої скінченної множини T. Колекція документів являє собою послідовність трійок $\Omega_n = \{(w_i, d_i, t_i)|i = 1, \dots, n\}$. Терми w_i та документи d_i є змінними, теми t_i не відомі та є латентними змінними. Гіпотеза умовної незалежності говорить про те, що поява термів у документі d по темі t залежить від теми, але не залежить від документу d та описується загальним для усіх документів розподіленням $p(w|t)$:

$$p(w|d,t) = p(w,t) \tag{1}$$

Відповідно до формули повної вірогідності і гіпотези умовної незалежності, розподілення термів у документі $p(w|d)$ описується імовірнісним розподіленням термів у темах $\phi_{wt} = p(w|t)$ із вагами $\theta_{td} = p(t|d)$:

$$p(w|d) = \sum_{t\in T} p(w|t,d)p(t|d) = \sum_{t\in T} p(w|t)p(t|d) = \sum_{t\in T} \phi_{wt}\theta_{td} \tag{2}$$

Задача тематичного моделювання – обернена задача: по заданій колекції D потрібно знайти параметри ϕ_{wt} та θ_{td}, при яких тематична модель добре наближує частотні оцінки умовних вірогідностей $\hat{p}(w|t) = \frac{n_{dw}}{n_d}$.

У просторі Ω_n вірогідності, які виражаються через змінні d та w, співпадають із частотами подій, які спостерігаються:

$$p(d,w) = \frac{n_{dw}}{n}, \tag{3}$$

$$p(d) = \frac{n_d}{n}, \tag{4}$$

$$p(w) = \frac{n_w}{n}, \tag{5}$$

$$p(w|d) = \frac{n_{dw}}{n_d}, \tag{6}$$

де n_{dw} – число входжень терма w у документ d;

$n_d = \sum_w n_{dw}$ – довжина документа d в термах;

$n_w = \sum_d n_{dw}$ – число входжень терма w у всі документи колекції;

$n = \sum_d \sum_w n_{dw}$ – довжина колекції у термах.

Вірогідності, які пов’язані із прихованою змінною t, також визначаються як частоти:

$$p(t) = \frac{n_t}{n} \tag{7}$$

$$p(w|t) = \frac{n_{wt}}{n_t}, \tag{8}$$

$$p(t|d) = \frac{n_{td}}{n_d}, \tag{9}$$

$$p(t|d,w) = \frac{n_{tdw}}{n_{dw}}, \tag{10}$$

де $n_t = \sum_d \sum_w n_{tdw}$ – довжина колекції у термах;

$n_{td} = \sum_w n_{tdw}$ – число трійок, у яких терм документа d пов’язаний із темою t;

$n_{wt} = \sum_d n_{tdw}$ – число входжень терма w у всі документи колекції;

n_{tdw} – число трійок, у яких терм w документа d пов’язані з темою t.

Знаючи умовні розподіли $p(t|d,w)$, можно оцінити параметри тематичної моделі $\varphi_{wt} = p(w|t)$ та $\theta_{td} = p(t|d)$. Та навпаки, знаючи параметри моделі можна виразити умовні вірогідності $p(t|d,w)$ по формулі Байєса:

$$p(t|d,w) = \frac{p(t,w|d)}{p(w|d)} = \frac{p(w|t)p(t|d)}{p(w|d)} = \frac{\varphi_{wt}\theta_{td}}{\sum_s \varphi_{ws}\theta_{sd}} \tag{11}$$

Таким чином, можна отримати систему нелінійних рівнянь відносно параметрів моделі $\varphi_{wt}, \theta_{td}$ та допоміжних змінних p_{tdw}, n_{wt}, n_{td}:

$$p_{tdw} = \frac{\varphi_{wt}\theta_{td}}{\sum_s \varphi_{ws}\theta_{sd}}, \tag{12}$$

$$n_{wt} = \sum_{d \in D} n_{dw} p_{tdw}, \tag{13}$$

$$\varphi_{wt} = \frac{n_{wt}}{\sum_{w'} n_{w't}}, \tag{14}$$

$$\theta_{td} = \frac{n_{td}}{\sum_{t'} n_{t'd}}, \tag{15}$$

$$n_{td} = \sum_{w \in d} n_{dw} p_{tdw}. \tag{16}$$

Для її рішення зручно застосувати метод простих ітерацій: спочатку обираються початкові наближення параметрів φ_{wt} та θ_{td}, по ним розраховуються допоміжні змінні p_{tdw}, які дають змогу знайти наступні наближення параметрів φ_{wt} та θ_{td}. Такі розрахунки продовжуються у циклі до моменту збігу моделі. У цьому і полягає робота EM алгоритму [1].

Для дослідження методів оцінки складності тексту було обрано три формули: Фліша, Дейла-Челла та Ганнінга. Застосування даних формули буде досліджено у межа атестаційної роботи. Відповідно до результатів дослідження буде обрано найбільш оптимальну.

Головні параметри формули Фліша – сумарна кількість слів у тексті, сума речень, а також загальна кількість складів. Формула має наступний вигляд:

$$complexity = 206.835 - 1.015\left(\frac{totalWords}{totalSentences}\right) - 84.6\frac{totalSyllables}{totalWords} \quad (17)$$

Отримане значення лежить у межах від 0 до 100. Клас складності визначається відповідно до таблиці можливих значень.

Дещо інший підхід має метод Дейла-Челла, оскільки він визначає складні слова не за кількістю складів, а у відповідності до розробленого словника із 3 000 простих слів. Формула Дейла-Челла наступна:

$$complexity = 0.1579\left(\frac{complexWords}{totalWords} * 100\right) + 0.0496\frac{totalWords}{totalSentences} \quad (18)$$

Значення, отримане у результаті розрахунків, лежить у межах від 0 до 10. В залежності від значення, текст відповідно до таблиці відноситься до одного із шести класів складності сприйняття.

Третій метод – метод Ганнінга. Відповідно до методу Ганнінга обирається фрагмент речення у приблизно 100 слів. Ця формула також враховує складні слова – слова, які містять три або більше складів, власні назви, жаргонізми та зіставлені слова не враховуються. Суфікси (-es, -ed, ing) не рахуються як склади. Складність тексту вираховується за наступною формулою:

$$complexity = 0.4(\frac{totalWords}{totalSentences} + 100\frac{complexWords}{totalWords}) \quad (19)$$

Складність тексту в залежності від отриманого значення може бути віднесена до одного із дванадцяти класів.

3.2 Проектування інтерфейсу користувача

Для системи передбачається веб-інтерфейс, для якого було розроблено прототип [20] у Figma. Приклад інтерфейсу переставлений на рисунку 3.2.

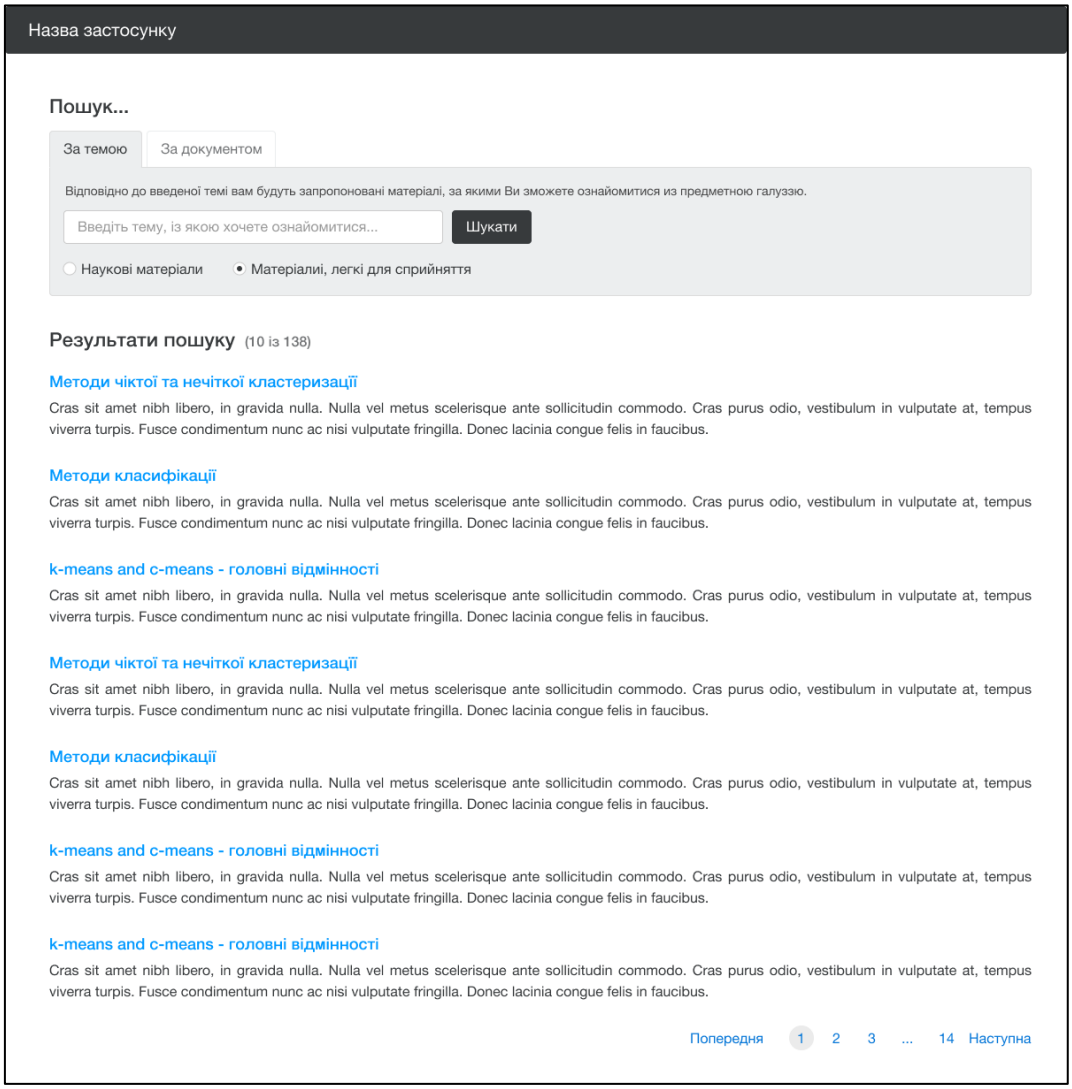

Рисунок 3.2 – Результати пошуку

Зрозуміло, що головний акцент необхідно робити на пошуку. Оскільки планується, що користувач зможе вводити запит як у вигляді теми, так і у якості документу, було вирішено зробити дві вкладинки «За темою» та «За документом».

