ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Методика формування векторної бази знань на основі методів візуалізації багатовимірних даних»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Максим ТОВСТЕНКО
(підпис)

Виконав: здобувач вищої освіти групи ПДМ-61
Максим ТОВСТЕНКО

Керівник: Володимир САДОВЕНКО
к.ф.-м.н., доцент

Рецензент: _____

Київ 2025

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Товстенку Максиму Андрійовичу

1. Тема кваліфікаційної роботи: «Методика формування векторної бази знань на основі методів візуалізації багатовимірних даних»

керівник кваліфікаційної роботи Володимир САДОВЕНКО, к.ф.-м.н., доцент,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «15» жовтня 2024 р. № 320.

2. Строк подання кваліфікаційної роботи «17» грудня 2024 р.

3. Вихідні дані до кваліфікаційної роботи:

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Дослідження ШІ-систем на основі векторних баз даних для створення інтерактивної довідкової.
2. Аналіз методів зменшення розмірності та візуалізацій даних в контексті векторних баз даних.
3. Адаптація методики візуалізації багатовимірних даних для відлагодження ШІ-систем.

5. Перелік ілюстративного матеріалу: *презентація*

1. Мінімальна довідкова ШІ-система.
2. Перелік алгоритмів зменшення розмірності.
3. Початкові умови дослідження для порівняння алгоритмів.
4. Оцінка алгоритмів візуалізації.
5. Математична модель адаптованого методу.
6. Методика формування векторної бази знань.

6. Дата видачі завдання «16» жовтня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	16.10-23.10.2024	
2	Вивчення матеріалів щодо структури сучасних ШІ-систем з векторними базами даних	24.10-28.10.2024	
3	Вивчення алгоритмів зменшення розмірності	29.10-02.11.2024	
4	Вивчення алгоритмів візуалізації векторних баз даних	03.11-07.11.2024	
5	Пошук оптимальних параметрів для адаптації алгоритму зменшення розмірності для візуалізації векторних баз даних	08.11-12.11.2024	
6	Розробка методики формування векторної бази знань на основі методів візуалізації багатовимірних даних	13.11-18.11.2024	
7	Оформлення роботи: вступ, висновки, реферат	19.11-23.11.2024	
8	Розробка демонстраційних матеріалів	24.11-28.11.2024	
9	Попередній захист роботи	29.11-13.12.2024	

Здобувач вищої освіти _____ Максим ТОВСТЕНКО
(підпис)

Керівник
кваліфікаційної роботи _____ Володимир САДОВЕНКО
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 83 стор., 13 рис., 1 таблиця, 50 джерел.

Мета роботи – адаптація методики візуалізації багатовимірних даних для відлагодження інтерактивних довідкових ШІ-систем на основі великих мовних моделей та векторних баз даних.

Об’єкт дослідження – візуалізація багатовимірних даних векторної бази даних.

Предмет дослідження – алгоритм візуалізації багатовимірних даних.

У роботі використано різноманітні методи, такі як:

- Алгоритми зменшення розмірності: t-SNE, UMAP, LargeVis, PCA.
- Методи векторизації текстових даних: використання моделей вбудовування, зокрема bge-multilingual-gemma2.

Проведено аналіз сучасних методів візуалізації багатовимірних даних, що дозволяє виявляти кластери та аномалії. Було відзначено, що алгоритми t-SNE та UMAP є найбільш ефективними для аналізу семантичних зв’язків між векторами, тоді як PCA може використовуватися для базового огляду.

Розроблено та оптимізовано методику формування векторної бази знань, яка включає етапи збору текстових даних, їх перетворення у вектори вбудовування, зберігання у векторній базі даних, а також візуалізацію векторів у двовимірному просторі.

Проведено експерименти для оцінки ефективності різних методів зменшення розмірності, що підтвердило коректність та точність алгоритмів. Застосування адаптованого методу LargeVis дозволило знизити час відлагодження до 30 хвилин.

КЛЮЧОВІ СЛОВА: Великі мовні моделі, векторні бази данних, RAG-системи, алгоритми зменшення розмірності, візуалізація багатовимірних даних.

ABSTRACT

Text part of the master's qualification work: 83 pages, 13 figure, 1 table, 50 sources.

The purpose of the work is to adapt the methodology of multidimensional data visualization for debugging interactive AI help systems based on large language models and vector databases.

The object of research is the visualization of multidimensional data in a vector database.

The subject of research is an algorithm for visualizing multidimensional data.

The work uses a variety of methods, such as:

- Dimensionality reduction algorithms: t-SNE, UMAP, LargeVis, PCA.

- Methods of text data vectorization: use of embedding models, in particular bge-multilingual-gemma2.

An analysis of modern methods of visualizing multidimensional data was carried out, which allows to identify clusters and anomalies. It was noted that the t-SNE and UMAP algorithms are the most effective for analyzing semantic relationships between vectors, while PCA can be used for a basic overview.

A methodology for the formation of a vector knowledge base has been developed and optimized, which includes the stages of collecting textual data, converting them into embedding vectors, storing them in a vector database, and visualizing the vectors in two-dimensional space.

Experiments were conducted to evaluate the effectiveness of various dimensionality reduction methods, which confirmed the correctness and accuracy of the algorithms. The use of the adapted LargeVis method allowed to reduce the debugging time to 30 minutes.

KEYWORDS: Large language models, vector databases, RAG systems, dimensionality reduction algorithms, multidimensional data visualization.

ЗМІСТ

ВСТУП

Сучасні фахівці з обробки природньої мови активно інтегрують великі мовні моделі нейронних мереж в різні ШІ-системи. Одним із різновидів таких систем є інформаційно-посібникова ШІ-системи, в яких кінцевий користувач може задавати питання природньою мовою, в наслідок чого система шукає релевантну інформацію за наданим питанням у своїй базі знань і формулює відповідь. Ці технології вимагають вирішення низки викликів, зокрема, забезпечення точності, релевантності та прозорості згенерованих результатів. У цьому контексті векторні бази даних виступають ключовим інструментом, надаючи ефективні засоби для зберігання, пошуку та аналізу семантично значущих багатовимірних текстових вбудовувань.

Ефективне формування векторних баз даних потребує сучасних методів візуалізації багатовимірних просторів для коректної роботи інтерактивного посібника. Візуалізація дозволяє дослідникам отримати цінне уявлення про структуру даних: виділити кластери, виявити аномалії, оцінити семантичну близькість та зрозуміти природу інтеграції інформації у контекст ВММ. Алгоритми, такі як t-SNE, UMAP та PCA, дають змогу зменшувати розмірність даних, зберігаючи ключові семантичні характеристики, що відкриває нові перспективи для вивчення та оптимізації роботи ВММ.

Актуальність цього дослідження обумовлена необхідністю створення нових методик формування векторних баз знань із використанням алгоритмів зменшення розмірності та візуалізації багатовимірних даних. Такі підходи сприяють побудові адаптивних і прозорих систем, які не лише покращують взаємодію між користувачами та штучним інтелектом, але й мають значний потенціал для застосування у різних сферах — від наукових досліджень до автоматизації бізнес-процесів.

Мета роботи – адаптація методики візуалізації багатовимірних даних для відлагодження інтерактивних довідкових ШІ-систем на основі великих мовних моделей та векторних баз даних.

Об’єкт дослідження – візуалізація багатовимірних даних векторної бази даних.

Предмет дослідження – алгоритм візуалізації багатовимірних даних.

Для досягнення цієї мети були поставлені наступні завдання:

1. Провести аналіз наукової літератури щодо алгоритмів зменшення розмірності та їх застосування у візуалізації векторних баз даних.
2. Адаптувати методику формування векторної бази знань, яка забезпечує інтеграцію з великими мовними моделями.
3. Реалізувати інтерактивний програмний прототип для аналізу і візуалізації багатовимірних даних.
4. Виконати серію вимірювань для оцінки ефективності розробленої методики.

Результати цього дослідження спрямовані на вирішення ключових завдань у галузях аналізу даних, машинного навчання та інформаційних технологій. Запропонована методика дозволяє підвищити точність, ефективність та прозорість інтерактивних інформаційно-допоміжних систем, що створює нові можливості для оптимізації використання великих мовних моделей та підтримки їх розвитку у різних доменах.

1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 RAG-системи

Стаття «Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks» розглядає обмеження великих мовних моделей (LLMs) у виконанні завдань, що потребують точних, актуальних та перевірених даних. Щоб вирішити цю проблему, автори пропонують фреймворк Retrieval-Augmented Generation (RAG), який поєднує параметричну пам'ять (попередньо навчені моделі послідовності до послідовності, наприклад, BART) з непараметричною пам'яттю (щільний векторний індекс, такий як Wikipedia, запитуваний за допомогою нейронних методів пошуку). Цей гібридний підхід покращує адаптивність LLMs, інтегруючи зовнішні системи пошуку для підсилення знань.

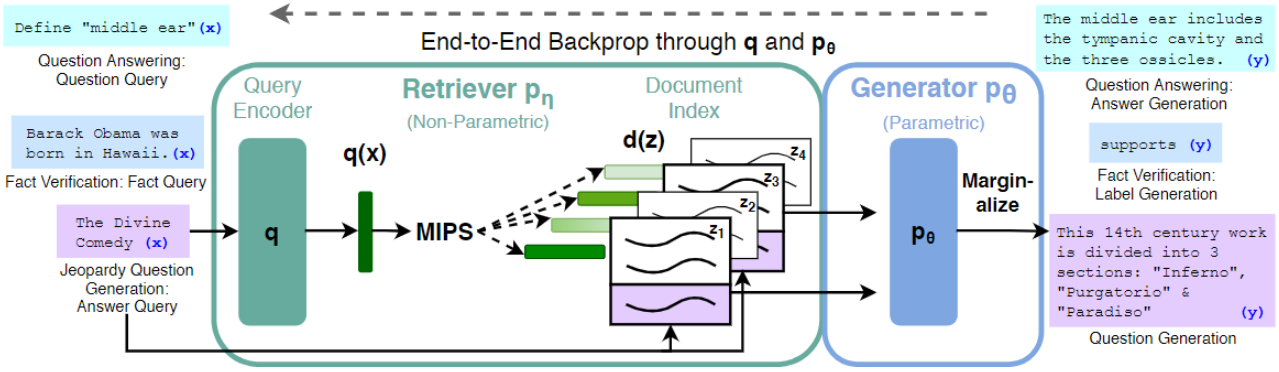


Рис. 1.1 Загальна структура RAG-системи

У статті представлені дві архітектури RAG: RAG-Sequence, де один отриманий документ інформує про весь вихід, і RAG-Token, де різні токени залежать від різних документів. Обидві архітектури демонструють найкращі результати (SOTA) у відкритих запитаннях (QA) і покращують генерацію тексту для завдань, що вимагають інтеграції знань.

RAG-Sequence маргіналізує ймовірності послідовностей по найкраще отриманим документам, покладаючись на один документ для генерації всього

виходу. У контрасті, RAG-Token маргіналізує ймовірності токенів по отриманих документах, дозволяючи різним токенам залежати від різних джерел доказів. Механізм пошуку реалізується за допомогою Dense Passage Retriever (DPR), який використовує бі-кодувальну архітектуру. Кодер запитів і кодер документів перетворюють запити та документи у щільні векторні представлення, а найкращі k релевантних документів витягуються за допомогою Maximum Inner Product Search (MIPS). Для генеративної складової використовується модель BART. Вона поєднує отримані знання з вхідним запитом, конкатенуючи запит і отриманий контент як вхід для моделі кодувальника-декодувальника. Парадигма навчання базується на енд-ту-енд навчанні, розглядаючи отримані документи як латентні змінні. У цьому процесі кодер запитів і генератор доопрацьовуються, тоді як індекс пошуку та кодер документів залишаються статичними. Стратегії декодування відрізняються між двома моделями RAG. RAG-Token генерує виходи, використовуючи пошук з променевою оптимізацією з маргіналізацією на рівні токенів, тоді як RAG-Sequence використовує пошук з променевою оптимізацією для кожного отриманого документа, після чого відбувається ранжування гіпотез.

Ефективність RAG оцінюється за кількома завданнями. У відкритих запитаннях QA використовуються бенчмарки, такі як Natural Questions (NQ), TriviaQA (TQA), WebQuestions (WQ) та CuratedTrec (CT). RAG постійно перевершує базові моделі лише на основі параметрів, такі як T5 та екстрактивні моделі, досягаючи SOTA-метрик. Для абстрактивних QA-завдань на MS-MARCO RAG демонструє вищу фактичну точність і специфічність порівняно з окремими генеративними моделями, такими як BART. Фреймворк також відзначається у створенні нюансованих і фактично обґрунтованих запитань у стилі Jeopardy, що підтверджується людською оцінкою. Крім того, на наборі даних FEVER RAG демонструє конкурентоспроможність у завданнях

верифікації заяв, категоризуючи заяви як «підтримується», «спростовується» або «недостатньо інформації» без специфічного доопрацювання завдання.

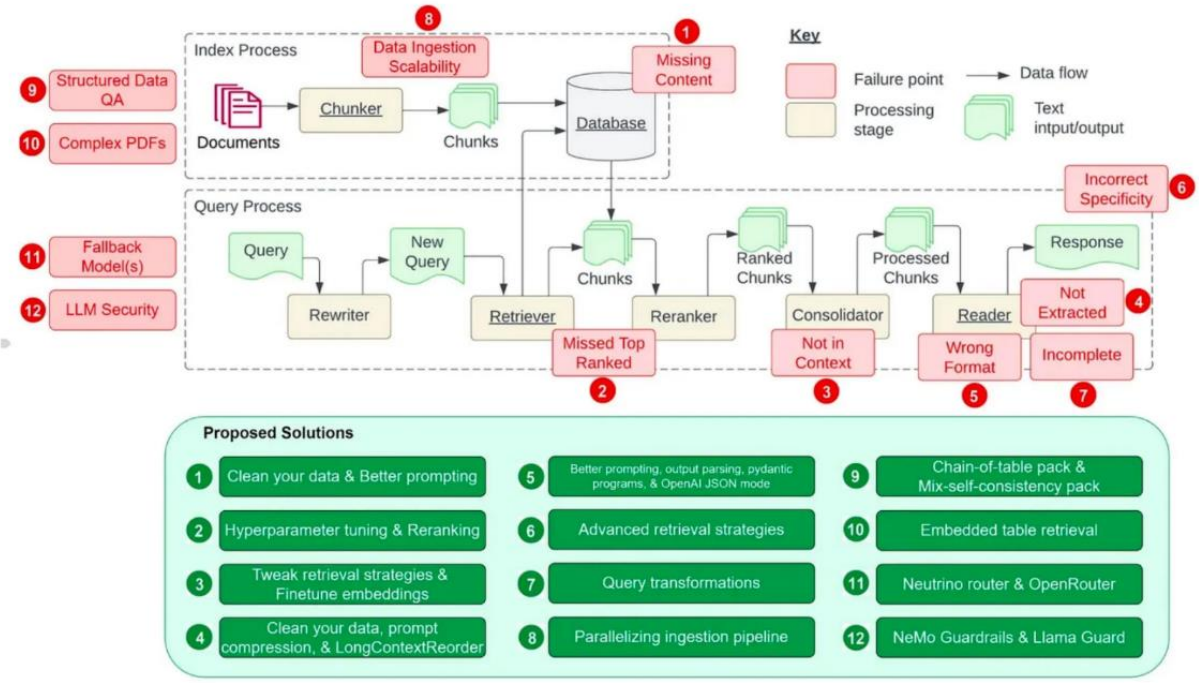


Рис. 1.2 Удосконалена структура RAG-системи

Результати виявляють кілька сильних сторін фреймворку RAG. Гібридна параметрична та непараметрична архітектура ефективно інтегрує зовнішні знання, пропонуючи покращену зрозумілість через отримані документи, які слугують прозорими обґрунтуваннями для згенерованих виходів. Крім того, фреймворк підтримує ефективні оновлення знань, змінюючи індекс документів без повторного навчання. Якість генерації також є сильною стороною, оскільки RAG виробляє різноманітні та обґрунтовані виходи в порівнянні з виключно параметричними системами. Однак, продуктивність значно погіршується, якщо ваги пошукача зафіксовані або якщо DPR замінюється простішими методами пошуку, такими як BM25.