Незалежно від типу пошуку, користувач отримуватиме короткий опис особливостей того чи іншого результату пошуку. Якщо користувач обере запит у

вигляді теми, він побачить стандартне поле пошуку та кнопку «Шукати». Далі користувач зможе фільтрувати публікації за складністю.

У випадку із пошуком за файлом, користувач побачить кнопку для завантаження файлу та пояснення, як здійснюватиметься пошук (див. рис. 3.3).

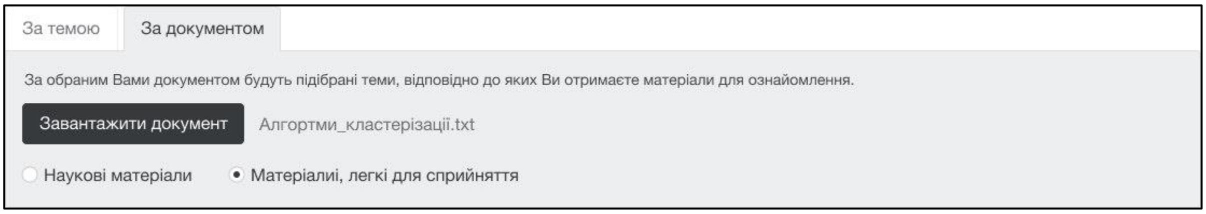

Рисунок 3.3 – Пошук за документом

Результати пошуку будуть представлені у вигляді назви публікації. Якщо користувач перейде за посиланням одного із результатів пошуку, він матиме змогу переглянути більш детальний опис публікації (див. рис. 3.4).

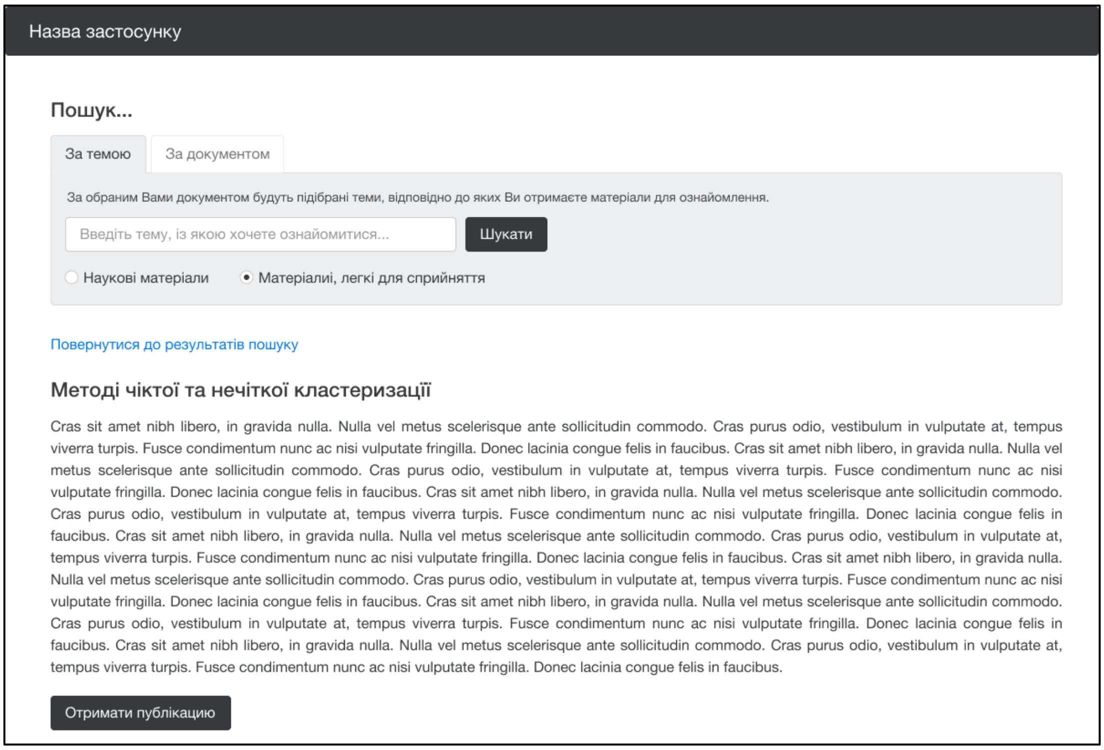

Рисунок 3.4 – Приклад сторінки публікації

Користувач зможе повернутися до результатів пошуку, клікнувши на посилання «Повернутися до результатів пошуку», або знову запустити пошук із цієї

ж сторінки, так як форма із пошуком не буде зникати на сторінці із деталями результату пошуку.

Якщо кількість результатів буде більша десяти, з’являтиметься пагінація, що розбиватиме результати на сторінки [21].

Звісно, даний інтерфейс не є ідеальним, оскільки він спроектований скоріше для демонстрації роботи алгоритму, аніж повноцінної роботи користувача. При вдосконаленні системи планується покращити інтерфейс з точки зору UX та зручності використання.

3.3 Засоби на інструменти для імплементації

У якості мови програмування було обрано Python. Вибір пояснюється великою кількістю бібліотек, інструментів та прикладів.

Що стосується бібліотек та фреймворків, було вирішено використовувати наступні:

- nltk для попередньої обробки текстів природної мови;
- бібліотека textStat класифікатору визначення складності тексту;
- бібліотека Pandas для аналізу та візуалізації даних;
- бібліотека BigARTM для тематичного моделювання [22].

4 АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Дослідження методів розвідувального пошуку складалося із двох частин: підбір оптимальної моделі для тематичного моделювання та вибір кращого підходу для аналізу легкості читання тексту.

4.1 Дослідження тематичного моделювання

Основою розвідувального пошуку є тематичне моделювання. Для його розробки було використано бібліотеку bigARTM. Ця бібліотека не є готовим рішенням, але дозволяє швидко будувати власні моделі за рахунок великої кількості реалізованих інструментів. Однією із основних переваг bigARTM є можливість застосувань регуляризаторів. Найбільш відомою тематичною моделлю, яка часто застосовується, є LDA (латентне розміщення Діріхлє), але вона має дуже багато можливих рішень та дає лише одне із них без можливості обрати найоптимальніший для рішення конкретної задачі. Використання регуляризаторів вирішують цю проблему. Теорія адитивної регуляризація дозволяє розробити власну модель із готових модулів та налаштувати параметри, таким чином, працюючи наче конструктор.

У якості датасету для дослідження було обрано 463 000 статті із Вікіпедії різних тематик. Над датасетом була проведена попередня обробка та видалені стоп-слова, після чого текст був приведений до формату мішка слів.

Використана бібліотека має можливість працювати із модальностями – свого роду категоріями різних типів даних (наприклад, текст, гіперпосилання, різноманітні метадані). Для забезпечення підтримки модальностей використовується формат даних Vowpal Wabbit – великий текстовий документ, де кожний рядок – окремий досліджуваний документ. Якщо у документі є назва, вона

йде спочатку (у вибраному датасеті вона є), після неї йде вертикальна риска, за якою – назва модальності. Потім через пробіли йдуть окремі слова, які були виявлені у документі. Якщо слово зустрілося у документі кілька разів, кількість буде вказана через двокрапку після слова. Матрицю частот слів бібліотека зберігає у вигляді батчів для розбиття колекції на менші частини.

Для тематичної моделі використовується об'єкт ARTM. До його параметрів належать кількість тем, назви тем, а також клас, який включає у себе назву модальності на її вагу. У межах дослідження датасету враховувався тільки одна модальність – текст. У подальшому можна експериментувати із більшою кількістю модальностей, але у цьому разі для більш оптимальної роботи краще налаштовувати модель для кожного датасету окремо. Спочатку було вказано 20 тем, а їх назви розрізнялися лише індексом.

Наступним кроком були встановлені метрики моделі для оцінки якості її роботи. Одною із таких метрик була обрана перплексія (PerplexityScore). Вона допомагає зрозуміти, наскільки добре модель описує дані (чим вона менше, тим краще). Іншими метриками стали розрідженість матриць Θ та ϕ (SparcityPhiScore та SparcityThetaScore). Якщо документ відноситься до невеликої кількості тем Θ буде розрідженою. А якщо слово входить у невелику кількість тем, ϕ буде розрідженою. Чим більша розрідженість, тим краща модель. У якості ще однієї метрики було використано слова, які зустрічаються у темі найбільш часто (TopTokensScore), по якість можна оцінити, як добре теми інтерпретуються. Параметр кількості слів, що будуть виводитися, було встановлено як 10.

Першим кроком роботи моделі є її ініціалізація: бібліотека проходить по усім даним за записує їх у свій словник. У результаті процесу ініціалізації створюються матриці ϕ і Θ, які заповнюються випадковими числами. BigARTM дозволяє у якості параметра ініціалізації встановити значення seed. Було обрано значення 1, щоб кожного разу матриця ініціювалася однаковими значеннями і було краще порівнювати та аналізувати результати роботи.

BigARTM пропонує два режими роботи: застосування EM-алгоритму офлайн та онлайн. Офлайн режим проходить по колекції у кілька ітерацій, проходить по

документу один раз, зберігає матрицю Θ, а матриця ϕ оновлюється після кожного проходу по колекції. Зазвичай такий підхід використовується для обробки відносно невеликих колекцій. Онлайн режим проходить по колекції лише раз, а по документам робить кілька ітерацій, не зберігає матрицю Θ, а матрицю ϕ оновлює через певну кількість опрацьованих документів. Такий підхід рекомендується використовувати при обробці великих колекцій у потоковому режимі. Оскільки датасет планувалося опрацювати не дуже великий датасет (а отже була можливість зберігати матрицю Θ), а також для пришвидшення обробки, для проведення дослідження було обрано офлайн режим (fit_offline). У якості параметрів метод приймає вхідну колекцію, кількість проходжень по колекції (встановлено 40) та кількість проходжень по документу (встановлено 1).

Після проведення деяких експериментів було виявлено, що результати, отримані на частині дасету, практично не відрізняються від отриманих на повному. Тож, було прийнято рішення проводити подальші дослідження лише на частині датасету, щоб зменшити навантаження під час обробки документів.

У результаті моделювання були отримані такі теми (приклади):

- party, political, law, president, court, election, minister, police, union, democratic;
- king, russian, roman, empire, republic, language, india, china, kingdom, emperor;
- army, air, force, military, aircraft, ship, battle, navy, forces, command;
- species, water, food, fish, plant, white, sea, black, areas, region;
- awards, magazine, award, business, show, radio, california, texas, million, america;
- research, institute, science, students, education, technology, india, development, engineering, studies;
- film, episode, show, tv, television, man, character, story, role, movie;
- album, song, band, you, records, songs, chart, track, live, guitar;
- district, population, town, council, municipality, region, polish, center, island, hospital;

– league, club, cup, px, football, stadium, games, division, sports, round.

Як можна помітити, результати тематичного моделювання доволі легко інтерпретуються: по найбільш частотним словам не складно зрозуміти, яка тема була представлена.

На рисунку 4.1 зображений графік збіжності моделі.

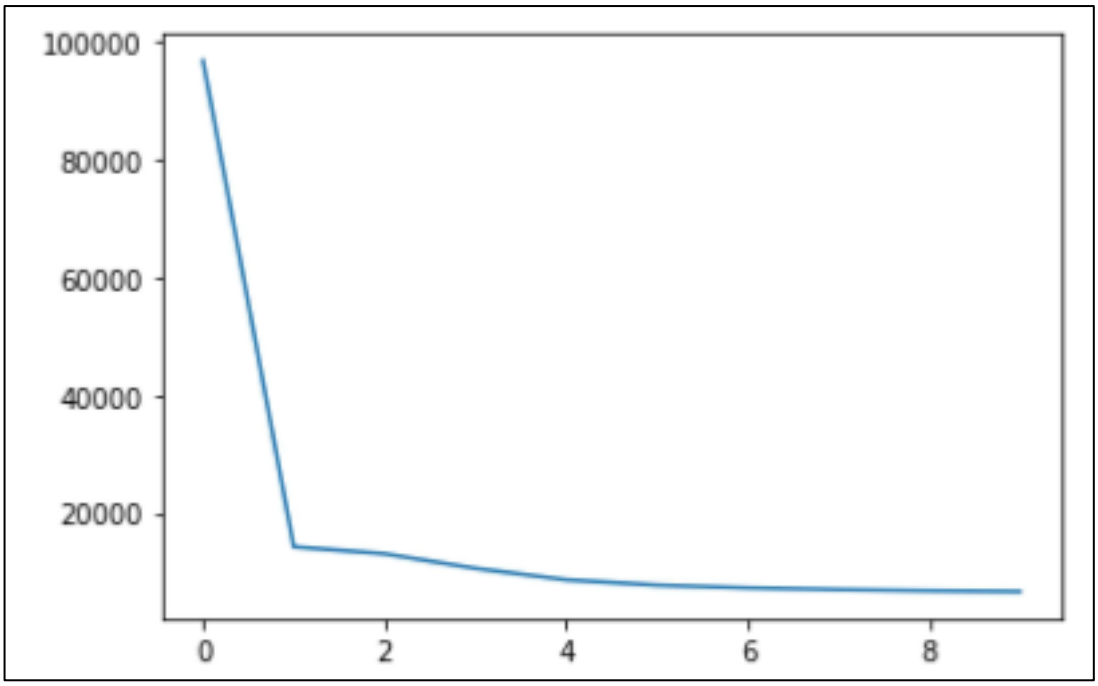

Рисунок 4.1 – Графік збіжності тематичної моделі без додаткових налаштувань

Вертикальній осі відповідають значення перплексії, а горизонтальній – кількість проходів по колекції. На початку роботи значення є випадковими, тому перплексія є високою. Як можна помітити з графіку, після восьмого проходу значення перплексії практично не змінюються, отже у подальших ітераціях не має сенсу.

Значення розрідженості матриці ϕ становило 0.1817848, а Θ – 0.00433088. Таким чином, розрідження є невеликим, особливо у випадку Θ. Якщо б із тексту не були видалені стоп-слова, це могло б спричинити погану інтерпретованість тем через велику концентрацію стоп-слів. Окрім цього, погано розріджені матриці займають багато місця, що погіршує роботу системи.

Приклад матриці ϕ зображений на рисунку 4.2. Рядки відповідають словам, а стовпці – темам. На їх перетині можна знайти вірогідність належності слова певній темі.

	topic_0	topic_1	topic_2	topic_3	topic_4	topic_5	topic_6	topic_7	topic_8	topic_9
akhtar	4.861241e-07	4.598508e-05	2.070681e-10	5.172975e-12	1.065542e-07	2.461598e-11	1.988626e-11	8.356108e-10	1.960576e-07	2.262768e-07
joliette	2.271633e-07	0.000000e+00	0.000000e+00	0.000000e+00	2.965770e-13	0.000000e+00	1.112328e-15	6.857262e-08	0.000000e+00	0.000000e+00
valley	1.392177e-06	2.374326e-04	9.251780e-05	5.533436e-05	3.767371e-04	4.111694e-04	1.117226e-04	2.192018e-03	2.415580e-05	1.949028e-05
cienfuegos	2.865482e-11	6.033656e-07	1.132625e-06	6.540939e-07	6.625693e-11	5.670102e-16	1.350337e-10	2.537574e-11	5.062892e-11	2.126063e-11
throughout	3.341162e-04	1.172879e-04	3.985548e-04	4.328706e-04	3.145070e-04	6.732992e-04	6.900394e-04	4.049320e-04	8.545406e-05	4.501540e-04
spokane	3.887451e-14	9.574509e-13	4.202956e-06	6.034241e-06	2.205139e-06	6.932452e-09	2.152316e-09	1.975569e-04	7.906145e-07	3.281627e-08
besieging	3.757460e-08	3.083088e-05	1.068333e-05	3.781699e-08	4.958097e-10	5.018336e-12	1.765113e-09	1.577631e-07	4.460541e-16	0.000000e+00
reunited	5.683158e-07	1.222489e-05	1.217893e-05	7.425802e-06	1.738911e-05	2.031657e-09	3.143379e-07	3.245167e-06	2.801751e-06	7.464292e-09
mwanza	7.486914e-12	1.066312e-08	1.319813e-15	1.065397e-12	1.026187e-07	0.000000e+00	9.549086e-11	1.755686e-11	2.602808e-06	1.113686e-07
shiba	0.000000e+00	1.089420e-10	3.981169e-11	8.618393e-12	0.000000e+00	2.040008e-06	1.066830e-15	2.073672e-16	0.000000e+00	5.916343e-13
real	1.460133e-04	1.593488e-04	3.038930e-05	1.529887e-03	1.680292e-04	1.814679e-05	5.747275e-04	9.516020e-05	5.328358e-04	5.712449e-05
backstory	2.712432e-12	1.406492e-11	1.230741e-09	1.277413e-10	7.710048e-12	3.127255e-10	1.619158e-10	3.872999e-16	6.825731e-13	4.938472e-15
shipwrecks	3.772677e-10	1.948199e-07	7.658664e-06	4.717799e-08	9.303281e-10	8.270242e-06	1.444080e-07	4.678057e-07	3.941445e-08	1.093436e-06
starters	2.959909e-13	1.415771e-09	4.496403e-07	4.908006e-09	8.232204e-05	1.579805e-05	1.087207e-09	2.085074e-11	4.454746e-06	1.875936e-09
henriques	1.137407e-07	2.298475e-05	2.886239e-08	2.749053e-08	2.113405e-07	2.869869e-10	3.446144e-11	3.643568e-08	8.019620e-06	4.170208e-07
glossary	2.222222e-06	2.618237e-08	1.115580e-05	2.449664e-07	9.229569e-08	2.972413e-05	2.320274e-05	4.460515e-10	2.046003e-09	1.123505e-06
holiness	1.070969e-04	6.690169e-06	1.427370e-12	1.381711e-07	5.254737e-06	8.565081e-11	1.729808e-06	8.061336e-09	4.251093e-14	2.889837e-07
chair	1.218837e-04	4.400459e-07	2.904407e-06	1.613391e-04	1.401719e-05	9.113067e-07	3.200291e-05	3.956229e-04	1.978695e-06	6.392008e-04

Рисунок 4.2 – Приклад матриці ϕ без регуляризатора

Як можна помітити із результатів, модель дійсно створює не дуже розріджену матрицю (є вірогідності 0.000002, 0.000041 та інші).