Стаття базується на кількох напрямках досліджень, включаючи фреймворки retrieve-and-read, такі як REALM, DPR та ORQA. Вона також черпає натхнення з мереж, посилених пам’яттю, таких як Memory Networks та Retrieval-

Augmented Transformers, а також з уніфікованих архітектур, на прикладі T5, BART та GPT.

RAG представляє собою модульний і адаптивний фреймворк для знаннєво-інтенсивних завдань, інтегруючи статичні та динамічні представлення знань. Він забезпечує оновлення знань у реальному часі без повторного навчання. Майбутні напрямки включають спільне попереднє навчання пошукачів і генераторів для підвищення їхньої синергії та глибше дослідження взаємодії між параметричною та непараметричною пам'яттю.

Фреймворк RAG має кілька позитивних наслідків. Він підвищує фактичну точність і зрозумілість згенерованих виходів і зменшує ризики галюцинацій, особливо в чутливих сферах, таких як охорона здоров'я та технічна комунікація. Однак існують потенційні ризики, пов'язані з його залежністю від зовнішніх джерел, які можуть поширювати упередження або неточності. Крім того, існує ризик зловживань для генерації оманливого або шкідливого контенту.

1.2 Великі мовні моделі

Великі мовні моделі (ВММ) – це вершина розвитку штучного інтелекту, створена для того, щоб імітувати обробку та генерацію тексту, подібну до людської, з дивовижною точністю. Ці моделі використовують архітектуру трансформаторів, підкріплену механізмами самоконтролю, для моделювання складних патернів і залежностей у текстових даних. Підготовлені на великих корпусах, таких як Common Crawl, який агрегує величезні обсяги інтернет-тексту, і WebText, що складається з високоякісних веб-сторінок, ВММ пропонують всебічне охоплення лінгвістичних стилів, доменів і нюансів. Ця широта сприяє їхній безпрецедентній здатності виконувати цілий ряд завдань, включаючи генерацію тексту, переклад, узагальнення та відповіді на складні

запитання, досягаючи найсучасніших результатів у галузі обробки природної мови (NLP).

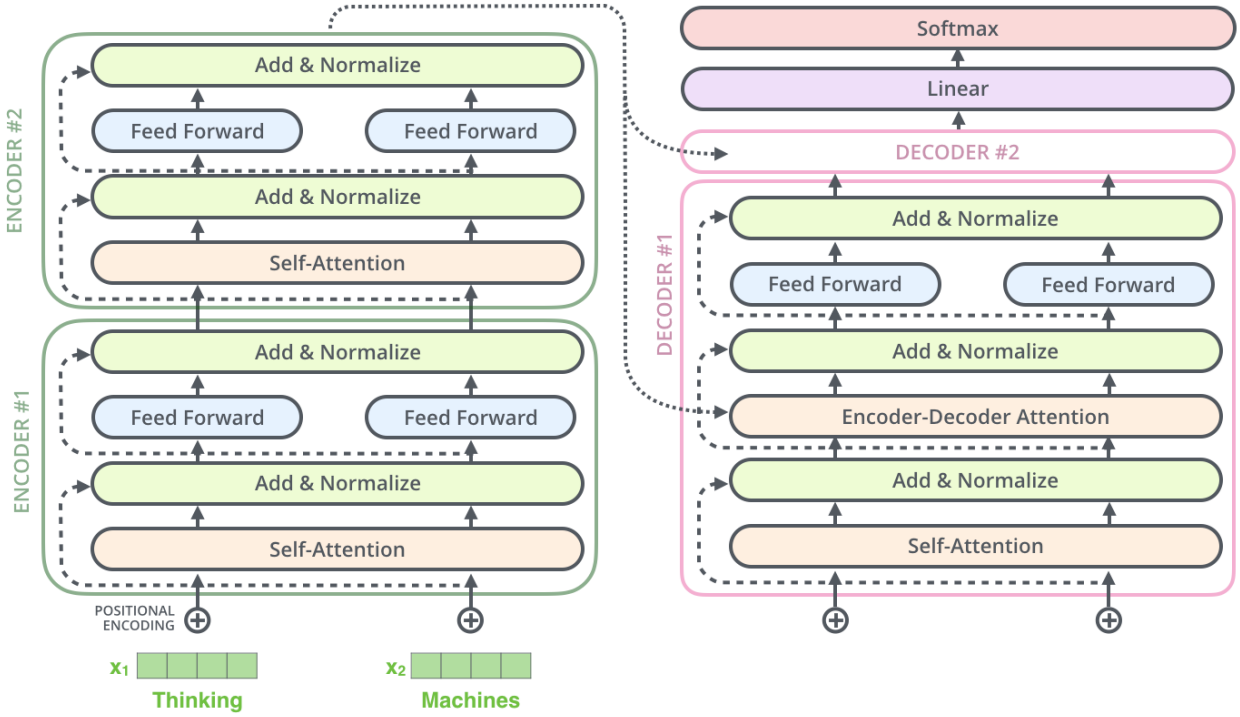


Рис. 1.3 Трансформерна архітектура великих мовних моделей

Архітектура ВММ ґрунтується на трансформаторі – революційному фреймворку, представленому Vaswani et al. (2017) у їхній фундаментальній праці «Attention is All You Need». Цей дизайн використовує механізми самоуваги для моделювання довготривалих залежностей у тексті, таких як підтримання тематичної зв'язності між абзацами або забезпечення синтаксичної та семантичної узгодженості у складних реченнях. Ключові архітектурні компоненти включають шари самоуваги, які розставляють пріоритети і зважують токени на основі їхньої контекстної релевантності; нейронні мережі прямого зв'язку, призначені для вилучення і перетворення ієрархічних ознак; і позиційні кодування, які компенсують притаманний трансформатору брак послідовної обізнаності. Серед відомих реалізацій цієї архітектури – серія GPT від OpenAI, яка встановила еталони в парадигмах навчання з нульовою та малою кількістю спроб; BERT від Google, який вперше застосував контекстні

вбудовування та адаптивність до конкретних завдань; і LlaMA від Meta, який демонструє виняткову ефективність у багатомовних завданнях і використанні ресурсів.

Парадигма навчання для ВММ поділяється на два окремі етапи: попередня підготовка та доопрацювання. Під час попередньої підготовки такі моделі, як GPT, використовують авторегресійну мету, передбачаючи наступний токен у послідовності, методологію, яка за своєю суттю підходить для завдань, що вимагають генеративних здібностей. І навпаки, BERT використовує масковану мету моделювання мови, коли конкретні лексеми замасковані, а модель має передбачити їхню ідентичність, що сприяє надійному розумінню двонаправленого контексту для таких застосувань, як класифікація та міркування. Попереднє навчання проводиться на великих, різнорідних наборах даних, що дозволяє цим моделям кодувати всебічне розуміння лінгвістичних структур, семантичних зв'язків і знань про світ. На наступному етапі попередньо навчена модель адаптується до конкретних доменів або завдань, використовуючи порівняно менші набори даних. Такі методи, як навчання з підкріпленням і зворотним зв'язком з людиною (RLHF), часто застосовуються для того, щоб наблизити результати роботи моделі до людських уподобань та етичних стандартів.

Універсальність ВММ поширюється на широкий спектр сфер. Наприклад, у створенні контенту моделі GPT полегшують створення високоякісних маркетингових текстів, а такі платформи, як Jasper AI, дозволяють блогерам ефективно створювати SEO-оптимізований контент. Системи підтримки клієнтів інтегрують ВММ для покращення взаємодії з чат-ботами, надаючи контекстно-точні та природні відповіді в розмовному стилі. У сфері охорони здоров'я ці моделі допомагають узагальнювати складні медичні записи та покращувати комунікацію між пацієнтом і лікарем. Освітні платформи використовують ВММ для надання персоналізованого навчального досвіду за допомогою

інтелектуальних систем навчання, а в наукових дослідженнях вони дозволяють видобувати інформацію з величезних сховищ неструктурованих текстових даних, прискорюючи відкриття та інновації.

Великі мовні моделі уособлюють трансформаційний стрибок у штучному інтелекті, переосмислюючи те, як обробляються, розуміються і генеруються текстові дані. Їхній глибокий вплив охоплює різні галузі, від автоматизації повсякденних завдань до уможливлення складних аналітичних досліджень. Тим не менш, поточні дослідження повинні вирішувати такі критичні проблеми, як обчислювальна неефективність, етичні проблеми та інтерпретованість моделей, щоб повністю реалізувати їхній потенціал і забезпечити відповідальне застосування в різних суспільних контекстах.

1.3 Моделі вбудовування даних

У багатьох задачах обробки природної мови (NLP) доступно обмежена кількість даних, що ускладнює застосування методів глибокого навчання. Через високу вартість анотування даних, великі набори даних часто недоступні. Для вирішення цієї проблеми використовуються попередньо навчені векторні представлення слів, однак, дослідження показали ефективність попередньо навчених векторних представлень речень для трансферного навчання.

У статті представлено дві моделі для створення таких представлень: на основі трансформера (для високої точності) та глибокої мережі усереднення (DAN) (для ефективного виведення). Обидві моделі, реалізовані в TensorFlow, генерують 512-вимірні вектори, що відображають речення, та можуть використовуватись для оцінки семантичної подібності.

Модель на основі трансформера використовує механізм уваги для створення контекстно-залежних представлень слів, які потім сумуються для отримання векторного представлення речення. Модель DAN усереднює векторні

представлення слів і біграм, а потім пропускає їх через глибоку нейронну мережу. Обидві моделі навчаються з використанням багатозадачного навчання на неконтрольованих даних з веб-джерел (Вікіпедія, новини, форуми) та контрольованих даних з корпусу SNLI.

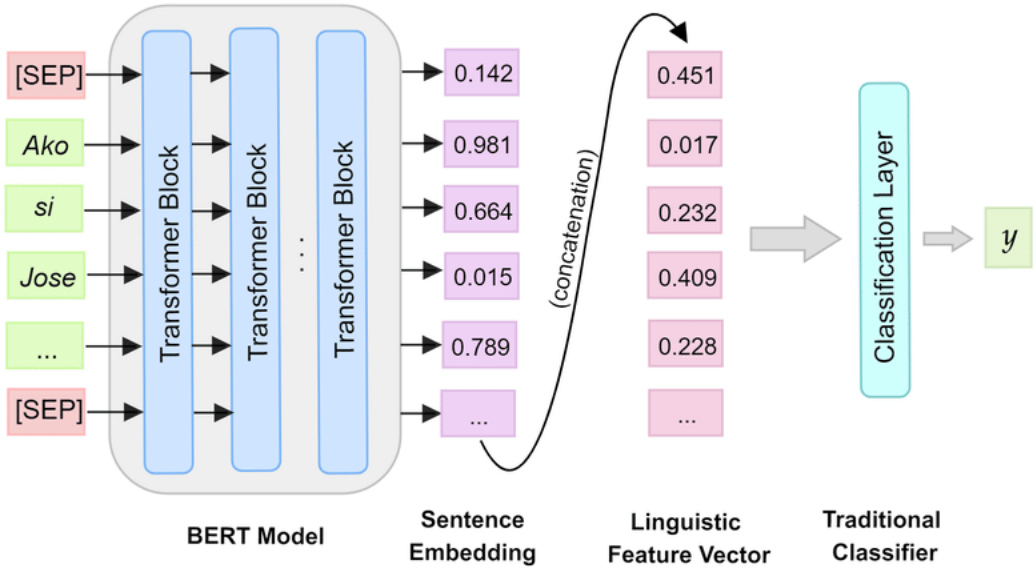


Рис. 1.4 Трансформерна архітектура моделі вбудовування даних (речень)

Для експериментів з трансферним навчанням, вихідні дані кодерів подаються на DNN, специфічну для завдання. Порівнюється продуктивність з моделями, що використовують лише трансфер на рівні слів (word2vec) та без трансферного навчання. Досліджується поєднання трансферу на рівні речень і слів. Гіперпараметри налаштовуються за допомогою Vizier та ручного налаштування. Результати оцінювання усереднюються по десяти запусках з різними випадковими ініціалізаціями.

Трансферне навчання з кодером на основі трансформера зазвичай ефективніше, ніж з кодером DAN, хоча в деяких випадках DAN може працювати так само добре. Моделі з трансфером на рівні речень перевершують моделі з трансфером лише на рівні слів. Найкращі результати досягаються при комбінуванні обох типів трансферу. Трансферне навчання особливо важливе при обмеженій кількості навчальних даних.

Часова складність моделі трансформера – $O(n^2)$, DAN – $O(n)$. Для коротких речень різниця в швидкості невелика, але для довгих речень трансформер значно повільніший. Просторова складність трансформера також $O(n^2)$, тоді як DAN має постійну складність.

Обидві моделі демонструють високу продуктивність трансферу в завданнях NLP, особливо при обмеженій кількості даних. Вони пропонують різні компроміси між точністю та складністю. Попередньо навчені моделі доступні для використання.

1.4 Векторні бази даних

Векторні бази даних відіграють вирішальну роль у сучасних застосунках штучного інтелекту, зокрема у великих мовних моделях та системах семантичного пошуку. Ці системи оперують з багатовимірними векторними вбудовуваннями, що містять закодовану семантичну інформацію. Ця інформація відображається у відносних позиціях векторів у багатовимірному просторі. Візуалізація такого простору є надзвичайно складною задачею. Тому векторні бази даних пропонують оптимізовані механізми зберігання та пошуку, ефективно вирішуючи проблему управління подібністю між векторами. Традиційні бази даних не здатні ефективно обробляти такі дані. Векторні бази даних, такі як Pinecone, забезпечують необхідну продуктивність, масштабованість та гнучкість для роботи з векторними вбудовуваннями. Вони поєднують функціональність традиційних баз даних, включаючи CRUD операції та фільтрацію метаданих, зі спеціалізованими можливостями для обробки векторних даних. Нове покоління векторних баз даних, особливо безсерверні рішення, оптимізують витрати та масштабування, розділяючи вартість зберігання та обчислень. Це дозволяє ефективніше використовувати ресурси та забезпечує гнучкість у масштабуванні залежно від потреб. Таким чином,

векторні бази даних стають незамінним інструментом для сучасних ШІ-застосунків.