Схожа ситуація і з матрицею Θ. Практично всі значення вірогідності належності слів темі не є нульовими. Варто зазначити, що розрідженість Θ корелює із розрідженістю ϕ.

Для усунення цієї проблеми було прийнято рішення додати регуляризатор матриці ϕ за допомогою методу SmoothSparsePhiRegularizer. Параметром, який впливає на розрідженість, є tau. Шляхом експериментів для нього було підібрано

значення -8*1e5. Таке значення було найбільш оптимальним з точки зору збільшення розрідженості та можливості інтерпретувати теми.

На рисунку 4.3 розміщена матриця ϕ після застосування регуляризатору.

	topic_0	topic_1	topic_2	topic_3	topic_4	topic_5	topic_6	topic_7	topic_8	topic_9
akhtar	0.000000	5.274959e-05	0.000000e+00	0.000000	0.000003	0.000000	0.000000	0.000000	0.000000	0.000000e+00
joliette	0.000005	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000001	0.000000	0.000000e+00
valley	0.000000	3.555855e-04	7.421022e-05	0.000041	0.000254	0.000734	0.000000	0.003311	0.000000	0.000000e+00
cienfuegos	0.000000	0.000000e+00	6.799211e-07	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00
throughout	0.000280	4.016671e-04	5.153800e-04	0.000661	0.000316	0.000771	0.000615	0.000159	0.000050	5.912869e-04
spokane	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000028	0.000000	0.000000	0.000261	0.000000	0.000000e+00
besieging	0.000000	2.545736e-05	2.785976e-05	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00
reunited	0.000000	1.701789e-05	9.456598e-06	0.000000	0.000028	0.000000	0.000000	0.000000	0.000000	0.000000e+00
mwanza	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000006	2.703170e-08
shiba	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000002	0.000000	0.000000	0.000000	0.000000e+00
real	0.000067	9.815397e-05	0.000000e+00	0.001074	0.000003	0.000000	0.000966	0.000000	0.000563	7.182085e-05
backstory	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00
shipwrecks	0.000000	5.427598e-07	4.413759e-05	0.000000	0.000000	0.000011	0.000000	0.000000	0.000000	0.000000e+00
starters	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000112	0.000010	0.000000	0.000000	0.000000	0.000000e+00
henriques	0.000000	7.520783e-06	0.000000e+00	0.000000	0.000002	0.000000	0.000000	0.000000	0.000005	0.000000e+00
glossary	0.000000	0.000000e+00	4.480218e-06	0.000000	0.000000	0.000027	0.000027	0.000000	0.000000	0.000000e+00
holiness	0.000003	1.181843e-06	0.000000e+00	0.000000	0.000000	0.000000	0.000002	0.000000	0.000000	0.000000e+00
chair	0.000147	0.000000e+00	0.000000e+00	0.000032	0.000032	0.000000	0.000000	0.000234	0.000000	9.647461e-04
recanted	0.000002	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	9.878352e-07
filmworks	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00

Рисунок 4.3 – Приклад матриці ϕ після застосування регуляризатору

Можна помітити, що кількість нулів суттєво збільшилася, що зробило теми більш специфічними. Особливо корисно це було б для датасету із стоп-словами.

Після застосування регуляризатору розрідженість матриці ϕ стала 0.6884208. Завдяки оновленню ϕ збільшилася розрідженість і для матриці Θ – 0.0052103.

Таким чином, можна зробити висновок, що застосування даного регуляризатору суттєво збільшує показники розрідженості обох матриць.

Також потрібно зазначити, що інтерпретованість тем практично не змінилися.

Слід відмітити, що інтерпретуємість тем може змінюватися в залежності від кількості тем, яка служить вхідним параметром при кластеризації. Так як

інтерпретуємість тем є важливим параметром якості тематичної моделі, за допомогою функції TopTokensScore була підрахована когерентність в залежності від кількості тем, після чого був побудований графік когерентності (див. рис. 4.4).

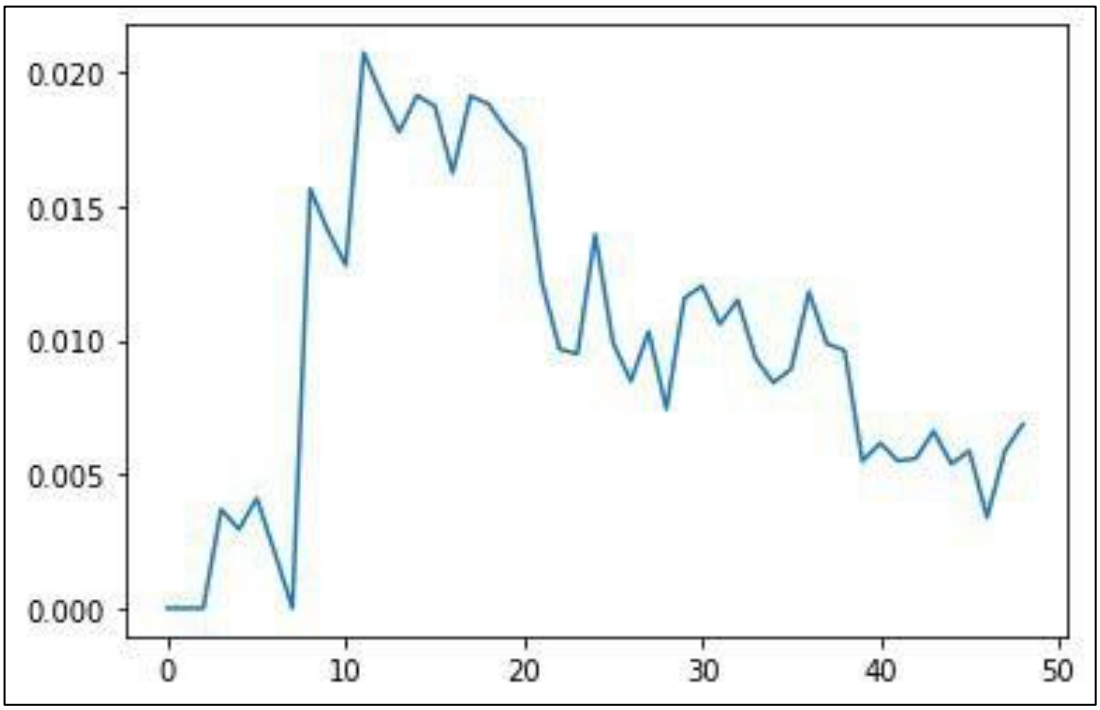

Рисунок 4.4 – Графік когерентності

Вісь Y відповідає значенню когерентності, а X – кількості тем. Із графіку легко можна помітити, що когерентність найбільша при кількості тем приблизно від 12 до 25.

Щоб перевірити дані, що були отримані в результаті аналізу когерентності, а також знаходження більш конкретної кількості тем, було прийнято рішення проаналізувати інтерпретуємість тем власноруч.

При кількості тем 10 розрідженість ϕ становила 0.5211899, а Θ – 0.0020705. Як можна помітити, показники розрідженості не дуже високі. Окрім цього, деякі теми доволі важко піддавалися інтерпретації. Наприклад, «film, album, song, band, show, you, episode, love, tv, television», «air, space, system, aircraft, design, level, program, model, development, different». Аналізуючи слова складається враження, що тема не одна, і через це було не зовсім зрозуміло, до як краще назвати виділені теми.

Кращі показники були отримані при 15 темах: ϕ – 0.6321236, а Θ – 0.0037176. Деякі «змішані» теми, які погано піддавалися інтерпретації, були розділені на дві окремі, наприклад, «album, band, song, you, records, live, chart, songs, video, track» (музична продукція) та «film, episode, show, tv, television, game, man, character, story, role» (кінематограф). Таким чином, можна зробити висновок, що інтерпретуємість тем суттєво покращується при 15 темах.

Ще трохи покращити результати вдалося після збільшення кількості тем до 20. Розрідженість ϕ стала 0.6884208, а Θ – 0.0052103. Але що більш важливо, зросла інтерпретованість тем. Цікаво, що якість тем, які вже добре інтерпретувалися при 15 темах не впала, натомість виокремилися деякі нові та доволі однозначні теми, такі як «system, data, using, systems, software, model, standard, version, space, computer» (комп'ютерні науки), «often, bar, light, color, text, usually, till, type, body, system» (друкарська продукція). Отож, було зроблено висновок, що моделювання для 20 тем працює краще, аніж для 15.

Далі було вирішено збільшити кількість тем до 30. У результаті розрідженість ϕ дорівнювала 0.7539408, а Θ – 0.008963. Таким чином, було підтверджено, що при збільшенні кількості тем зростає розрідженість тем для обох матриць. На жаль, при такій кількості тем погіршилася інтерпретуємість тем. Деякі теми виглядали наступним чином: «social, students, theory, education, level, different, society, example, culture, space» (важко визначитися із темою), «bar, color, text, till, red, often, green, seen, blue, white» (з'явилося багато кольорів, через що зникли деякі ключові слова, за якими можна було визначити теми), «show, radio, women, david, photograph, co, peter, bbc, book, matter» (важко зрозуміти, про що йде мова), «canada, canadian, association, community, program, african, society, hospital, health, medical». Окрім цього, у темах з'явилися такі слова, як «you», «said», які не несуть ніякої інформації щодо теми. Таким чином, було вирішено, що при збільшенні кількості тем до 30, результати погіршуються. Для того, щоб впевнитися у цьому, було прийнято рішення перевірити якість роботи класифікатору на 40 темах. Якщо якість тем продовжує падати – збільшувати кількість тем не варто.

При спробі збільшити кількість тем до 40 розрідженість матриць зросла, але інтерпретуємість стала ще гірше. Таким чином, було вирішено, що 30 тем – забагато, а оптимальне значення лежить у діапазоні від 15 до 30.

Спроба виконати тематичне моделювання для 25 тем виявилася найбільш вдалою. Значення розрідженості були вищими, аніж у випадку із 20 темами (ϕ дорівнювала 0.7230567, а Θ – 0.0072765). Окрім цього, інтерпретованість тем суттєво не знизилася порівняно із 20 темами.

Тож, емпіричним шляхом було визначено, що оптимальна кількість тем для обраного датасету становить 25. Але варто зазначити, що цей параметр потрібно налаштовувати в залежності від датасету. Чим він менший, тим менше тем, ймовірно, знадобиться. У деяких випадках теми, їх кількість та назви взагалі можуть бути відомі завчасно.

Також можна зробити висновок, що інтерпретуємість тем дійсно корелює із когерентністю. Щонайменше, аналіз когерентності дозволяє визначити приблизний діапазон оптимальної кількості тем. Далі краще спробувати кілька значень із діапазону та проаналізувати вручну. При дослідженні такий метод показав, що має сенс розглядати і значення, що розміщені трохи за межами запропонованого діапазону. У випадку із досліджуваним датасетом інтерпретуємість тем суттєво не змінювалася у діапазоні від 17 до 27 тем, тож було прийнято рішення зупинитися на значенні 25, оскільки цей підхід давав більшу кількість тем, а також мав вищі показники розрідженості матриць ϕ та Θ.

Також за допомогою засобів bigARTM було розраховано відношення когерентності та кількості проходів (раніше оптимальна кількість проходів була виявлення за допомогою аналізу перплексії). Графік залежності зображений на рисунку 4.5. Як можна помітити, найвищого значення когерентності можна досягти при 18-20 проходах по колекції. Таким чином, значення когерентності можна покращити при збільшенні кількості проходів. Збільшення проходів по документам не дають такого ефекту. Тож, можна зробити висновок, що варто при визначенні кількості проходів по колекції краще не спиратися лише на дані перплексії, а спробувати визначити оптимальне співвідношення інтерпретуємості теми та

кількості проходів (чим більше проходів, тим повільніше буде працювати алгоритм).

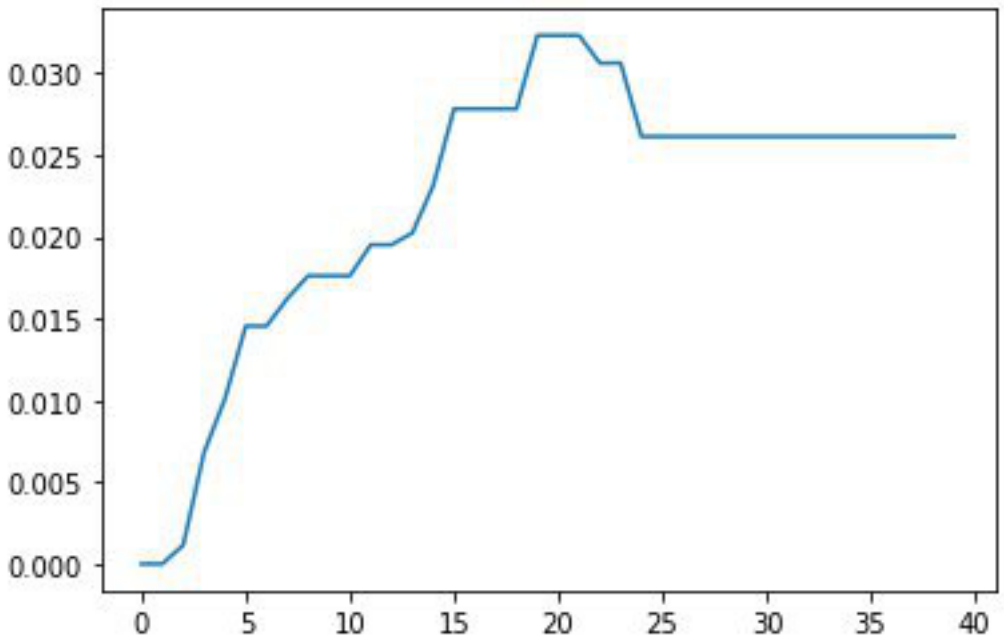

Рисунок 4.5 – Графік залежності когерентності та кількості проходів

У межах роботи дослідження проводилося на датасетів із статей. Але потрібно мати на увазі, що дані за характером контенту можуть дуже відрізнятися за контентом. Наприклад, при роботі із науковими публікаціями скоріше за все буде корисно працювати із кількома модальностями, враховуючи метадані та їх вагу. Ще одним специфічним типом даних – є контент, який згенерований користувачами. Такі дані можуть бути у нагоді для аналізу інтересів користувачів, сплесків обговорення певних тем в залежності від дат та подій, тощо. Варто зазначити, що часто такі тексти мають певні особливості: друкарські та орфографічні помилки, використання скорочень та жаргонізмів.

Для роботи із текстами, що згенеровані користувачами, планувалося застосувати автоматичне виправлення помилок (як друкарський, так і орфографічних). Потенційно, це могло б дещо покращити роботу тематичної моделі, оскільки у мішку слів було б, наприклад, тільки одне слово, а не кілька (наприклад, саме слово та його дві версії з друкарською та орфографічною

помилкою). Під час проведення експериментів, на жаль, готові рішення для виправлення помилок спрацювали не так як очікувалося: слова, який не було у програмі, замінювалися за схожі за написанням, але із зовсім іншим змістом. «Виправлені» слова несли зовсім інші семантичні значення та часто відносилися до інших тем. . Оскільки для тематичного моделювання семантика слів має критичне значення, подібне виправлення слів могло б спричинити спотворення тем та погіршення їх інтерпретованості. Особливо багато проблем виникало з іменованими сутностями. Через це було прийнято рішення не застосовувати автоматичне виправлення помилок для системи розвідувального пошуку у якості попередньої обробки.

Тим не менше, проведене дослідження дало змогу висунути гіпотезу, що хоч застосування існуючих рішень автоматичного виправлення і є небезпечним рішенням для систем розвідувального пошуку, при їх доопрацюванні вони все-таки можуть бути корисними та покращити результати. Зокрема, перед запуском автоматичного виправлення із текстів потрібно вилучити хоча б іменовані сутності та не застосовувати його для них. В ідеалі при роботі із згенерованими користувачами текстами також окремо працювати зі скороченнями та жаргонізмами.

4.2 Дослідження методів оцінки складності тексту

Для дослідження було прийнято рішення обрати такі методи Ганнінга, Дейла-Челла та Фліша. Аналіз роботи методів відбувався із використанням бібліотеки Textstat. Методи Ганнінга та Фліша є схожими, оскільки для розрахунків використовують кількість слів у реченнях та символів у словах. Метод Дейла-Челла базується на використанні словника із 3 000 слів, які було визнано легкими для сприйняття. Дослідження проводилося на частині датасету (10 000 статей), оскільки якість роботи формул не залежить від розміру датасету.

Першим було застосовано метод Фліша. На рисунку 4.6 зображені результати сортування матеріалів, використовуючи метод Фліша.

	0	1	dale_chall	gunning_fog	flesch_kincaid
4980	wikipedia-194230	Why Can't We Be Friends? "Why Can't We Be Fr...	6.99	6.29	4.2
8868	wikipedia-5657238	Maayavi Maayavi is a 2005 Tamil comedy-drama...	6.69	6.83	4.7
7079	wikipedia-3879432	Avacha Bay Avacha Bay () is a Pacific Ocean ...	6.17	8.28	4.9
7442	wikipedia-14978382	Touchdown Club of Columbus The Touchdown Clu...	5.87	6.00	5.0
9420	wikipedia-1222346	Battle Cry of Freedom The "Battle Cry of Fre...	5.82	6.67	5.1
...	...	...	...	...	...
9815	wikipedia-8707922	Anatomical neck of humerus The anatomical ne...	11.78	36.93	33.7
2753	wikipedia-6538673	Bronze Bauhinia Star The Bronze Bauhinia Sta...	11.76	38.02	36.0
2138	wikipedia-39256416	Index of Andhra Pradesh-related articles Thi...	13.10	39.01	40.8
2018	wikipedia-35045680	Meaningful learning Meaningful learning is o...	13.97	48.00	47.0
9732	wikipedia-8614989	Vestibule of the ear Definition. The vestibu...	12.84	52.84	50.9