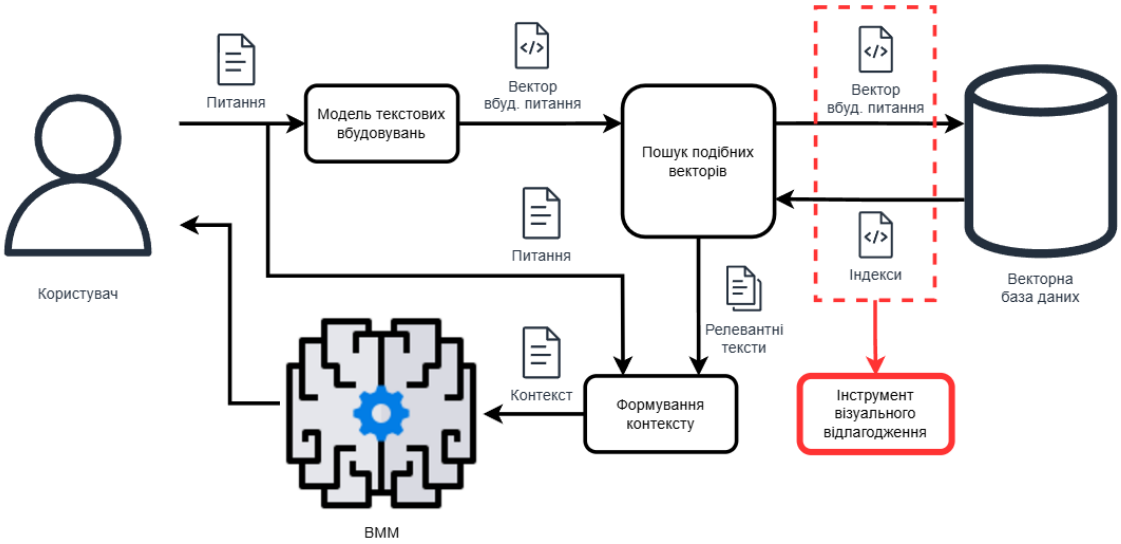


Рис. 1.5 Векторна база даних у контексті інтерактивної довідкової ШІ-системи

Процес роботи з векторною базою даних починається зі створення векторних вставок для індексованого вмісту за допомогою моделі вбудовування. Ці вставки формують багатовимірний векторний простір, де кожен вектор представляє певний елемент вмісту. Згенеровані векторні вбудовування зберігаються у векторній базі даних разом зі зв'язком на оригінальний вміст. Це дозволяє швидко знаходити відповідний вміст за допомогою пошуку по подібним векторам. Коли надходить запит, для нього також генерується вбудовування. Це вбудовування використовується для пошуку найбільш подібних вбудовувань у базі даних. Результати пошуку повертаються користувачеві, забезпечуючи швидкий та релевантний доступ до інформації. На відміну від автономних векторних індексів, таких як FAISS, векторні бази даних пропонують повний набір функцій баз даних. Це робить їх потужнішим інструментом для роботи з векторними даними у складних застосунках штучного інтелекту. Вони забезпечують не тільки зберігання та пошук, але й управління даними, фільтрацію та інші важливі функції.

Векторні бази даних використовують різноманітні алгоритми для оптимізації пошуку найближчих сусідів (ANN). Серед них варто виділити

випадкову проекцію, яка проектує вектори у простір меншої розмірності, зберігаючи при цьому відносну подібність між ними. Квантування продукту (PQ) розбиває вектори на підвектори та створює "кодові книги" для кожного з них, що дозволяє ефективно стискати дані та прискорювати пошук. Локально-чутливе хешування (LSH) хешує подібні вектори в однакові "відра", що також сприяє швидкому пошуку. Ієрархічний навігаційний малий світ (HNSW) створює ієрархічну графову структуру, яка дозволяє швидко знаходити найближчих сусідів у великих наборах даних. Pinecone автоматично керує вибором та налаштуванням цих алгоритмів, звільняючи розробників від необхідності ручної конфігурації. Це значно спрощує процес інтеграції та використання векторних баз даних.

Для порівняння векторів використовуються різні метрики подібності, такі як косинусна подібність, яка вимірює косинус кута між векторами, евклідова відстань, що вимірює відстань між векторами в багатовимірному просторі, та скалярний добуток, який враховує як величини векторів, так і кут між ними. Вибір конкретної метрики залежить від специфіки завдання та типу даних, з якими працює система. Векторні бази даних також дозволяють фільтрувати результати пошуку за метаданими, що зберігаються разом з векторами. Фільтрація може бути попередньою (pre-filtering) або постобробкою (post-filtering), кожна з яких має свої переваги та недоліки. Операції бази даних, такі як вставка, видалення та оновлення даних (CRUD), є стандартними для векторних баз даних. Для забезпечення високої продуктивності та відмовостійкості використовуються такі механізми, як шардування та реплікація даних. Шардування розподіляє дані між кількома вузлами, а реплікація створює копії даних для забезпечення доступності у разі збою.

Для ефективного управління векторною базою даних необхідний моніторинг різних аспектів її роботи. Це включає моніторинг ресурсів, таких як CPU, пам'ять, дисковий простір та мережева активність. Також важливим є

моніторинг продуктивності запитів, включаючи затримку, пропускну здатність та кількість помилок. Загальний стан системи також потребує постійного контролю. Контроль доступу до даних є критично важливим аспектом, що забезпечує захист конфіденційної інформації. Векторні бази даних підтримують резервне копіювання даних для відновлення у разі втрати або пошкодження. Pinecone також надає можливість створювати "колекції" – резервні копії окремих індексів. Для зручної взаємодії з базою даних розробникам надаються API та SDK для різних мов програмування.

Нове покоління векторних баз даних, представлене безсерверними рішеннями, оптимізує витрати та масштабування за рахунок розділення зберігання та обчислень. Вони використовують мультитентність та забезпечують свіжість даних. Ці системи використовують геометричне розділення індексів та шар свіжості для оптимізації пошуку та забезпечення актуальності даних. Безсерверні векторні бази даних дозволяють користувачам зосередитися на розробці застосунків, не турбуючись про управління інфраструктурою. Вони автоматично масштабуються залежно від потреб, що забезпечує гнучкість та економічність. Таким чином, векторні бази даних є потужним та ефективним інструментом для роботи з векторними вбудовуваннями в сучасних застосунках штучного інтелекту. Вони значно перевершують традиційні бази даних та автономні векторні індекси завдяки своїй спеціалізації та розширеним функціональним можливостям.

1.5 Поточні RAG-системи

Бізнес-потенціал сучасних RAG-систем (Retrieval-Augmented Generation – генерація, доповнена пошуком) є значним та багатогранним, охоплюючи різноманітні сфери діяльності. Ці системи, поєднуючи потужність великих мовних моделей (LLM) зі здатністю отримувати релевантну інформацію з

зовнішніх джерел, забезпечують генерацію більш точних, контекстуалізованих та актуальних відповідей. Ключовою перевагою RAG є здатність доступу до інформації в реальному часі, на відміну від LLM, обмежених даними навчання. Це особливо цінно для галузей, де інформація динамічно змінюється, таких як фінанси, новини та наукові дослідження. Завдяки цьому, RAG-системи забезпечують не тільки актуальність інформації, але й її контекстуалізацію, враховуючи специфіку запиту та отримуючи відповідні дані для формування релевантних відповідей. Важливим аспектом є також зниження ймовірності генерації неправдивої інформації (галюцинацій), що є проблемою для деяких LLM, завдяки верифікації інформації через зовнішні джерела.

Застосування RAG-систем сприяє підвищенню ефективності бізнес-процесів. В автоматизації підтримки клієнтів RAG забезпечують швидкі та точні відповіді на запитання, використовуючи бази знань компанії, FAQ та документацію, що оптимізує час очікування та підвищує рівень задоволеності клієнтів. У сфері створення контенту RAG можуть генерувати статті, звіти, описи продуктів та маркетингові матеріали, використовуючи інформацію з різних джерел, що значно заощаджує час та ресурси. Аналіз даних та дослідження з використанням RAG дозволяє обробляти великі обсяги інформації з різних джерел та генерувати звіти, резюме та висновки, що сприяє прийняттю обґрунтованих бізнес-рішень.

RAG-системи відкривають нові можливості для бізнесу, зокрема в персоналізації пропозицій, рекомендацій та контенту на основі інформації про клієнтів, що підвищує рівень залучення та лояльності. Вони також сприяють створенню інтелектуальних пошукових систем, які надають більш точні та контекстно-залежні результати пошуку. Розвиток інноваційних продуктів та послуг на основі аналізу даних, генерації контенту та автоматизації процесів також стає можливим завдяки RAG. Приклади застосування RAG охоплюють

фінансовий аналіз, створення медичних звітів, юридичні консультації та автоматизацію підтримки клієнтів в електронній комерції.

Впровадження RAG також пов'язане з певними викликами. Якість даних, що використовуються для пошуку, є критично важливою для ефективності системи, тому забезпечення достовірності, актуальності та повноти даних є першочерговим завданням. Оптимізація процесу пошуку інформації для забезпечення швидкого та точного доступу до релевантних даних також потребує уваги. Важливим є також врахування етичних питань, пов'язаних з використанням даних, та забезпечення конфіденційності інформації. Отже, RAG-системи демонструють значний потенціал для трансформації бізнес-процесів та створення нових можливостей, поєднуючи потужність LLM з доступом до зовнішніх знань, що робить їх ефективним інструментом для вирішення різноманітних бізнес-завдань.

1.5.1 Vertex AI Search

Vertex AI Search, як складова платформи Vertex AI від Google Cloud, є потужним інструментом для створення пошукових систем корпоративного рівня. Він базується на багаторічному досвіді Google у сфері пошукових технологій. Система інтегрує передові функції, такі як розуміння природної мови (NLU), семантичний пошук та алгоритми ранжування на основі машинного навчання. Ці технології дозволяють системі ефективно обробляти складні та розмовні запити користувачів.

NLU дозволяє системі розпізнавати наміри користувача, навіть якщо запит сформульовано нечітко. Семантичний пошук аналізує значення слів та їхні зв'язки, що забезпечує пошук концептуально релевантної інформації. Алгоритми ранжування на основі штучного інтелекту динамічно оптимізують результати пошуку, враховуючи контекст. Для цього використовуються моделі, навчені на великих обсягах даних.

Vertex AI Search підтримує інтеграцію з різними джерелами даних, включаючи веб-сайти, бази даних та текстові документи. Це забезпечує гнучкість у використанні системи для різних потреб. Завдяки інтеграції з іншими сервісами Vertex AI, такими як Vertex AI Workbench, користувачі можуть розробляти та налаштовувати власні моделі. У контексті RAG (Retrieval Augmented Generation) Vertex AI Search відіграє ключову роль у забезпеченні LLM (великих мовних моделей) необхідною інформацією.

Сильними сторонами Vertex AI Search є використання передової інфраструктури Google, акцент на семантичному розумінні та інтеграція з екосистемою Vertex AI. Це робить його потужним та ефективним інструментом для корпоративних користувачів. Проте, слід враховувати потенційну складність налаштування та конфігурації, а також орієнтацію на корпоративний сегмент. Це може вимагати певних технічних знань.

1.5.2 Azure Cognitive Search

Azure Cognitive Search, розроблений Microsoft на платформі Azure, є хмарним сервісом, що надає пошук на основі штучного інтелекту (ШІ) та тісно інтегрований з іншими компонентами екосистеми Azure, забезпечуючи інструменти для індексування, збагачення та обробки запитів до різноманітних джерел даних. Архітектура сервісу дозволяє гнучко налаштовувати індексування для адаптації до різних форматів та джерел даних, зокрема, користувачі можуть застосовувати власні стратегії індексування та трансформації даних. Ключовою особливістю Azure Cognitive Search є збагачення даних за допомогою ШІ, що автоматично видобуває інформацію з індексованих даних, використовуючи такі методи, як розпізнавання сутностей, видобування ключових фраз, аналіз тональності, оптичне розпізнавання символів (OCR) для обробки зображень, переклад тексту та розпізнавання мови. Цей функціонал значно розширює

можливості пошуку, дозволяючи оперувати не тільки текстовими, але й візуальними та аудіоданими.

Сервіс підтримує розширену мову запитів, що охоплює повнотекстовий, фасетний та геопросторовий пошук, забезпечуючи виконання складних пошукових запитів з високою точністю. Мова запитів Azure Cognitive Search базується на синтаксисі Lucene Query Parser, розширюючи його можливостями фільтрації, сортування, фасетної навігації та геопросторового пошуку. Існують різні типи індексів, оптимізовані для різних сценаріїв використання, включаючи індекси з векторним пошуком для пошуку на основі семантичної близькості. Масштабування сервісу здійснюється шляхом розподілу індексів та збільшення обчислювальних ресурсів, що дозволяє обробляти великі обсяги даних та забезпечувати високу продуктивність. Ціноутворення базується на обсязі зберігання даних, кількості запитів та використанні додаткових функцій ШІ.

Безпека та відповідність стандартам гарантуються інтеграцією з інфраструктурою безпеки Azure, що забезпечує відповідність нормативним вимогам. Тісна інтеграція з екосистемою Azure, зокрема з Azure Blob Storage, Azure Cosmos DB, Azure Machine Learning, Azure Functions та Azure Logic Apps, оптимізує процеси інтеграції та обробки даних, дозволяючи створювати комплексні рішення. Наприклад, Azure Functions можуть використовуватись для попередньої обробки даних перед індексуванням, а Azure Logic Apps – для автоматизації процесів індексації та оновлення даних. У контексті систем Retrieval Augmented Generation (RAG), Azure Cognitive Search виконує функцію пошуку та надає великим мовним моделям (LLM) необхідні контекстні дані, що покращує якість відповідей та зменшує галюцинації моделей. Сервіс знаходить застосування в різноманітних сценаріях, таких як електронна комерція (пошук товарів), управління знаннями (пошук документів та інформації) та інші.

Перевагами сервісу є інтеграція з Azure, потужні можливості збагачення даних за допомогою ШІ та надійна система безпеки. Водночас, орієнтація сервісу

на екосистему Azure може створювати певні обмеження, а ефективне використання Azure Cognitive Search вимагає певного часу на вивчення специфіки роботи з платформою Azure, що необхідно враховувати при виборі відповідного рішення.

1.5.3 Amazon Kendra

Amazon Kendra – це сервіс корпоративного пошуку на основі машинного навчання, який аналізує запити природною мовою та знаходить потрібну інформацію у різних сховищах даних. Архітектура сервісу розроблена з урахуванням простоти використання та інтеграції з існуючими робочими процесами. Kendra використовує NLU для розуміння контексту та намірів користувачів, що підвищує точність пошуку.

Сервіс пропонує велику кількість конекторів для інтеграції з різними джерелами даних, такими як Amazon S3, SharePoint, бази даних та веб-сайти. Це забезпечує доступ до широкого спектру інформації. Користувачі можуть налаштовувати ранжування результатів пошуку відповідно до своїх потреб. Kendra також надає аналітику щодо пошукових запитів користувачів та використання даних.

Безпека та відповідність стандартам забезпечуються завдяки інтеграції з функціями безпеки AWS та відповідним сертифікаціям. Це гарантує захист даних та відповідність вимогам. У архітектурах RAG Kendra виконує роль модуля пошуку, забезпечуючи доступ до різноманітних джерел інформації.

Серед сильних сторін Kendra – простота розгортання та використання, велика кількість конекторів та фокус на NLU та релевантності. Це робить його зручним та ефективним інструментом для пошуку інформації. Проте, варто враховувати орієнтацію на хмарне середовище AWS. Для складних налаштувань може знадобитися технічна експертиза.

1.6 Алгоритми зменшення розмірності

У контексті візуалізації багатовимірних векторів текстових вбудовувань, методи зменшення розмірності відіграють ключову роль, дозволяючи проектувати дані з високовимірного простору у простір з меншою кількістю вимірів, зазвичай 2D або 3D, для подальшої візуалізації. Розглянемо детальніше чотири поширені алгоритми: t-SNE, LargeVis, UMAP та PCA.

t-SNE (t-distributed Stochastic Neighbor Embedding) є нелінійним алгоритмом, розробленим для візуалізації даних високої розмірності. Його основна мета – збереження локальної структури даних. Алгоритм моделює кожну точку даних як точку на карті, намагаючись мінімізувати різницю між ймовірностями подібності пар точок у високовимірному та низьковимірному просторах. Важливим параметром t-SNE є perplexity, який визначає локальну область, що враховується алгоритмом, та впливає на кількість найближчих сусідів, що використовуються при обчисленні ймовірностей. Інші параметри включають n_components (кількість вимірів кінцевого простору), learning_rate (швидкість навчання) та n_iter (максимальна кількість ітерацій). Хоча t-SNE ефективний для даних з високою розмірністю (від десятків до тисяч вимірів), він може бути обчислювально витратним для дуже великих наборів даних.

LargeVis є методом візуалізації великих обсягів даних високої розмірності. Він базується на ідеях t-SNE, але використовує оптимізацію графа для підвищення швидкості та масштабованості. LargeVis будує граф k-найближчих сусідів, а потім оптимізує його для відображення у низькорозмірний простір. Ключовими параметрами є dims (кількість вимірів кінцевого простору), perplexity (аналогічно t-SNE) та knn (кількість найближчих сусідів). LargeVis розроблений для роботи з дуже великими наборами даних, що містять мільйони зразків та високу розмірність.