Рисунок 4.6 – Визначення текстів за складністю відповідно до оцінки Фліша

Для оцінки якості аналізу складності тексту було власноруч прочитано кілька статей із різних частин відсортованої вибірки, особлива увага приділялася першим та останнім текстам. Приклад фрагменту першого тексту: «Maayavi is a 2005 Tamil comedy-drama film directed by D. P. Singapuli that stars Suriya and Jyothika who plays an extended cameo as herself, whilst Vijaykanth and Roja play short cameo roles. The film's score and soundtrack were composed by Devi Sri Prasad. Plot. Abhesh Balaya (Surya), the guide-con man-thief, is footloose and fancy-free. He and his acolyte Sathyaraj (Sathyan) have no hang ups in life. He is also a huge fan of actor Ajith. Balaya's problems start when he and Satyan land up in a huge villa on the beachfront. He is there for making some quick money. It is actually actress Jyothika's house.». Як можна помітити, дійсно, у тексті переважають короткі речення та відносно короткі слова. Тим не менше, не всі слова є дуже простими. Окрім того, у тексті використовуються власні назви, що дещо ускладнюють читання. Ця стаття була не єдиною, яку було дещо важко сприймати, але яка була визначена формулою як легка для сприйняття. Фрагмент одного із найскладніших текстів: «Overseas region Overseas region () is a recent designation given to the overseas departments that have similar powers to those of

the regions of metropolitan France. As integral parts of the French Republic, they are represented in the National Assembly, Senate and Economic and Social Council, elect a Member of the European Parliament (MEP), and use the Euro as their currency. Although these territories have had these political powers since 1982, when France\'s decentralization policy dictated that they be given elected regional councils along with other regional powers, the designation "overseas regions" dates only to the 2003 constitutional change; indeed, the new wording of the constitution aims to give no precedence to either appellation "overseas department" or "overseas region", although the second is still virtually unused by French media.». Читати цей текст і справді важко через велику кількість складнопідрядних та безсполучникових речень, а також через велику кількість довгих, складних слів і власних назв.

Таким чином, можна зробити висновок, що метод Фліша дає доволі непогані результати та його можна застосовувати для сортування результатів пошуку.

Наступною була проаналізована робота оцінки Дейла-Челла. Було зроблено припущення, що цей метод має спрацювати краще, оскільки дозволить враховувати складність слів відповідно до словнику (слова із невеликою кількістю складів не завжди є схожими, і навпаки – не завжди довге слово є складним).

Цікаво, що і за методом Фліша, і за формулою Дейла-Челла, у найскладнішими текстами датасету були визначені одні й ті ж матеріали (але у дещо іншому порядку), а ось найлегші тексти трохи відрізнялися. Приклад фрагмента найлегшого тексту відповідно до сортування за методом Дейла-Челла: «Christian child\'s prayer A Christian child\'s prayer is Christian prayer recited primarily by children that is typically short, rhyming or has a memorable tune. It is said before bedtime, to give thanks for a meal, or as a nursery rhyme. Many of these prayers have a long tradition and come directly from the Bible.». Цей текст був доволі зрозумілим так легко читався, оскільки слова були не складними, а речення – короткими.

Далі було застосовано метод Ганнінга. Найскладнішими текстами були визначені ті ж самі, що і при сортування відповідно до оцінки Фліша, в останніх текстах співпадав навіть порядок (див. рис. 4.7). Найлегші тексти співпадали лише частково.

	0	1	dale_chall	gunning_fog	flesch_kincaid
4825	wikipedia-7784468	Dakota Bowl The Dakota Bowl is the champions...	4.95	4.68	5.8
5426	wikipedia-12924099	Telephone numbers in Vietnam Country code: +...	5.54	5.61	5.3
7132	wikipedia-36583286	Qavi Khan Muhammad Qavi Khan, better known b...	6.34	5.87	5.4
7207	wikipedia-36674872	Novoselki Novoselki () is the name of severa...	5.26	5.93	6.4
7442	wikipedia-14978382	Touchdown Club of Columbus The Touchdown Clu...	5.87	6.00	5.0
...	...	...	...	...	...
9815	wikipedia-8707922	Anatomical neck of humerus The anatomical ne...	11.78	36.93	33.7
2753	wikipedia-6538673	Bronze Bauhinia Star The Bronze Bauhinia Sta...	11.76	38.02	36.0
2138	wikipedia-39256416	Index of Andhra Pradesh-related articles Thi...	13.10	39.01	40.8
2018	wikipedia-35045680	Meaningful learning Meaningful learning is o...	13.97	48.00	47.0
9732	wikipedia-8614989	Vestibule of the ear Definition. The vestibu...	12.84	52.84	50.9

Рисунок 4.7 – Визначення текстів за складністю відповідно до оцінки Ганнінга

Приклад найлегшого тексту: «Qavi Khan Muhammad Qavi Khan, better known by his screen name Qavi Khan, is a Pakistani film and television actor. He has also worked in Radio Pakistan and on stage. He was born on 15 November 1932. Though now largely working in television, he has been in 200 films. Career. Early years, Radio Pakistan and PTV. Khan started at the age of five at Radio Pakistan. After success, he went on to work in Pakistani television in 1964 when the TVs became more common in the country, being one of the first actors of PTV. He started his film career in 1965. Retiring from films and television plays. He retired from films and began supporting actor roles in the various television plays.». Текст справді сприймається легко, речення є короткими та простими, а слова – зрозумілими та здебільшого короткими.

Таким чином, аналіз трьох оцінок показав, що в цілому всі три формули дають схожі результати. Втім, при перечитуванні найлегших текстів тексти, відсортовані за методом Дейла-Челла та Ганнінга сприймалися трохи легше. Однак, це враження не може слугувати повноцінною основою, щоб визнати, що метод Фліша працює гірше: по-перше, для більш-менш однозначної оцінки має бути залучена велика кількість людей; по-друге, на складність сприйняття також впливає освіченість людини. До того ж, оцінювалися тексти іноземною мовою.

Втім, те, що найскладніші тексти були відсортовані всіма трьома методами практично однозначно і їх справді було важко сприймати (як суб'єктивно, так і за відомими показниками), дає змогу припустити, що методи і справді можна використовувати для сортування.

Також у межах дослідження була розрахована кореляція трьох методів (див. рис. 4.8). Високий рівень кореляції був у методу Ганнінга та Фліша (скоріше за все, це пов'язано із тим, що вони використовують однакові параметри для оцінки). Менший показник кореляції був у методу Дейла-Челла по відношенню до двох інших.

	dale_chall	gunning_fog	flesch_kincaid
dale_chall	1.000000	0.669183	0.542440
gunning_fog	0.669183	1.000000	0.906679
flesch_kincaid	0.542440	0.906679	1.000000

Рисунок 4.8 – Кореляція оцінок складності за методом Пірсона

Окрім цього, був підрахований час, за який кожен із методів оцінює 10 000 статей. Виявилося, що метод Фліша відпрацьовує за 7.18 с, Дейла-Челла – за 8.7 с, а метод Ганнінга – за 6.46 с. Тож, різниця у швидкості роботи методів є невеликою.

Таким чином, результати порівняння трьох методів оцінки складності текстів показали, що суттєвої різниці у сортуванні текстів немає. Для даного датасету найлегші тексти за методом Фліша виявилося дещо складнішими для сприйняття, аніж у випадку із двома іншими формулами, але це не є достатньою підставою стверджувати, що він працює гірше. У якості методу для системи було обрано метод Ганнінга, оскільки його результати найбільш корелюють і з формулою Дейла-Челла, із формулою Фліша. Ще однією причиною вибору даного методу є швидкість його роботи. Для оцінки великих датасетів це може бути важливим.

Також при виконанні дослідження було висунуто припущення, якість роботи окремих методів може залежати від мови, якою написані тексти. Особливо

критичною мова скоріше за все буде для методу Дейла-Челла, оскільки його робота базується на використанні спеціального словнику із відповідною для певної мови лексики. Це припущення потребує додаткових досліджень, які не проводилися у межах цієї роботи через необхідність знання мов для оцінки роботи методів.

Окрім цього, під час при аналізі результатів була помічена ще одна особливість, яка впливала на якість оцінки: у деяких текстах, які були визначені складними, були речення із великою кількістю фрагментів, розділених крапкою з комою. Однією із причин, через яку такі тексти були віднесені до складних, є те, що методи визначали такі речення як дуже складні. Насправді виявилося, що ці тексти містили списки, які під час зміни формату були злиті в одне речення. Хоча таких текстів було небагато (і складність речень є не єдиним параметром при оцінки складності), все ж списки спрощують сприйняття, тому можна зробити припущення, що врахування оригінального форматування (щонайменше списків), може дещо вплинути на результати сортування.

Тож, можна зробити висновок, що застосування формул для оцінки легкості текстів для сприйняття, працюють не бездоганно, але можуть бути використані хоча б для приблизного сортування текстів за рівнем складності, так як практично всі тексти, які були визначені складними, містили складні довгі речення та велику кількість доволі специфічних та довгих слів, що погіршують сприйняття.

4.3 Шляхи подальшого розвитку дослідження

Подальші розробки можуть стосуватися як доробок моделі, так і розробки програмної системи. Для покращення тематичного моделювання планується застосувати бібліотеку для виправлення друкарських помилок та помилок правопису для текстів, згенерованих користувачами. Поточне дослідження показало ризикованість використання виправлення помилок через можливу

докорінну зміну слова, тож реалізація виправлення помилок буде потребувати окремої розробки.

Окрім цього, для покращення результатів пошуку можна дати користувачеві виконувати додаткові налаштування для покращення результатів пошуку (наприклад, в особистому кабінеті), а також спробувати враховувати попередні запити для підбору результатів пошуку. Схожий підхід сьогодні вже використовується у системах із пошуком за ключовими словами. Ще одним варіантом для потенційного покращення підбору результатів є виявлення тем, у яких користувач більше зацікавлений, за допомогою аналізу посилань, за якими зрештою переходить користувач, коли отримує результати пошуку.

Для покращення роботи програмної системи та досвіду користувача також добре додати можливість зберігати матеріали, які виявилися корисними, у закладки. Потенційно, ці дані також можна буде використовувати для підбору результатів [23].

5 РОЗРОБКА ПРОТОТИПУ ПРОГРАМНОЇ СИСТЕМИ

5.1 Розробка архітектури та back-end частини

Відповідно до встановлених вимог до програмного забезпечення [24] була розроблена архітектура прототипу програмної системи. Прототип складається з трьох частин: клієнтська частина, сервер та сховища даних [25].

Клієнтська частина являє собою застосунок браузері та взаємодіє з серверною частиною для виконання пошукових запитів. Можливі три варіанти запитів: пошуковий запит за документом, пошуковий запит за переліком тем, запит на список доступних тем. У відповідь на пошукові запити сервер відправляє перелік документів, відсортований за складністю.

Back-end частина слугує для виконання пошукових запитів по корпусу текстів [26] та написана мовою програмування Python [27]. Програмний код відповідає стандартам програмуваня [28].

Сховище міститить наявні матеріали та їх оцінку відповідно до формули оцінки складності текстів Ганнінга. Окрім цього, у сховищі також зберігаються дані, отримані у результаті тематичного моделювання (матриці φ та θ). Сам процес попередньої обробки текстів та машинного навчання (тематичне моделювання) відбувається поза системою і написаний мовою програмування Python [29] та із використанням бібліотеки bigARTM. Одразу після виконання тематичного моделювання відбувається оцінка матеріалів за складністю (щоб не розраховувати її щоразу). Система лише звертається за необхідними даними до блоку із машинним навчанням. Під час отриманння запиту у вигляді теми, відбувається звернення до матриці θ, завдяки чому вилучаються найбільш релеватні матеріали відповідно до теми. Якщо запит був у вигляді документу, відбувається його векторизація (приведення до формату «мішка слів») та за допомогою даних матриці φ визначається розподіл тем у цьому документі. Далі виконуються ті ж кроки, що і у випадку із темами.

5.2 Розробка front-end частини

Розробка розмітки відбувалася за допомогою фреймворку Bootstrap 4. Цей вибір можна тим, що від надає змогу швидко розроблювати інтерфейси, розмітка яких адаптується до різних екранів. Також даний фреймворк має велику кількість готових компонентів, що значно пришвидшує процес розробки. Більшість стилів Bootstrap 4 вже є достатньо зручними та відповідають стандартам доступності, тому більшість з них були просто перевикористані. Решта стилів були перекриті із використанням CSS3.

Розробка макетів відбувалася у відповідності до спроектованого на етапі моделювання інтерфейсу (див. рис. 5.1).

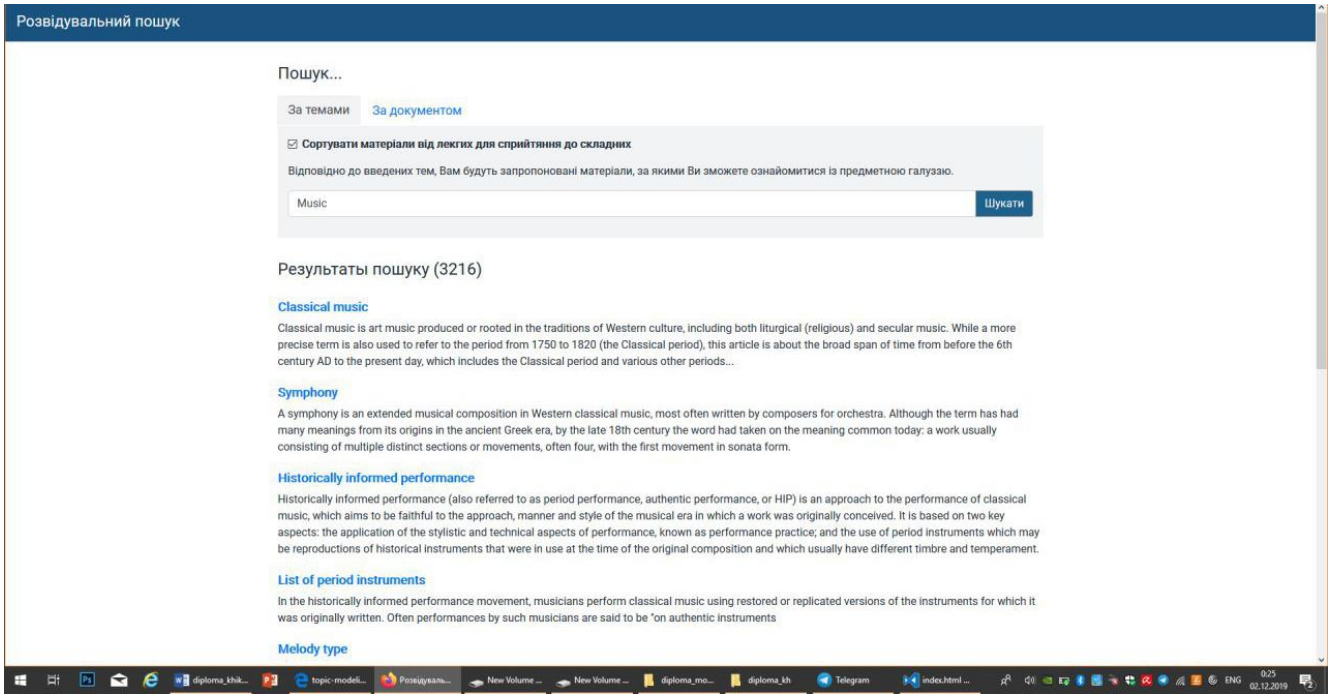

Рисунок 5.1 – Приклад головної сторінки

Головна сторінка включає у себе блок із полями для пошуку, що має два види: пошук за темою та пошук за документом. Переключення між видами відбувається за допомогою табів.

При переключенні видів перезавантаження сторінки не відбувається. Дане перемикання реалізоване завдяки простого приховування елементів за допомогою налаштувань CSS. Це робить перехід менш помітним.

У межах розробленого прототипу також передбачається перехід на сторінку матеріалу: статті, документу, тощо (див. рис. 5.2).

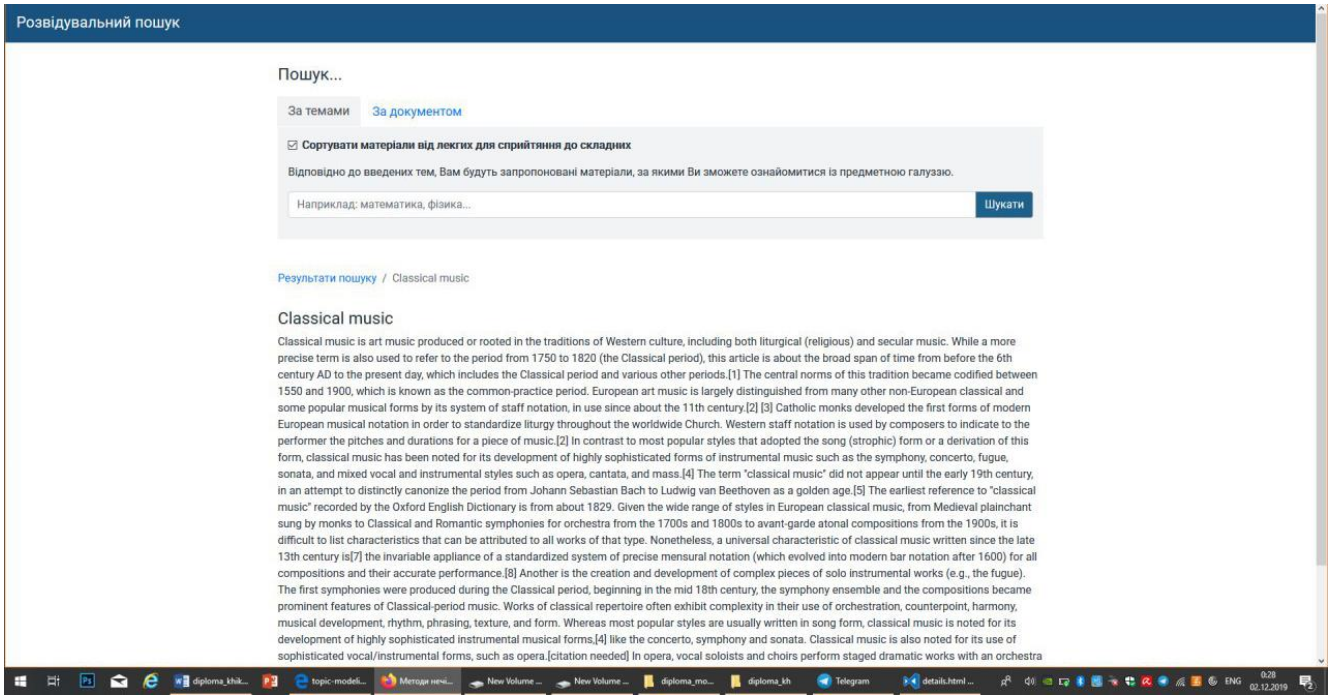

Рисунок 5.2 – Приклад сторінки матеріалу

При переході з’являються хлібні крихти, за якими користувач зможе повернутися до результатів пошуку. На сторінці матеріалу користувач зможе більш детально ознайомитися із контентом. У подальшому очікується додати також мета інформацію щодо матеріалу. При переході за допомогою Ajax перезавантажується тільки частина сторінки (та, що розташована під блоком пошуку), що забезпечує кращий досвід користувача [30].