UMAP (Uniform Manifold Approximation and Projection) – це алгоритм зменшення розмірності, що використовує теорію многовидів та топологію. Він будує граф на основі локальної структури даних та оптимізує його для збереження цієї структури у низькорозмірному просторі. UMAP є швидшим та більш масштабованим, ніж t-SNE, та часто краще зберігає глобальну структуру даних. Важливими параметрами є n_components (кількість вимірів кінцевого простору), n_neighbors (кількість сусідів), min_dist (мінімальна відстань між точками у низькорозмірному просторі) та metric (метрика відстані). UMAP добре працює з даними різної розмірності та з великими наборами даних.

PCA (Principal Component Analysis) є лінійним методом зменшення розмірності, який шукає напрямки (головні компоненти), вздовж яких дані мають найбільшу дисперсію. Він проєктує дані на підпростір меншої розмірності, зберігаючи при цьому якомога більше інформації. Ключовим параметром є n_components (кількість головних компонентів для збереження). PCA є обчислювально ефективним та добре підходить для даних з високою розмірністю, де потрібно зменшити кількість змінних, зберігаючи при цьому глобальну структуру даних.

Порівнюючи ці алгоритми, можна відзначити наступне: t-SNE добре підходить для візуалізації, коли важливе збереження локальної структури, але є обчислювально витратним для великих даних. LargeVis є ефективним для візуалізації дуже великих наборів даних. UMAP є швидкою та масштабованою альтернативою t-SNE, що добре зберігає як локальну, так і глобальну структуру. PCA є швидким лінійним методом, що зберігає глобальну дисперсію даних. Вибір конкретного алгоритму залежить від специфіки задачі, розміру даних та вимог до збереження структури даних.

2 АНАЛІЗ МЕТОДІВ ЗМЕНШЕННЯ РОЗМІРНОСТІ ТА ВІЗУАЛІЗАЦІЙ ДАНИХ В КОНТЕКСТІ ВЕКТОРНИХ БАЗ ДАНИХ

2.1 Аналіз векторного простору вбудовувань речень

Векторні бази даних відіграють важливу роль у зберіганні та пошуку семантично близьких даних, зокрема текстових. Вони дозволяють ефективно оперувати з багатовимірними векторними представленнями, що відображають семантику текстів. Аналіз векторного простору вбудовувань речень є критично важливим для розуміння семантичних відносин між текстами, виявлення кластерів схожих текстів та їхньої візуалізації. Це дозволяє отримати глибше розуміння даних, виявити приховані закономірності та покращити ефективність пошуку. Метою цього підрозділу є дослідження методів аналізу векторного простору вбудовувань речень для ефективної роботи з векторними базами даних.

Існує багато методів та моделей для створення векторних представлень речень. Серед них варто виділити:

- **Sentence-BERT (SBERT):** Модель, що базується на архітектурі BERT та оптимізована для створення якісних вбудовувань речень шляхом використання Siamese та triplet мереж.
- **Universal Sentence Encoder (USE):** Модель від Google, що використовує глибокі згорткові мережі або Transformer для створення вбудовувань.
- **Word2Vec/GloVe з усередненням:** Простіший підхід, що полягає в усередненні векторів окремих слів, отриманих за допомогою Word2Vec або GloVe.
- **OpenAI embeddings:** Вбудовування, що надаються API OpenAI на основі потужних мовних моделей.

- **FastText:** Розширення Word2Vec, що враховує морфологію слів, що особливо корисно для мов зі складною будовою.

Порівняльний аналіз цих методів показує, що SBERT та USE зазвичай демонструють кращу якість вбудовувань, особливо для задач семантичної близькості. Однак, методи на основі усереднення векторів слів можуть бути достатньо ефективними для певних задач та мають меншу обчислювальну складність. Вибір конкретної моделі для подальшого дослідження залежить від специфіки задачі, доступних ресурсів та вимог до точності.

Векторний простір вбудовувань є багатовимірною математичною структурою, що має кілька важливих характеристик, таких як розмірність, щільність і розподіл векторів. Розмірність визначає кількість компонентів у кожному векторі, щільність відображає частку ненульових елементів, а розподіл векторів характеризує їхнє взаємне розташування у просторі. Ці параметри є основними для розуміння властивостей векторного простору, оскільки вони впливають на точність і ефективність аналізу даних, що здійснюється за його допомогою.

Семантична близькість між реченнями або іншими текстовими елементами в цьому просторі відображається через схожість їхніх векторних представлень. Для вимірювання такої схожості використовуються спеціальні метрики, кожна з яких має свої особливості та переваги залежно від поставленої задачі. Зокрема, косинусна відстань дозволяє оцінити кут між векторами. Це особливо зручно для задач семантичної близькості, де важлива подібність у напрямках, а не абсолютна величина векторів. Вона часто є оптимальним вибором у ситуаціях, коли необхідно порівняти об’єкти з різними масштабами.

Окрім косинусної відстані, застосовуються й інші метрики, наприклад, евклідова та манхеттенська. Евклідова відстань вимірює геометричну відстань між точками у просторі, що дає змогу оцінити абсолютну різницю між

векторами. Манхеттенська ж відстань враховує суму модулів різниць між відповідними компонентами, що дозволяє врахувати зміни по окремих вимірах. Проте ці підходи не завжди є ідеальними для задач семантичної схожості, оскільки вони більше орієнтовані на оцінку фізичної відстані, ніж на пошук смислових подібностей.

Вибір метрики є важливим етапом аналізу, оскільки він суттєво впливає на результати, особливо у задачах візуалізації чи кластеризації. Наприклад, у випадках, коли пріоритетом є виявлення саме семантичних зв'язків між текстами, косинусна відстань зазвичай демонструє більш релевантні результати. Натомість евклідова та манхеттенська відстані краще підходять для задач, що потребують точного обчислення абсолютних відстаней. Таким чином, розуміння властивостей і специфіки кожної метрики є ключовим для ефективного аналізу даних у векторному просторі.

Для візуалізації багатовимірних векторних даних використовуються методи зменшення розмірності, що дозволяють відобразити дані у дво- або тривимірному просторі. У роботі будуть розглянуті наступні методи:

- **Principal Componen t Analysis (PCA):** Метод лінійного зменшення розмірності, що знаходить напрямки максимальної дисперсії даних.
- **t-distributed Stochastic Neighbor Embedding (t-SNE):** Нелінійний метод, що добре зберігає локальну структуру даних та ефективний для візуалізації кластерів.
- **Uniform Manifold Approximation and Projection (UMAP):** Сучасний метод, що поєднує переваги t-SNE та зберігає як локальну, так і глобальну структуру даних.
- **Autoencoders:** Нейронні мережі, що використовуються для нелінійного зменшення розмірності шляхом навчання стиснутого представлення даних.

Для візуалізації векторного простору будуть використані 2D/3D scatter plots та інтерактивні графіки. Результати застосування різних методів зменшення розмірності (PCA, t-SNE, UMAP) будуть порівняні на одному й тому ж наборі даних. Аналіз отриманих графіків дозволить виявити кластери, групи схожих речень та аномалії. Інтерпретація результатів буде здійснюватися в контексті досліджуваної задачі. Для відображення додаткової інформації буде використано колір та/або розмір точок. Для візуалізації будуть використані бібліотеки Matplotlib, Seaborn, Plotly та Bokeh. У роботі будуть наведені конкретні приклади візуалізацій з поясненнями та коментарями.

Ефективність різних методів зменшення розмірності буде проаналізовано на конкретному наборі даних. Критеріями оцінки якості візуалізації будуть збереження локальної та глобальної структури даних. Також буде проаналізовано обчислювальну складність різних методів. На основі цього аналізу буде обґрунтовано вибір оптимального методу для конкретної задачі.

Візуалізація векторного простору може бути використана для покращення роботи з векторними базами даних. Зокрема, вона може допомогти зрозуміти структуру даних у базі та оптимізувати параметри індексування, візуально шукати схожі вектори та виявляти помилки в даних або нетипові записи.

2.2 Аналіз методів зменшення розмірності

Методи зменшення розмірності є важливими інструментами для аналізу багатовимірних даних, оскільки вони знижують складність вхідних наборів, зберігаючи найінформативніші характеристики. Такі підходи знаходять застосування в різних галузях, включаючи машинне навчання, обробку даних, економіку, природничі науки та техніку. Вони допомагають покращити аналіз, візуалізацію та інтерпретацію великих обсягів даних.

Методи зменшення розмірності загалом можна класифікувати на ті, що зосереджені на збереженні лінійної структури даних, і на методи, що орієнтуються на нелінійні взаємозв'язки. Перші забезпечують простоту реалізації та інтерпретації, що робить їх придатними для задач, де ознаки даних мають лінійний характер. Другі створені для врахування складної нелінійної структури, що часто виявляється важливим у високовимірних даних, характерних для сучасних наукових і прикладних досліджень.

У багатьох випадках такі методи допомагають не лише зменшити обсяг даних, але й покращити їх якість шляхом видалення шуму та надлишкової інформації. Результати застосування таких підходів стають основою для подальших аналітичних операцій, включаючи класифікацію, кластеризацію чи прогнозування. Вибір оптимального методу залежить від специфіки задачі та структури вхідних даних. Наприклад, якщо необхідно зберегти локальні взаємозв'язки між об'єктами, підходи з акцентом на локальних властивостях виявляються більш ефективними. Водночас, якщо важливе розуміння глобальної структури даних, доцільним є використання методів, що забезпечують збереження загальної конфігурації.

Основними викликами є баланс між збереженням релевантної інформації та зменшенням втрат, а також складність інтерпретації отриманих результатів. Незважаючи на ці виклики, методи зменшення розмірності залишаються важливими інструментами, що сприяють ефективному вирішенню широкого спектра дослідницьких задач.

2.2.1 Аналіз PCA

Аналіз основних компонентів (PCA) представляє фундаментальну техніку зменшення розмірності, яка полегшує перетворення наборів даних великої розмірності в представлення меншої розмірності, зберігаючи при цьому основні

характеристики дисперсії. У цій статті розглядаються теоретичні основи, методологія та практичне застосування PCA в сучасному аналізі даних.

Експоненціальне зростання складності даних викликало необхідність ефективних методів зменшення розмірності. Аналіз основних компонентів стає важливою математичною структурою, яка дозволяє дослідникам зменшувати кількість змінних, зберігаючи цілісність даних. Цей підхід оптимізує компроміс між простотою моделі та точністю, особливо корисним у програмах машинного навчання, де розмірні обмеження можуть значно вплинути на продуктивність моделі та ефективність обчислень.

Рис. 2.1 Результат зменшення розмірності векторів вбудовувань за допомогою алгоритму PCA

Основний принцип PCA обертається навколо визначення оптимальних гіперплощин, які найкраще відповідають розподілу даних. Цей процес оптимізації максимізує дисперсію точкових проекцій на вибрані осі,

забезпечуючи мінімальну втрату інформації під час зменшення розмірності. Геометрична інтерпретація передбачає максимізацію дисперсії точок уздовж вибраних осей при мінімізації середньої квадратичної відстані між вихідними точками даних та їх проекціями в перетвореній системі координат.

Реалізація PCA дотримується структурованого алгоритмічного підходу, починаючи зі стандартизації даних, яка має вирішальне значення через чутливість PCA до масштабу. Цей етап попередньої обробки забезпечує порівнювані масштаби змінних, запобігаючи домінуванню функцій з більшими діапазонами в аналізі та потенційному викривленню результатів. Після стандартизації обчислення коваріаційної матриці забезпечує критичне розуміння зв'язків змінних. Ця матриця виявляє як сильні кореляції, так і надлишковість інформації в наборі даних, формуючи основу для подальшого вилучення компонентів.

Вилучення власних векторів і власних значень із коваріаційної матриці є ключовим кроком у процесі аналізу. Власні вектори визначають напрямки головних компонентів, тоді як відповідні власні значення кількісно визначають дисперсію, пояснену кожним компонентом. Ця декомпозиція дає змогу визначити пріоритети компонентів на основі їх інформаційного вмісту. Згодом формування матриці векторів ознак включає вибрані власні вектори на основі їх значущості. Зменшення розмірності досягається шляхом стратегічного виключення компонентів з мінімальними власними значеннями, оптимізуючи баланс між збереженням даних і структурним спрощенням.

Останній етап PCA передбачає проектування вихідного набору даних на новий простір ознак, визначений основними компонентами. Це перетворення дає представлення зі зменшеною розмірністю, яке зберігає найважливіші характеристики даних. Універсальність PCA поширюється на численні сфери, від біоінформатики до фінансового аналізу. Його застосування виявляється особливо цінним для оптимізації моделі машинного навчання, покращення

візуалізації даних, уточнення кластерного аналізу та покращення розпізнавання образів.

Аналіз основних компонентів є незамінним інструментом у сучасному аналізі даних. Його системний підхід до зменшення розмірності в поєднанні з надійною математичною основою робить його важливою технікою для дослідників і практиків, які працюють з масивами даних великої розмірності. Здатність методу зберігати цілісність даних, одночасно зменшуючи складність, продовжує спонукати до його впровадження в різних наукових дисциплінах. Завдяки його впровадженню дослідники можуть ефективно керувати широкомасштабним аналізом даних, зберігаючи статистичну значущість та можливість інтерпретації своїх результатів.

Значення PCA в сучасній науці про дані неможливо переоцінити, оскільки воно забезпечує математично обґрунтовану основу для вирішення завдань аналізу даних великої розмірності. Його незмінна актуальність у методології дослідження підкреслює важливість методів розмірного зменшення в сучасних наукових дослідженнях. Оскільки складність даних продовжує зростати в різних областях, роль PCA як фундаментального аналітичного інструменту залишається надзвичайно важливою для вдосконалення нашого розуміння складних наборів даних і сприяння значущим науковим відкриттям.

2.2.2 Аналіз t-SNE

t-SNE (t-розподілене стохастичне вкладення сусідів) є складною та новаторською технікою зменшення розмірності, яка значно змінила підхід до аналізу багатовимірних даних у різних наукових і обчислювальних галузях. Розроблена як інноваційний ймовірнісний алгоритмічний метод, t-SNE дозволяє дослідникам і аналітикам даних елегантно перетворювати складні багатовимірні набори даних у просторові представлення меншої розмірності, зберігаючи при

цьому внутрішні топологічні та структурні особливості, що визначають основні геометричні відносини вихідних даних.

Основний механізм t-SNE базується на витонченій ймовірнісній стратегії відображення, яка систематично перетворює багатовимірні евклідові відстані в умовні ймовірності, що представляють схожість між точками. Використовуючи розподіл Стьюдента з одним ступенем свободи у просторі малої розмірності, алгоритм стратегічно долає проблему скупчення точок, яка раніше була характерною для традиційних лінійних методів зменшення розмірності.

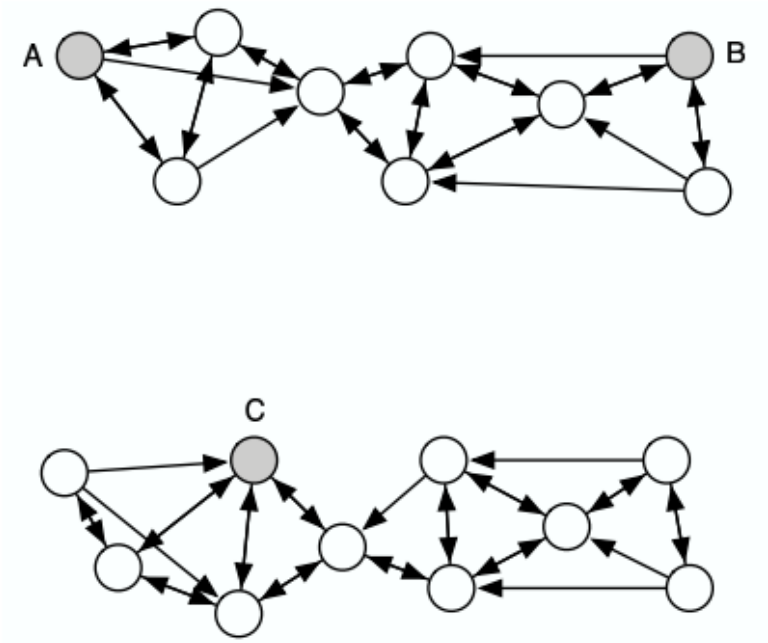


Рис. 2.2 Версія t-SNE на основі випадкового блукання краще відображає структуру даних, ніж стандартні орієнтирні підходи, враховуючи зв'язність графа, а не лише відстані.