ВИСНОВКИ

При роботі над атестаційною роботою магістра було проведено аналіз застосування методів семантичного аналізу для пошукових механізмів. Із розглянутих методів було обране тематичне моделювання, яке є основою розвідувального пошуку, а також методи оцінки складності тексту для читання. Ці покращені методи були застосовані у прототипі системи розвідувального пошуку.

У межах виконання атестаційного роботи були виконані наступні завдання:

- проведено аналіз предметної галузі;
- розглянуто застосування методів семантичного аналізу для пошуку;
- розглянуто методи попередньої обробки текстів;
- проаналізовані методи тематичного моделювання та обраний найкращий;
- розроблено математичну модель для задачі розвідувального пошуку;
- підібрано оптимальні параметри, кількість тем та застосовані адитивні регуляризатори для оптимізації роботи системи для підвищення якості виділених тем;
- досліджено вплив кількості проходів по колекції на інтерпретуємість тематичних моделей;
- проаналізовано роботу бібліотек для автоматичного виправлення помилок у текстах;
- порівняно роботу методів оцінки складності текстів для сприйняття та обрано найкращий;
- розроблено прототип розвідувальної системи.

Імплементація прототипу системи за допомогою мови програмування Python та фреймворку Bootstrap 4. Розроблена система підбирає матеріали із бази за введеними темами та за документом, а також може сортувати із за легкістю для сприйняття.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Большакова Е. Автоматическая обработка текстов на естественном языке и анализ данных: учеб. пособие / Е.И. Большакова, К.В. Воронцов, Н.Э. Ефремова, Э. С. Клышинский – М.: Изд-во НИУ ВШЭ, 2017. – 269 с.
2. Кравченко Ю. Семантический поиск в semantic web / Ю. А. Кравченко, В. В. Марков, А. А. Новиков. – УДК 002.53:004.89. – URL: https://cyberleninka.ru/article/v/semanticheskiy-poisk-v-semantic-web (дата звернення: 20.09.19)
3. Morris D. Exploratory Search Technique / E. Horvitz, G. Venolia, M. Morris, R. White. – United States patent: US20070767142. – URL: https://en.patentfield.com/patents/US20070767142#description (дата звернення: 22.09.19)
4. Musetti A. Aemoo: Exploratory Search based on KnowledgePatterns over the Semantic Web / A. Musetti, A. Nuzzoles, F. Draicchio, V. Presutti. – URL: https://pdfs.semanticscholar.org/111a/5a8533a123d3716cbd911fb6f64d025d305a.pdf (дата звернення: 07.10.19)
5. Yue Z. An investigation of search processes in collaborative exploratory web search / Z. Yue, S. Han, D. He. – Proceedings of the American Society for Information Science and Technology. – 2013, №49(1). – DOI: 10.1002/meet.14504901197. – URL: https://asistdl.onlinelibrary.wiley.com/doi/full/10.1002/meet.14504901386 (дата звернення 23.10.2019)
6. O'Connor B. TweetMotif: exploratory search and topic summarization for Twitter / M. Krieger, D. Ahn. – URL: https://www.academia.edu/2821410/Tweetmotif_Exploratory_search_and_topic_summarization_for_twitter (дата звернення: 23.10.19)
7. Barbara M. Subjectivity: Its Role in Exploratory Search Processes and Evaluation / M. Barbara. – URL: https://www.researchgate.net/publication/228891058_

Subjectivity_Its_Role_in_Exploratory_Search_Processes_and_Evaluation (дата звернення: 23.10.19)

8. Hugget M. Evaluating an automatically constructed hypertext for improved searching / M. Hugget, J. Lanir. – URL: http://ryenwhite.com/proceedings/ESI2007.pdf#page=13 (дата звернення: 23.10.19)
9. Falalios P. Exploratory Patent Search with Faceted Search and Configurable Entity Mining / P. Falalios, M. Salampasis, Y. Tzizikas. – URL: https://www.researchgate.net/publication/256839625_Exploratory_Patent_Search_with_Faceted_Search_and_Configurable_Entity_Mining (дата звернення: 14.10.19)
10. Mirizzi R. Semantic Wonder Cloud: Exploratory Search in DBpedia / R. Mirizzi, A. Ragone, T. Noia, E. Sciascio. – DOI: 10.1007/978-3-642-16985-4_13. – URL: https://www.researchgate.net/publication/220940544_Semantic_Wonder_Cloud_Exploratory_Search_in_Dbpedia (дата звернення: 16.10.19)
11. Alonso O. Exploratory search using timelines / O. Alonso, R. Baeza-Yates, M. Gertz. – URL: https://www.researchgate.net/publication/228826308_Exploratory_search_using_timelines (дата звернення: 16.10.19)
12. Ahn J. Semantic annotation based exploratory search for information analysts / J. Ahn, P. Brusilovsky, J. Grady, D. He, R. Florian. – Information Processing & Management. – 2010, №46(4). – DOI: 10.1016/j.ipm.2010.02.001. – URL: https://www.researchgate.net/publication/223449323_Semantic_Annotation_Based_Exploratory_Search_for_Information_Analysts (дата звернення 26.03.2010)
13. Vorontsov K. BigARTM: Open Source Library for Regularized Multimodal Topic Modeling of Large Collections / K. Vorontsov, O. Frei, M. Apishev, P. Romov, M. Durarenko – International Conference on Analysis of Images, Social Networks and Texts Science. – 2015 p. – C. 370-381. – DOI: 10.1007/978-3-319-26123-2_36. – URL: https://www.researchgate.net/publication/300135972_BigARTM_Open_Source_Library_for_Regularized_Multimodal_Topic_Modeling_of_Large_Collections (дата звернення: 17.10.19)
14. Янина А. Мультимодальные тематические модели для разведочного поиска в коллективном блоге / А. О. Янина, К. В. Воронцов. – Интеллектуализация

обработки информации ИОИ. – 2016. – С. 186-187. – DOI: 10.21469/22233792.2.2.04. – URL: http://jmlda.org/papers/doc/2016/no2/Yanina2016Multimodal.pdf (дата звернення: 19.10.2019)

15. Гитис Л. Кластерный анализ в задачах классификации, оптимизации и прогнозирования / Л. Гитис. – М.: Издательство Московского государственного горного университета, 2001. – 104 с.
16. Тараскина А. Нечеткая кластеризация по модифицированному методу с-средних и ее применение для обработки микрочиповых данных / А. С. Тараскина. – URL: https://www.iis.nsk.su/files/articles/sbor_kas_13_taraskina.pdf (дата звернення: 18.10.19)
17. Readibility / Wikipedia, the free encyclopedia. – URL: https://en.wikipedia.org/wiki/Readability (дата звернення: 20.10.19)
18. Loyd J. Typographic Readibility and Legibility / J. Joyd. – URL: https://webdesign.tutsplus.com/articles/typographic-readability-and-legibility--webdesign-12211 (дата звернення: 25.10.2019)
19. Уильямс Р. Дизайн для недизайнеров / Р. Уильямс. – Символ-Плюс, 2016. – 192 с.
20. Варфел Т. Прототипирование. Практическое руководство / Т. Варфел. – М. Манн, Иванов и Фербер, 2013 – 240 с.
21. Раскин Д. Интерфейс: новые направления в проектировании компьютерных систем: пер. с англ. – СПб: Символ-плюс, 2005. – 272 с.
22. Воронцов В. Тематическое моделирование в BigARTM: теория, алгоритмы, приложения / К. В. Воронцов, А. И. Фрей, М. А. Апишев, А. А. Потапенко. – URL: http://www.machinelearning.ru/wiki/images/b/bc/Voron-2015-BigARTM.pdf (дата звернення: 20.10.2019)
23. Бондаренко М., Концепції уніфікації інформаційно-інтелектуальних технологій в системах мовлення / М. Ф. Бондаренко, З. В. Коноплянко, Г. Г. Четвериков. – Бионика интеллекта. – 2011, №1(77). – С. 114-118.
24. Вигерс К. Разработка требований к программному обеспечению. / К. И. Вигерс. – М.: Русская редакция, 2004. – 576 с.

25. Фаулер М. Архитектура корпоративных программных приложений. – М.: Издательский дом "Вильямс", 2006. – 544 с.
26. Захаров А. Архитектура информационно-вычислительных сетей: методические указания / А. С. Захаров; Яросл. гос. ун-т им. П. Г. Демидова. – Ярославль: ЯрГУ, 2013. – 48 с.
27. Лутц М. Изучаем Python, 4-е издание / М. Лутц. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с.
28. Флаулер М. Рефакторинг: улучшение существующего кода. – Пер. с англ. – СПб: Символ-Плюс, 2003. – 432 с.
29. Bird S. Natural Language Processing with Python / S. Bird, E. Klein, E. Loper. – O'Reilly Media, 2009 p. – 482 с.
30. Робинс Д. HTML5, CSS3 и JavaScript. Исчерпывающее руковство / Д. Робинс. – Пер. с англ. – М.: Эксмо, 2014. – 528 с.