На відміну від класичних технік, таких як метод головних компонент (PCA) і багатовимірне шкалування (MDS), які часто стикаються з труднощами при роботі зі складними нелінійними структурами даних, t-SNE демонструє вражаючу здатність обробляти такі конфігурації. Основна перевага цього алгоритму полягає у його здатності зберігати локальні взаємини між точками, водночас відкриваючи глобальні структурні особливості, наприклад, приховані

кластери даних, які могли залишатися непоміченими за допомогою традиційних методів.

Обчислювальний процес t-SNE є складною математичною процедурою, яка передбачає точне обчислення умовних ймовірностей, що відображають ймовірність того, що точки даних є сусідами як у багатовимірному просторі, так і в просторі меншої розмірності. Мінімізуючи розходження Кульбака-Лейблера між цими розподілами ймовірностей, t-SNE створює просторове представлення, яке відтворює геометричні й топологічні властивості вихідних даних із високою точністю.

Стохастичний характер процесу вкладення, який ґрунтується на принципах випадкових блукань, дозволяє t-SNE генерувати візуалізації, що забезпечують більш інтерпретовані та змістовні інсайти, ніж традиційні лінійні методи. Ця особливість особливо цінна в сучасних галузях, таких як машинне навчання, обчислювальна біологія, нейронаука, геноміка та дослідження даних, де складні багатовимірні набори даних вимагають удосконалених стратегій зменшення розмірності.

Однією з найбільш привабливих характеристик t-SNE є його гнучкість і адаптивність у різних наукових і обчислювальних контекстах. Дослідники можуть використовувати цю техніку для виявлення прихованих патернів, дослідження складних ландшафтів даних і створення візуальних представлень багатовимірної інформації, які були б недосяжними за допомогою традиційних методів аналізу.

Однак важливо підходити до використання t-SNE з розумінням його методологічних нюансів. Хоча алгоритм надзвичайно точно зберігає локальні структури даних, інтерпретація глобальних взаємозв'язків вимагає обережності. Стохастична оптимізація в t-SNE може вводити варіації між різними ітераціями, тому необхідний виважений підхід до інтерпретації результатів.

Такі параметри алгоритму, як perplexity (складність) і learning rate (швидкість навчання), відіграють ключову роль у визначенні якості та інтерпретованості отриманого представлення. Ретельне налаштування параметрів і емпірична перевірка є важливими для того, щоб візуалізація відображала основні структурні характеристики даних.

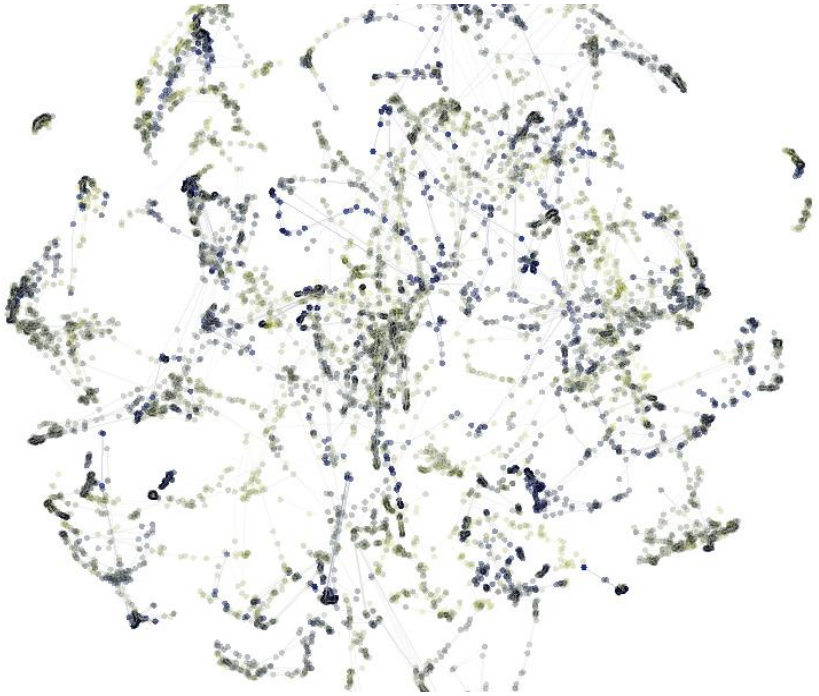


Рис. 2.3 Результат зменшення розмірності
векторів вбудовувань за допомогою алгоритму t-SNE

З огляду на стрімкий розвиток обчислювальних методів і зростаючу популярність багатовимірних наборів даних у різноманітних наукових дисциплінах, алгоритм t-SNE залишається прикладом ефективності інноваційних підходів для аналізу складних структур даних. Його здатність поєднувати абстрактні математичні моделі з інтуїтивно зрозумілими візуалізаціями робить t-SNE ключовим інструментом сучасного аналітичного арсеналу.

Для низьковимірних точок, що відповідають високовимірним даним, обчислюється умовна ймовірність за аналогією до високовимірного простору. У алгоритмі t-SNE використовується t-розподіл Стьюдента з одним ступенем

свободи для оцінки подібності в низьковимірному просторі. Якщо точки візуалізації адекватно моделюють подібність між точками високовимірного простору, умовні ймовірності для обох просторів будуть близькими. Алгоритм t-SNE спрямований на знаходження такого низьковимірного представлення даних, яке мінімізує розбіжність між цими ймовірностями. Для досягнення цієї мети використовується міра розбіжності Кульбака-Лейблера.

Мінімізація розбіжності Кульбака-Лейблера сприяє тому, що розподіл ймовірностей у низьковимірному просторі наближається до розподілу у високовимірному просторі, забезпечуючи збереження структури даних. Використання t-розподілу Стьюдента дозволяє зменшити проблему "скупченості" точок, яка часто виникає при застосуванні інших методів зменшення розмірності. Важливо зазначити, що t-SNE добре зберігає локальні структури (тобто близькість між точками), проте не гарантує збереження глобальних відстаней між кластерами. Таким чином, хоча кластери, отримані за допомогою t-SNE, відображають групи подібних точок, відстань між цими кластерами може не відповідати фактичній відстані між групами у вихідному багатовимірному просторі.

2.2.3 Аналіз UMAP

Алгоритм уніформного наближення та проєкції многовидів (UMAP) є сучасною технікою зменшення розмірності, яка відзначається своєю ефективністю у візуалізації складних багатовимірних структур даних. Цей метод, що став важливим інструментом у багатьох галузях досліджень, базується на фундаментальному припущенні, що дані розподілені на ріманновому многовиді. У межах цієї концепції ріманновий многовид розглядається як простір, у якому локальна структура характеризується послідовною або майже послідовною ріманновою метрикою. Це означає, що в локалізованих областях многовиду вимірювання відстаней між точками є узгодженими та

передбачуваними. Крім того, топологія многовиду має достатню локальну складність для адекватного відображення взаємозв'язків у даних, що дозволяє алгоритму розпізнавати та зберігати важливі структурні особливості.

UMAP використовує нечітке топологічне представлення, яке забезпечує гнучкість для моделювання складних взаємозв'язків між точками даних. Замість жорстких визначень сусідства алгоритм застосовує нечіткі множини, у яких об’єкти можуть належати до кількох груп з різним ступенем ймовірності. Такий підхід дозволяє UMAP враховувати як локальні, так і глобальні аспекти даних, що критично важливо для збереження їхньої складної структури. У результаті алгоритм створює низьковимірне вбудовування, яке відображає ключові структурні характеристики оригінальних даних, роблячи їх придатними для візуалізації та подальшого аналізу.

Методологія UMAP охоплює кілька ключових етапів. Спочатку створюється зважений граф для представлення локальної структури даних. У цьому графі вузли відповідають окремим об’єктам даних, а ребра визначаються на основі метрики відстані між найближчими сусідами. Вибір метрики відстані є критичним етапом, оскільки він визначає, як обчислюється подібність між об'єктами. Кожне ребро в графі характеризується функцією приналежності, яка кількісно оцінює ймовірність зв'язку між парами вузлів. Ця функція приналежності є нечіткою, тобто вона не є бінарною, а дозволяє об’єктам належати до кількох груп із різним ступенем впевненості. Така нечітка структура забезпечує гнучке моделювання складних взаємозв'язків, враховуючи як локальні, так і глобальні аспекти даних, що є вирішальним для збереження структурної цілісності.

Алгоритм UMAP починається зі встановлення набору найближчих сусідів для кожного об’єкта, виходячи з обраної метрики відстані. Цей етап є критичним, оскільки він визначає локальну структуру даних, яку алгоритм прагне зберегти. Після ідентифікації сусідів обчислюються ваги для кожного з'єднання на основі

локальної щільності даних. Об'єкти в щільніших регіонах отримують вищі ваги, що відображає їхню важливість у структурі даних. Важливим етапом є нормалізація ваг, яка забезпечує правильність топологічного представлення та мінімізує втрату інформації під час зменшення розмірності. Нормалізація гарантує, що ваги мають узгоджений масштаб, що дозволяє алгоритму ефективно порівнювати з'єднання між різними регіонами даних. Після нормалізації ваг формується орієнтований граф, який захоплює ключові особливості локальної топології. Орієнтований характер графа дозволяє враховувати спрямованість зв'язків між об'єктами, що є важливим для збереження їхніх взаємозв'язків.

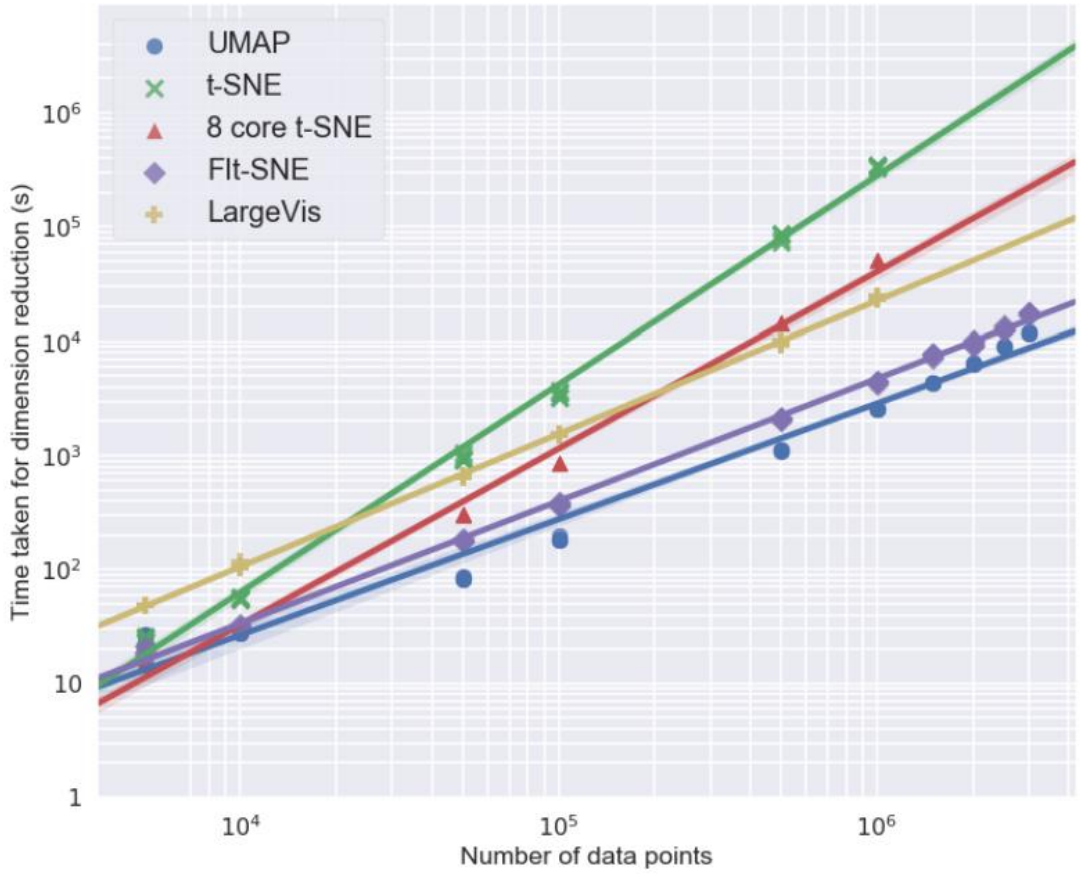


Рис. 2.4 Порівняння швидкості виконання алгоритмів зменшення розмірності

Для побудови низьковимірного вбудовування використовується оптимізація, яка мінімізує розходження між початковою та проєктованою

структурою графа. Ця оптимізація досягається шляхом максимізації схожості між вагами ребер у початковому та зменшеному просторах. По суті, алгоритм прагне знайти розташування об’єктів у низьковимірному просторі, де їхні зв’язки тісно відповідають зв’язкам у початковому просторі. У випадках, коли граф має дубльовані зв'язки, вони агрегуються, що спрощує структуру без втрати її основних характеристик. Агрегація дубльованих зв’язків знижує складність графа, що полегшує обчислення та підвищує ефективність алгоритму.

UMAP має значні переваги над іншими методами зменшення розмірності, такими як t-SNE, завдяки своїй здатності зберігати як локальні, так і глобальні структури даних. Хоча t-SNE ефективно візуалізує локальні структури, він часто не справляється із збереженням глобального контексту даних. Натомість UMAP здатний надійно представляти як дрібні деталі, так і загальну організацію даних. Його застосування охоплює широкий спектр завдань, включно з аналізом складних біологічних систем, вивченням текстових даних, обробкою зображень і моделюванням поведінки в складних багатофакторних системах. У біології UMAP використовується для аналізу геномних даних і ідентифікації різних популяцій клітин. У природній обробці мови UMAP допомагає візуалізувати семантичні взаємозв’язки між словами та документами. У обробці зображень UMAP дозволяє зменшити розмірність зображень, зберігаючи їх ключові характеристики. Завдяки своїй універсальності та масштабованості, UMAP є важливим інструментом для сучасних досліджень, які вимагають глибокого розуміння багатовимірних структур. Алгоритм ефективно обробляє великі набори даних, що робить його придатним для аналізу складних систем.

UMAP забезпечує значні переваги у порівнянні з іншими методами, такими як t-SNE, завдяки здатності зберігати як локальні, так і глобальні структури даних. Його застосування охоплює широкий спектр задач, включаючи аналіз складних біологічних систем, вивчення текстових даних, обробку

зображень і моделювання поведінки у складних багатофакторних системах. Завдяки своїй універсальності та масштабованості, UMAP є важливим інструментом для сучасних досліджень, які потребують глибокого розуміння багатовимірних структур.

Рис. 2.5 Результат зменшення розмірності
векторів вбудовувань за допомогою алгоритму UMAP

Множина ребер графа моделюється як нечітка множина, де кожне ребро відповідає випадковій величині, що підпорядковується закону Бернуллі. Алгоритм будує новий граф у низьковимірному просторі, прагнучи наблизити множину його ребер до вихідної, аби зберегти топологічну структуру. Для цього використовується мінімізація суми дивергенцій Кульбака-Лейблера між кожним ребром початкової та нової нечітких множин.

Задача оптимізації розв’язується за допомогою стохастичного градієнтного спуску. На основі отриманої множини ребер визначається нове

розташування об’єктів, що дозволяє створити низьковимірне відображення вихідного простору.

Метод демонструє високу ефективність для широкого спектра задач. Згідно з документацією відповідної бібліотеки, він успішно застосовується в таких завданнях, як розпізнавання зображень, кластеризація та класифікація зірок. Результати, отримані за допомогою UMAP, є якісними та подібними до результатів алгоритму t-SNE, який вважається одним із найкращих. Водночас UMAP має значно вищу швидкість роботи.

2.2.4 Аналіз LargeVis

Ця публікація представляє LargeVis, новий метод візуалізації, розроблений для ефективної обробки обчислення розміщення для масивних багатовимірних наборів даних, що є складним завданням для існуючих методів, таких як t-SNE. Хоча t-SNE виявився ефективним для менших наборів даних, його продуктивність значно погіршується при роботі з мільйонами точок даних і сотнями вимірів через обчислювальні вузькі місця. LargeVis безпосередньо вирішує ці обмеження завдяки кільком ключовим нововведенням:

Основним вузьким місцем у t-SNE є побудова графа KNN, який базується на деревах точок огляду, що стають неефективними у багатовимірних просторах. LargeVis використовує значно покращений алгоритм для наближеного побудування графа KNN, що базується на деревах випадкових проєкцій. Цей новий підхід пропонує вищу точність і значно швидше обчислення, особливо у багатовимірних сценаріях. Це вирішує перше основне вузьке місце t-SNE.

LargeVis представляє нову ймовірнісну модель для візуалізації графа KNN. На відміну від попередніх методів, ця модель явно враховує як спостережувані (існуючі), так і неспостережувані (неіснуючі) зв'язки в межах графа. Моделюючи як позитивні, так і негативні відносини, LargeVis ефективніше зберігає основну структуру багатовимірних даних у нижчовимірній візуалізації. Це гарантує, що

подібні точки даних розміщуються близько одна до одної, а несхожі точки залишаються далеко одна від одної, що призводить до більш точних та інтерпретованих візуалізацій.

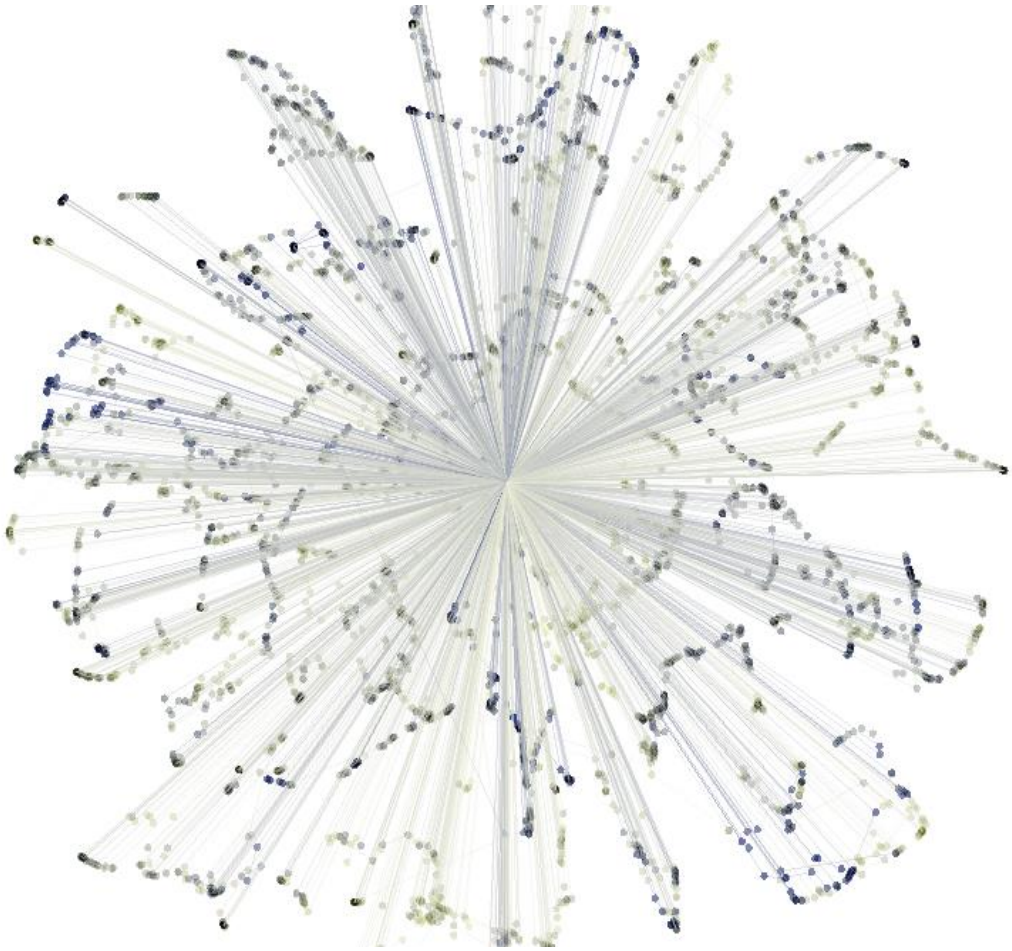


Рис. 2.6 Результат зменшення розмірності
векторів вбудовувань за допомогою алгоритму LargeVis

Процес оптимізації, що використовується t-SNE, також стає обчислювально витратним з великими наборами даних. LargeVis використовує асинхронний стохастичний градієнтний спуск для оптимізації своєї цільової функції. Ця стратегія оптимізації масштабується лінійно з розміром даних (O(N)), що робить її значно ефективнішою для великих наборів даних порівняно з методами, що використовуються t-SNE. Це вирішує друге основне вузьке місце t-SNE.

Ще одним недоліком t-SNE є його чутливість до налаштувань параметрів. Знаходження оптимальних параметрів часто вимагає тривалих і трудомістких

пошуків, особливо для великих наборів даних. LargeVis демонструє більшу стабільність параметрів для різних наборів даних, зменшуючи потребу в ретельному налаштуванні та роблячи його більш практичним для реальних застосувань.

Автори провели великі експерименти на різних реальних наборах даних, включаючи текст (слова та документи), зображення та мережі, щоб оцінити продуктивність LargeVis. Результати показують, що LargeVis досягає порівнянної якості візуалізації з t-SNE на менших наборах даних і створює більш інтуїтивно зрозумілі та інформативні візуалізації на більших наборах даних. Важливо, що LargeVis демонструє значно покращену ефективність, будучи до тридцяти разів швидшим у побудові графа та в сім разів швидшим у візуалізації графа на наборі даних з трьома мільйонами точок даних і ста вимірами. Це покращення продуктивності дозволяє LargeVis візуалізувати мільйони багатовимірних точок даних на одному комп'ютері всього за кілька годин.

Підсумовуючи, LargeVis пропонує значний прогрес у візуалізації даних великого масштабу, вирішуючи обчислювальні вузькі місця існуючих методів. Його ефективне побудування графа KNN, принципова ймовірнісна модель, масштабована оптимізація та стабільність параметрів роблять його потужним інструментом для дослідження та розуміння складних багатовимірних даних.

Цей розділ заглиблюється в специфіку LargeVis, нового методу візуалізації багатовимірних даних. Ось розбивка його ключових компонентів:

LargeVis визнає обмеження існуючих методів (таких як t-SNE) у побудові графів KNN для великих наборів даних. Ці методи стають обчислювально витратними через часову складність O(N^2d) обчислення парних подібностей.

LargeVis використовує дерева випадкових проєкцій, метод, відомий ефективним пошуком найближчих сусідів у багатовимірних просторах. Однак досягнення високої точності за допомогою цього методу зазвичай вимагає побудови багатьох дерев, що впливає на ефективність.

LargeVis пропонує рішення: замість численних дерев для високої точності, він будує кілька дерев для початкового наближеного графа KNN. Потім він використовує «дослідження сусідів» для покращення точності. Цей метод використовує принцип, що «сусід мого сусіда, ймовірно, також є моїм сусідом». Шукаючи сусідів сусідів, LargeVis уточнює початковий граф KNN.

Ваги ребер у графі KNN визначаються аналогічно t-SNE, з урахуванням перплексії.

Після побудови графа KNN LargeVis проєктує вузли графа в низьковимірний простір (зазвичай 2D або 3D) для візуалізації.

Основним принципом цієї моделі є збереження подібностей між вершинами (точками даних) у низьковимірному просторі. Це означає, що подібні точки даних повинні бути розміщені близько одна до одної, тоді як несхожі точки повинні бути далеко одна від одної.

LargeVis визначає ймовірність спостереження ребра (зв'язку) між двома вершинами на основі відстані між їхніми вкладеннями (низьковимірними представленнями) у цільовому просторі. Менші відстані відповідають вищим ймовірностям ребер.

Модель виходить за межі спостереження ребер, щоб розглянути неіснуючі ребра (негативні ребра). Вона призначає ймовірність зваженим ребрам, включаючи як спостережувані ребра, так і негативні ребра. Мета полягає в максимізації функції, яка відображає ймовірність спостережуваних ребер (зберігаючи подібні точки близько) та ймовірність негативних ребер (зберігаючи несхожі точки далеко).

Безпосередня максимізація цієї функції є обчислювально витратною через велику кількість негативних ребер. LargeVis використовує методи негативного семплінгу для вирішення цієї проблеми. Він випадковим чином вибирає деякі негативні ребра для кожної вершини та розглядає їх як негативні зв'язки під час оптимізації моделі.

Для ефективної оптимізації цільової функції LargeVis використовує асинхронний стохастичний градієнтний спуск, який особливо підходить для розріджених графів, таких як графи KNN. Цей підхід дозволяє паралельні оновлення без значних конфліктів між потоками через розрідженість графа.

Часова складність оптимізації є лінійною від кількості точок даних (O(sMN)), де s – розмірність цільового простору (наприклад, 2 або 3), а M – кількість негативних зразків на позитивне ребро. Це демонструє масштабованість LargeVis для великих наборів даних.

2.3 Порівняння методів зменшення розмірності

У сучасних умовах спостерігається експоненційне зростання обсягів даних, що створює суттєві виклики для їх обробки та інтерпретації. Особливо складними є високовимірні набори даних, що містять велику кількість змінних, через їхню складність у виявленні прихованих закономірностей та взаємозв'язків. Для вирішення цієї проблеми застосовуються методи зменшення розмірності, що дозволяють знизити складність даних, зберігаючи при цьому їх ключові характеристики.

Метою даного дослідження є порівняльний аналіз чотирьох найбільш поширених методів зменшення розмірності: методу головних компонент (PCA), t-розподіленого стохастичного вкладення сусідів (t-SNE), уніфікованого наближення та проєкції многовидів (UMAP) та LargeVis. Оцінка цих методів проводиться за кількома критеріями, зокрема швидкість виконання, здатність зберігати локальну та глобальну структуру даних, зручність інтерпретації результатів та придатність до застосування в різних сценаріях.

Кожен з розглянутих методів має свої особливості, переваги та обмеження. PCA є лінійним методом, що забезпечує високу швидкість обробки та простоту інтерпретації, але може бути менш ефективним при роботі з нелінійними

даними. Метод t-SNE ефективно відображає нелінійні структури, проте є обчислювально витратним і чутливим до вибору параметрів. UMAP прагне забезпечити баланс між швидкістю та якістю, зберігаючи як локальну, так і глобальну структуру даних. LargeVis розроблений для роботи з великими наборами даних, надаючи високу швидкість та масштабованість.

Для оцінки ефективності цих методів було проведено серію експериментів на різних наборах даних, що дозволяє визначити оптимальні умови їх використання. Результати експериментів, що наведені в подальших розділах роботи, включають час виконання кожного методу, їх здатність до візуалізації кластерів та збереження структури даних, а також обговорення їхніх сильних та слабких сторін. Отримані результати дозволяють сформулювати рекомендації щодо вибору найбільш підходящого методу зменшення розмірності залежно від специфіки даних та вимог конкретного завдання.

PCA (Метод головних компонент) є найшвидшим методом, який потребує лише 10 хвилин для виконання. PCA досягає виняткової швидкості завдяки своїй залежності від принципів лінійної алгебри, що дозволяє ефективно зменшувати розмірність даних. Цей алгоритм швидко визначає головні компоненти, майже миттєво зменшуючи розмірність даних, навіть для великих наборів даних. Він особливо підходить для систем реального часу, чутливих до обчислювальних витрат, і відзначається мінімальним обчислювальним навантаженням порівняно з альтернативними методами.

Переваги PCA полягають у його ефективності для лінійних структур даних. Як базовий інструмент попередньої обробки, він виділяє найважливіші змінні з наборів даних, спрощуючи подальший аналіз. Його простота виключає необхідність у складному налаштуванні параметрів, що робить його інтуїтивно зрозумілим і доступним навіть для недосвідчених користувачів. Крім того, результати легко інтерпретуються, оскільки вони базуються на декомпозиції ортогональних векторів, що сприяє прямолінійним висновкам.

Однак PCA погано підходить для даних із складними, нелінійними залежностями. У таких випадках він не зберігає важливих взаємозв'язків між змінними, що може призвести до втрати інформації. Крім того, якщо вхідні дані не стандартизовані, PCA може дати неточні результати. Він також чутливий до шуму, який може спотворювати його виходи, тому необхідна попередня очистка даних.

PCA рекомендується для сценаріїв, що включають лінійні структури даних або коли важлива обчислювальна ефективність. Найкраще його використовувати як початковий етап у попередній обробці перед застосуванням більш складних аналітичних методів. Ідеально підходить для невеликих і середніх наборів даних із простими структурами, забезпечуючи швидку оцінку основних напрямків даних. Стандартизація набору даних перед застосуванням настійно рекомендується для оптимізації його продуктивності.

t-SNE (t-розподілене стохастичне вкладення сусідів) є найповільнішим методом, що потребує приблизно 45 хвилин для виконання. Складність алгоритму обумовлена багатоступінчастим оптимізаційним процесом, який накладає значні обчислювальні витрати. Час виконання зростає експоненційно зі збільшенням розмірності чи обсягу даних. Хоча високопродуктивне обладнання може частково пом'якшити ці обмеження, фундаментальні недоліки залишаються значними.

t-SNE відзначається своєю здатністю до візуалізації кластерних структур у даних, забезпечуючи неперевершену чіткість у цій галузі. Він особливо ефективний для виявлення прихованих шаблонів і взаємозв'язків, що робить його безцінним для дослідницького аналізу даних. Завдяки відображенню складних залежностей t-SNE перетворює набори даних у високозрозумілі візуальні представлення, полегшуючи ідентифікацію латентних груп і структур. Цей метод незамінний для високовимірних наборів даних, які важко візуалізувати прямолінійно.

Проте обчислювальні витрати t-SNE є надзвичайно високими для великих наборів даних. Залежність алгоритму від випадкової ініціалізації вводить варіативність, ускладнюючи відтворюваність результатів. Крім того, він надає перевагу локальній структурі даних над глобальною, що може приховувати ширші закономірності набору даних. Значні ресурси, як часові, так і обчислювальні, необхідні для його виконання, що робить його непридатним для завдань з обмеженим часом.

t-SNE найкраще підходить для завдань візуалізації, де ідентифікація кластерів є головною, а швидкість виконання — другорядною. Його рекомендується використовувати для подання даних або дослідницького аналізу, спрямованого на виявлення унікальних шаблонів. Хоча t-SNE оптимальний для невеликих наборів даних, він вимагає ітеративного налаштування параметрів для надійних результатів. Він найбільш ефективний, коли основною метою є чітке пояснення локальних структур даних.

UMAP (уніфіковане наближення та проєкція многовидів) досягає балансу між швидкістю та якістю, виконуючи завдання за 25 хвилин. Алгоритм UMAP ефективно управляє як локальними, так і глобальними структурами даних, що робить його універсальним інструментом для різних аналітичних контекстів. Його масштабованість дозволяє обробляти набори даних із мільйонами точок, зберігаючи розумні обчислювальні вимоги та час виконання.

Переваги UMAP полягають у його здатності зберігати як детальні локальні взаємозв'язки, так і загальні глобальні шаблони в даних. Ця подвійна можливість робить його надійним вибором для аналізу різнорідних наборів даних. Гнучкість методу в налаштуванні параметрів дозволяє адаптувати його до специфічних аналітичних потреб, підвищуючи його корисність. Крім того, результати UMAP зазвичай легко інтерпретувати, особливо при аналізі кластерних структур.

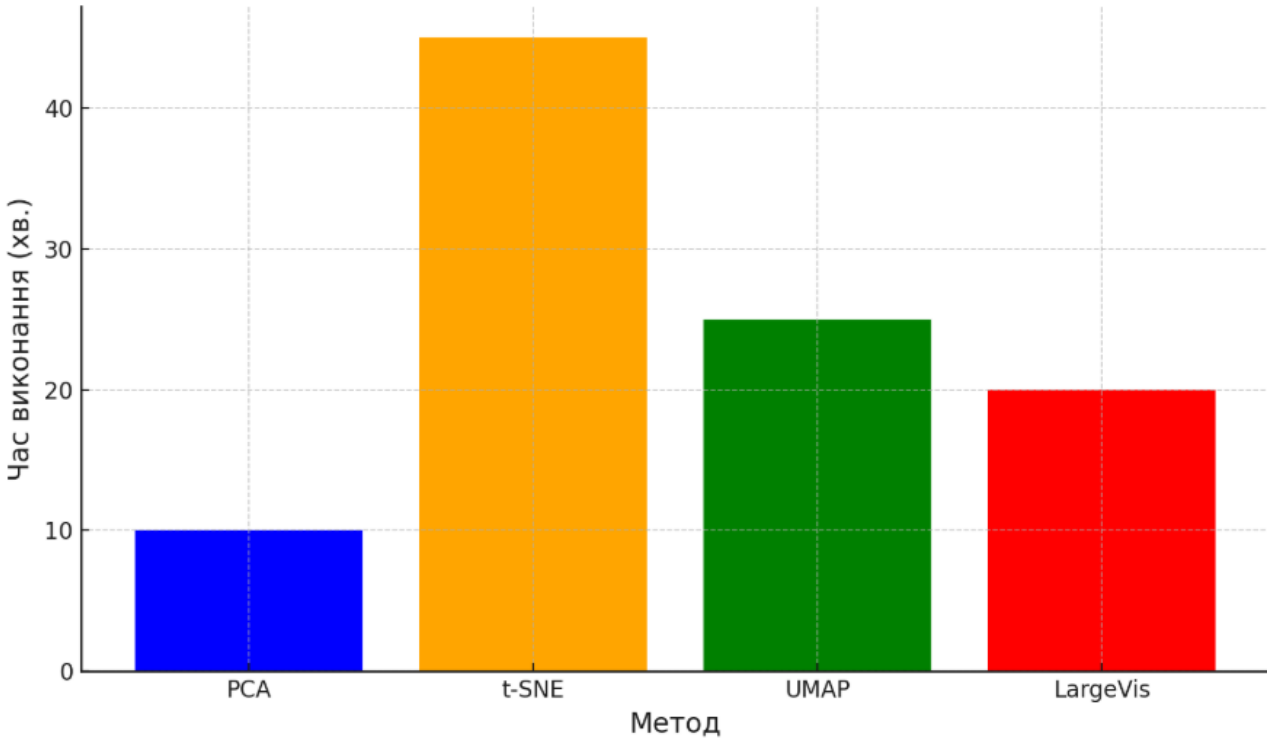


Рис. 2.1 Порівняння швидкості виконання алгоритмів зменшення розмірності

Однак UMAP вимагає ретельного налаштування параметрів, зокрема таких, як "n_neighbors" і "min_dist", які суттєво впливають на результати. Неправильна параметризація може призвести до втрати важливої інформації. Для початківців процес налаштування параметрів може бути складним. Крім того, великі набори даних можуть потребувати значних ресурсів пам'яті, що накладає додаткові обчислювальні витрати.

UMAP рекомендується як універсальний підхід для великих наборів даних, що балансує між швидкістю та аналітичною точністю. Він особливо підходить для складних даних із багатовимірними залежностями. Експерименти з параметрами є необхідними для оптимізації його продуктивності. UMAP є найбільш ефективним, коли збереження як локальних, так і глобальних зв'язків даних є критичним для аналізу.

LargeVis виконується за 20 хвилин і оптимізований для роботи з великими наборами даних. Розроблений для високовимірних даних, LargeVis ефективно

зменшує розмірність за допомогою спрощеного алгоритму. Він може обробляти мільйони точок даних за відносно короткий час, що робить його ідеальним рішенням для завдань масштабного аналізу.

LargeVis пропонує неперевершену продуктивність у роботі з великими наборами даних, де інші методи можуть виявитися недостатніми. Його алгоритм зберігає як локальні, так і глобальні структури даних, забезпечуючи чіткі та всеохоплюючі візуалізації. Метод ефективно масштабується, що забезпечує його застосовність до сучасних викликів аналізу даних. Крім того, його скромні апаратні вимоги підвищують доступність у різних обчислювальних середовищах.

Однак відносна невідомість LargeVis створює певні труднощі, такі як обмежена документація та підтримка спільноти. Його інтерпретація може бути складною для користувачів із мінімальним досвідом, що вимагає глибокого розуміння алгоритму для ефективного застосування. Налаштування параметрів вимагає ретельності, а у випадках надзвичайно складних даних результати можуть поступатися деталізацією порівняно з t-SNE або UMAP.

LargeVis особливо ефективний для аналізу великих, високовимірних наборів даних, де його ефективність перевершує альтернативні методи. Рекомендується для сценаріїв, що вимагають швидкої обробки та чіткої візуалізації великих структур даних. Коли інші методи виявляються занадто ресурсомісткими, LargeVis забезпечує надійну альтернативу для отримання значущих інсайтів із великих наборів даних.

2.4 Математична модель адаптованого методу візуалізації

Наведений текст описує метод LargeVis, призначений для візуалізації великомасштабних та багатовимірних наборів даних. Формально, маючи набір даних $X = \{x_i \in R^d\}_{i=1,2,..,N}$, метою є представлення кожної точки даних x_i

вектором $y_i \in R^s$, де s зазвичай дорівнює 2 або 3. Основна ідея візуалізації полягає у збереженні внутрішньої структури даних у низьковимірному просторі.

Існуючі підходи часто спочатку обчислюють подібності між усіма парами $\{x_i, x_j\}$ та потім зберігають ці подібності у низьковимірному відображенні. Оскільки обчислення попарних подібностей є обчислювально витратним ($O(N^2d)$), сучасні методи, такі як t-SNE, спочатку конструюють граф K-найближчих сусідів (KNN) та потім проєктують цей граф у двовимірний простір. LargeVis дотримується цієї процедури, використовуючи ефективний алгоритм побудови KNN графу та ґрунтовну ймовірнісну модель для візуалізації графу.

Побудова KNN графу вимагає метрики відстані. У LargeVis використовується евклідова відстань $||x_i - x_j||$ у багатовимірному просторі, аналогічно до t-SNE. Побудова точного KNN графу має часову складність $O(N^2d)$, що є неприйнятним для великих обсягів даних. Для наближеної побудови KNN графу використовуються дерева випадкових проєкцій. Алгоритм починається з розділення всього простору та побудови дерева. Для кожного нелистового вузла дерева алгоритм вибирає випадкову гіперплощину для розділення підпростору, що відповідає цьому вузлу, на два підпростори, які стають дочірніми вузлами. Гіперплощина визначається шляхом випадкового вибору двох точок з поточного підпростору та взяттям гіперплощини, що знаходиться на однаковій відстані від цих двох точок. Цей процес продовжується до досягнення певного порогу кількості вузлів у підпросторі. Після побудови дерева випадкових проєкцій кожна точка даних може пройти по дереву для знаходження відповідного листового вузла. Точки в підпросторі цього листового вузла розглядаються як кандидати в найближчі сусіди вхідної точки даних. Для підвищення точності на практиці будується кілька дерев.

Однак, побудова точного KNN графу вимагає значної кількості дерев, що суттєво знижує ефективність. Для вирішення цієї проблеми LargeVis використовує підхід дослідження сусідів: "сусід мого сусіда, ймовірно, також є

моїм сусідом". Замість побудови великої кількості дерев для отримання високоточного KNN графу, будується невелика кількість дерев для створення наближеного графу. Потім для кожного вузла графу здійснюється пошук сусідів його сусідів, які також є потенційними кандидатами в найближчі сусіди. Цей процес може повторюватися кілька разів для підвищення точності графу. Практика показує, що кількох ітерацій достатньо для досягнення майже 100% точності.

Для визначення ваг ребер у KNN графі використовується підхід, аналогічний t-SNE. Умовна ймовірність переходу від точки x_i до x_j обчислюється за формулою:

$$p_{ji} = exp(-||x_i - x_j||^2/2\sigma_i^2) \,/\, \Sigma_{(i,k)} \in E\ exp(-||x_i - x_k||^2/2\sigma_i^2), \text{де } p_i|_i = 0.$$

Параметр σ_i вибирається таким чином, щоб перплексія умовного розподілу $p\cdot|_i$ дорівнювала заданому значенню u. Потім граф симетризується шляхом встановлення ваги між x_i та x_j як:

$$w_{ij} = \frac{p_{ji} + p_{ij}}{2N}, \tag{3.1}$$

Після побудови KNN графу для візуалізації даних необхідно спроєктувати вузли графу у дво- або тривимірний простір. LargeVis використовує для цього ймовірнісну модель, яка зберігає подібності між вершинами у низьковимірному просторі. Ймовірність спостереження бінарного ребра $e_{ij} = 1$ між вершинами v_i та v_j визначається як:

$$P(e_{ij} = 1) = f\left(\left||y_i - y_j|\right|\right), \tag{3.2}$$

де y_i – відображення вершини v_i у низьковимірному просторі, а $f(\cdot)$ – ймовірнісна функція від відстані між y_i та y_j. Коли y_i знаходиться близько до y_j, ймовірність спостереження ребра є високою. Можуть використовуватися різні функції, наприклад $f(x) = 1/(1+ax^2)$ або $f(x) = 1/(1+\exp(x^2))$.

Для узагальнення на зважені ребра, ймовірність спостереження ребра $e_{ij} = w_{ij}$ визначається як:

$$P(e_{ij} = w_{ij}) = P(e_{ij} = 1)^{wij}, \quad (3.3)$$

Для зваженого графу G = (V, E) функція правдоподібності обчислюється як:

$$O = \Pi_{(i,j) \in E} p(e_{ij} = 1)^{wij} \Pi_{(i,j) \in E^-} (1 - p(e_{ij} = 1))^{\gamma} \propto \Sigma_{(i,j) \in E} w_{ij} \log p(e_{ij} = 1) + \Sigma_{(i,j) \in E^-} \gamma \log(1 - p(e_{ij} = 1)), \quad (3.4)$$

де E^- – множина пар вершин, між якими немає ребер;

γ – вага, що присвоюється негативним ребрам.

Оптимізація цієї функції є обчислювально складною через квадратичну кількість негативних ребер. Для вирішення цієї проблеми використовується техніка негативного семплювання. Для кожної вершини i випадково вибираються вершини j згідно з розподілом:

$$P_n(j) \propto d_j^{0.75}, \quad (3.5)$$

де d_j – степінь вершини j.

Модифікована функція правдоподібності має вигляд:

$$O = \Sigma_{(i,j) \in E} w_{ij} \log p(e_{ij} = 1) + \Sigma_{k = 1}^{M} E_{jk} \sim P_n(j) \gamma \log(1 - p(e_{ijk} = 1)).$$

Для оптимізації використовується семплювання ребер з ймовірністю, пропорційною їх вагам, та асинхронний стохастичний градієнтний спуск для прискорення навчання. Часова складність оптимізації становить:

$$O(sMN), \quad (3.6)$$

де

M – кількість негативних семплів;

s – розмірність низьковимірного простору, а N – кількість вершин.

3 АДАПТАЦІЯ МЕТОДИКИ ВІЗУАЛІЗАЦІЇ БАГАТОВИМІРНИХ ДАНИХ ДЛЯ ВІДЛАГОДЖЕННЯ ШІ-СИСТЕМ

3.1 Інтерактивна довідкова ШІ-система

Інтерактивна довідкова ШІ-система створена для забезпечення високоточної обробки складних інформаційних запитів через інтеграцію Retrieval-Augmented Generation (RAG), великих мовних моделей (LLMs) та векторних баз даних. Ця система спрямована на покращення доступу до знань, надаючи ефективну, масштабовану та зручну у використанні інфраструктуру. Архітектура системи включає пошуковий компонент, що використовує спеціалізовані векторні бази даних для зберігання багатовимірних текстових вбудовувань. Пошук здійснюється за метриками подібності, які дозволяють ідентифікувати найбільш релевантні документи чи текстові фрагменти, забезпечуючи швидкий доступ до потрібної інформації. Особлива увага приділяється впровадженню розширених метаданих, які сприяють більш гнучкому фільтруванню результатів за категоріями, часовими межами чи іншими параметрами, що розширює можливості користувача.

Генеративний компонент базується на великих мовних моделях і забезпечує створення адаптивних відповідей, використовуючи релевантні дані як контекст. Параметри генерації дозволяють підлаштовувати стиль і тон відповіді відповідно до потреб користувача, що робить систему більш персоналізованою та зручною у використанні. Інфраструктура RAG реалізує двоетапний процес: спочатку здійснюється вибір релевантної інформації з векторної бази даних, після чого вона подається до мовної моделі для створення точного і змістовного результату. Інтерактивний інтерфейс пропонує

багатомодальні методи введення, що включають текстові запити, голосовий ввід та навіть зображення, а також підтримує візуалізацію результатів пошуку, що сприяє кращому розумінню даних. Крім того, інтегровані механізми зворотного зв’язку дозволяють користувачам оцінювати відповіді, тим самим вдосконалюючи систему в реальному часі.

Система надає широкий спектр можливостей для пошуку інформації у великих корпусах, включаючи наукові публікації, технічну документацію чи внутрішні корпоративні бази даних. Високошвидкісний доступ до даних забезпечує ефективну обробку інформаційних запитів, що особливо важливо для задач, які потребують обробки великих обсягів інформації в реальному часі. Генерація відповідей значно спрощує створення узагальнень, детальних пояснень чи інструкцій, що дозволяє покращити якість обслуговування та зменшити час на виконання рутинних завдань. Аналітичні можливості системи інтегруються з сучасними інструментами візуалізації та аналітики, що дозволяє створювати складні графіки, моделі чи звіти на основі як текстових, так і числових даних. Завдяки механізмам донавчання система адаптується до специфічних потреб організацій, що робить її надзвичайно гнучкою. Інтеграція з API надає додаткові можливості для автоматизації бізнес-процесів та створення індивідуальних рішень.

Великі мовні моделі відіграють ключову роль у розумінні складних мовних конструкцій і контексту, що дозволяє створювати високоточні, релевантні та адаптивні відповіді. Векторні бази даних, розроблені для роботи з багатовимірними просторами текстових вбудовувань, забезпечують надзвичайно швидке виконання пошукових запитів навіть у межах великих масивів даних. Методи обробки природної мови, включаючи токенізацію, семантичний аналіз, класифікацію та вилучення сутностей, забезпечують багаторівневу обробку тексту, що сприяє глибшому розумінню запитів. Хмарна та розподілена обчислювальна інфраструктура у поєднанні з оптимізованими

бібліотеками сприяє масштабованості та забезпечує високу продуктивність, знижуючи витрати на обчислення. Це особливо важливо для великих організацій, які працюють з об’ємними наборами даних та потребують надійної інфраструктури.

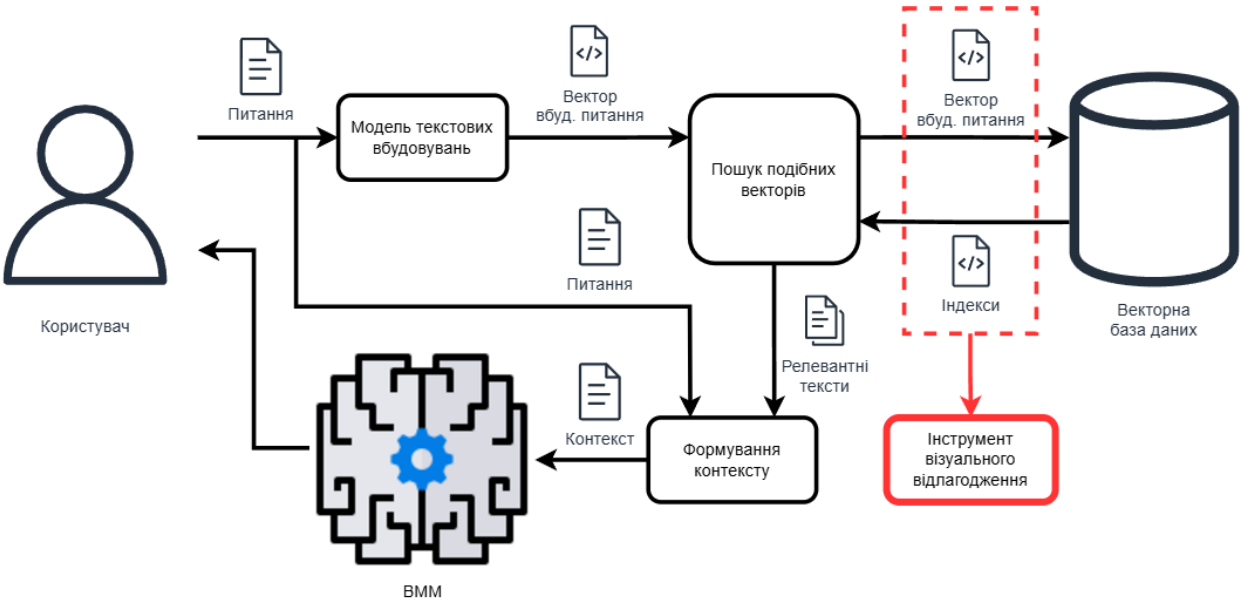


Рис. 3.1 Структура інтерактивної довідкової ШІ-системи

Потенційні застосування системи охоплюють різноманітні галузі, включаючи автоматизацію доступу до корпоративних знань, прискорення пошуку й узагальнення наукової інформації, інтерактивне навчання, надання технічної підтримки та багато іншого. Наприклад, система може використовуватись у медичних дослідженнях для швидкого аналізу клінічних даних, у фінансових установах для аналізу ринкових тенденцій або у сфері освіти для створення адаптивних навчальних програм. Однак, впровадження подібних систем пов’язане з низкою викликів, серед яких оптимізація алгоритмів для забезпечення максимальної точності, розробка метрик для оцінювання релевантності, а також дотримання стандартів конфіденційності, безпеки та етичності обробки даних. Особлива увага приділяється забезпеченню нейтральності системи, щоб уникнути упередженості в створенні відповідей.

Ефективне скорочення витрат на обчислення досягається через впровадження гібридних архітектур, які поєднують локальні обчислювальні ресурси з хмарними рішеннями. Такий підхід дозволяє не лише зменшити витрати, але й забезпечити високу швидкодію та гнучкість. Таким чином, інтерактивна довідкова ШІ-система, яка об’єднує RAG, великі мовні моделі та векторні бази даних, постає як революційний інструмент для автоматизації обробки інформаційних запитів. Вона надає користувачам потужні інструменти для роботи з інформацією, що значно підвищує ефективність і відкриває нові можливості у різних сферах діяльності. Її впровадження сприяє створенню інноваційних рішень, які можуть адаптуватися до динамічних потреб сучасного суспільства, забезпечуючи високу якість та доступність знань.

3.2 Опис вибірки текстів у векторній базі даних

Процес формування векторної бази даних знань було реалізовано через систематичний відбір текстів, що відповідають специфічним критеріям релевантності, якості й достатності обсягу. Основна мета полягала в створенні інформаційної основи для високоефективного функціонування чат-бота, орієнтованого на обслуговування клієнтів інтернет-магазину електроніки. Для цього було відібрано тексти, що тематично охоплюють опис товарів, технічні специфікації, багатопланові відгуки користувачів, експертні статті з блогів про технології, а також структуровані відповіді на часто задавані питання. Загальний обсяг вибірки включав 100 000 текстів із детальними описами та відгуками, 500 статей аналітичного характеру й 1 000 пар запитань-відповідей, що дозволяє забезпечити всебічне покриття користувацьких запитів.

Для побудови цієї вибірки були залучені різноманітні джерела. Серед відкритих даних ключову роль відіграли статті з Вікіпедії, які містять загальні відомості про електроніку, а також корпуси текстів Amazon Reviews, що

репрезентують реальні відгуки користувачів. Власні дані інтернет-магазину, зокрема база товарів із докладними технічними характеристиками та записи розмов із клієнтами, доповнювали вибірку, створюючи контекстуальну різноманітність. Крім того, для збору додаткової інформації застосовувалися API виробників електроніки та методи веб-скрейпінгу офіційних вебсайтів, що дозволяло отримувати найбільш актуальні й спеціалізовані дані.

Вибір текстів ґрунтувався на чітко визначених критеріях. Релевантність гарантувала тематичну відповідність текстів категорії електроніки, їхнім технічним характеристикам і сценаріям використання. Різноманітність забезпечувалася завдяки широкому охопленню товарних категорій (телефони, ноутбуки, телевізори) та врахуванню емоційної тональності відгуків (позитивні, негативні, нейтральні). Якість текстів забезпечувалася за рахунок виключення матеріалів із граматичними помилками й використанням зрозумілих формулювань. Урахування достатності обсягу вибірки було необхідним для забезпечення навчання моделі з високою точністю й продуктивністю.

Підготовка текстів включала кілька етапів обробки. Спочатку тексти проходили процедури очищення, зокрема видалення HTML-тегів, зайвих символів і повторів. На наступному етапі здійснювалася токенізація, яка передбачала розбиття текстів на окремі лексеми, що полегшувало їх подальший аналіз. Нормалізація текстів включала приведення слів до нижнього регістру, лематизацію, яка узагальнює різні словоформи до базової, а також видалення стоп-слів, що не несуть суттєвого змістового навантаження (наприклад, «і», «в», «на»). Цей етап дозволяв оптимізувати тексти для подальшої обробки.

Векторизація текстів здійснювалася за допомогою сучасних методів представлення даних. Word embeddings використовувалися для моделювання семантичної близькості між словами, що забезпечувало кореляцію між, наприклад, «телефон» і «смартфон». Sentence embeddings дозволяли порівнювати запити користувачів із релевантними текстами бази даних,

підвищуючи точність системи. Особливу роль відіграли contextualized word embeddings, такі як BERT, які враховували контекст використання слів, створюючи різні векторні представлення залежно від оточення (наприклад, «батарея телефону» проти «батарея опалення»).

Інфраструктурні аспекти зберігання даних були реалізовані через використання векторної бази даних Faiss, яка забезпечує високопродуктивний пошук. Для оптимізації цього процесу були застосовані алгоритми побудови індексів, такі як HNSW та IVF, що дозволили зменшити час обробки запитів і забезпечити масштабованість системи.

Для ілюстрації, до вибірки були включені описи товарів, наприклад: «Apple iPhone 15 Pro Max. Новий флагманський смартфон із покращеною камерою та потужним процесором A17 Bionic». Серед відгуків траплялися такі, як: «Купив цей телефон місяць тому, дуже задоволений якістю фотографій і швидкістю роботи». Тематичні статті охоплювали ключові питання, як-от: «Як вибрати телефон із найкращою камерою у 2024 році». FAQ включали запитання на кшталт: «Яка гарантія на iPhone 15 Pro Max?» із відповіддю: «Гарантія від виробника 12 місяців».

Таким чином, описана вибірка текстів слугує фундаментальною основою для формування векторної бази знань. Її структура й якість забезпечують ефективність функціонування системи, що дозволяє чат-боту оперативно й точно відповідати на широкий спектр запитів, значно підвищуючи якість обслуговування клієнтів та їхній користувацький досвід.

3.3 Візуалізація багатовимірних даних

Візуалізація багатовимірних даних з векторної бази даних є складним завданням, оскільки ми маємо справу з даними, що існують у просторах з великою кількістю вимірів, наприклад, сотні або навіть тисячі. Традиційні

методи візуалізації, такі як графіки розсіювання або гістограми, добре працюють для 2D або 3D даних, але стають неефективними для даних з більшою кількістю вимірів. Існує кілька методів, які можна використовувати для візуалізації багатовимірних даних з векторної бази даних. Серед них методи зменшення розмірності, такі як метод головних компонент (PCA), який знаходить лінійні комбінації вихідних змінних, що описують найбільшу дисперсію даних, t-SNE (t-distributed Stochastic Neighbor Embedding), що є нелінійним методом зменшення розмірності та добре підходить для візуалізації кластерів у даних, та UMAP (Uniform Manifold Approximation and Projection), ще один нелінійний метод зменшення розмірності, який часто показує кращі результати, ніж t-SNE, особливо для збереження глобальної структури даних. Існують також методи візуалізації на основі відстаней, такі як матриці відстаней, що відображають матрицю попарних відстаней між об'єктами у вигляді теплової карти, та дерева мінімального охоплення (MST), що будують дерево, яке з'єднує всі точки даних з мінімальною сумарною довжиною ребер. Серед інших методів варто відзначити паралельні координати, де кожна вісь представляє один вимір, а кожен об'єкт відображається як лінія, що з'єднує значення за кожним виміром, самоорганізуючі карти Кохонена (SOM), що є нейронною мережею, яка відображає багатовимірні дані на 2D сітку, зберігаючи топологічні властивості даних, та гіперболічне відображення, що відображає багатовимірні дані на гіперболічний простір. Конкретні кроки для візуалізації даних з векторної бази даних включають отримання векторів з бази даних, вибір методу зменшення розмірності або візуалізації, застосування вибраного методу за допомогою відповідного програмного забезпечення або бібліотеки та створення візуалізації даних. Важливо пам'ятати, що візуалізація багатовимірних даних завжди є компромісом між збереженням інформації та зрозумілістю, а вибір конкретного методу залежить від даних та цілей візуалізації.

TensorBoard Projector – це інструмент для візуалізації багатовимірних даних, зокрема векторних представлень (embeddings), у TensorBoard. Він дозволяє досліджувати дані в інтерактивному 3D просторі, знаходити кластери, аналізувати сусідів та багато іншого. Для використання Projector необхідно підготувати дані у форматі TSV, створивши два файли: файл з векторами (embeddings.tsv), де кожен рядок містить векторне представлення одного елемента даних, та файл з метаданими (metadata.tsv), що містить інформацію про кожен вектор. Для використання Projector необхідно створити TensorBoard Summary за допомогою Python та TensorFlow або PyTorch. Після створення summary, необхідно запустити TensorBoard, вказавши директорію з логами. Інтерфейс Projector відображає вектори у 3D просторі, дозволяє шукати вектори за метаданими, розфарбовувати точки на основі метаданих, вибирати методи зменшення розмірності, такі як PCA, t-SNE та UMAP, налаштовувати кількість ітерацій для t-SNE та UMAP, а також показує найближчих сусідів при виборі точки. Projector також підтримує спрайти зображень та, хоча зазвичай використовує евклідову відстань, існують способи використання кастомних функцій відстані. Використовуючи TensorBoard Projector, можна отримати цінну інформацію про структуру даних, виявити кластери, знайти схожі елементи та краще зрозуміти результати моделі.

3.4 Процес відлагодження ШІ-системи

Процес візуального відлагодження інтерактивної довідникової ШІ-системи на основі RAG, великих мовних моделей та векторних баз даних передбачає використання TensorBoard Projector для візуалізації даних, зокрема векторних представлень текстів та запитів. В якості векторної СУБД було обрано LanceDB, а модель вбудовування – bge-multilingual-gemma2. Векторна база даних містить 10 000 текстів з Української Вікіпедії. Для забезпечення

статистичної значущості кожен з методів було опробувано 5 разів з рандомізацією текстових вбудовувань для усереднення результатів дослідження. При кожному повторенні експерименту використовувались 20 унікальних запитів до векторної бази даних. Метою візуалізації є дослідження впливу різних параметрів та методів на якість пошуку релевантних документів у векторній базі даних, що є критично важливим для ефективності RAG-системи.

Для кожного з п'яти повторень експерименту створюються файли з метаданими та файли з векторними представленнями для подальшого аналізу в TensorBoard. Файл метаданих у форматі tsv містить інформацію про кожен текст, таку як заголовок статті та URL, а також відповідний індекс. Файл векторів, також у форматі tsv або bin, містить векторні представлення текстів у порядку, що відповідає порядку рядків у файлі метаданих. Після запуску TensorBoard з вказаною директорією, що містить файли метаданих та векторів, проводиться аналіз візуалізації в TensorBoard Projector. Аналіз включає спостереження за утворенням кластерів, де тексти зі схожим змістом повинні утворювати близькі групи. Розпорошення кластерів може свідчити про проблеми з якістю вбудовувань або з розмірністю векторного простору. За допомогою функції пошуку за ключовими словами в TensorBoard Projector перевіряється, чи знаходяться знайдені вектори поруч з векторами релевантних текстів. Для візуалізації додаткової інформації, наприклад, розділу Вікіпедії, до якого належить текст, використовується кольорове кодування. Важливим етапом є порівняння результатів різних повторень експерименту для оцінки стабільності результатів та впливу рандомізації.

Процес відлагодження та оптимізації передбачає аналіз викидів, тобто векторів, що знаходяться далеко від основних кластерів, які можуть вказувати на помилки в даних або на необхідність покращення моделі вбудовування. Проводиться підбір параметрів пошуку, таких як метрики відстані (наприклад, косинусна відстань, евклідова відстань) та кількість найближчих сусідів.

Візуалізація допомагає зрозуміти вплив цих параметрів на результати пошуку. У випадку, якщо кластери утворюються погано або пошук не дає релевантних результатів, розглядається можливість використання іншої моделі вбудовування або додаткового тренування існуючої на специфічних даних. Також аналізується розподіл запитів шляхом візуалізації векторів запитів разом з векторами текстів для перевірки, чи потрапляють запити в області, де знаходяться релевантні тексти.

Оскільки використовується LanceDB, враховуються її особливості, зокрема підтримка різних метрик відстані, з якими експериментують в TensorBoard для порівняння результатів. LanceDB оптимізована для швидкого пошуку, тому візуалізація не впливає на продуктивність бази даних, але дозволяє зрозуміти, як відбувається пошук. Наприклад, якщо після візуалізації виявляється, що тексти про історію Києва знаходяться далеко від текстів про сучасну культуру Києва, це може свідчити про недостатнє врахування контексту моделлю вбудовування або необхідність використання іншої стратегії вбудовування, наприклад, окремих вбудовувань для різних аспектів Києва. Використання TensorBoard Projector у поєднанні з LanceDB та ретельним аналізом результатів експериментів дозволяє ефективно відлагодити та оптимізувати інтерактивну довідкову ШІ-систему на основі RAG.

Описане дослідження з візуалізації векторних даних, отриманих з RAG-системи, з використанням алгоритмів t-SNE, UMAP, PCA та LargeVis має на меті візуалізацію багатовимірних векторних представлень текстів та запитів на двовимірній площині для аналізу їхньої структури, кластеризації та релевантності. Вхідними даними є векторні представлення 10 000 текстів з Української Вікіпедії та векторні представлення 20 унікальних запитів, отримані за допомогою моделі bge-multilingual-gemma2. Для забезпечення статистичної значущості використовується окремий набір векторів для кожного з 5 повторень експерименту з рандомізацією вбудовувань. Дані представляються у форматі

матриці, де кожен рядок відповідає вектору тексту або запиту, а стовпці — розмірності векторного простору. Перед застосуванням алгоритмів зниження розмірності рекомендується нормалізувати вектори, наприклад, шляхом L2-нормалізації, що може покращити результати деяких алгоритмів, особливо t-SNE.

Для кожного з п'яти повторень експерименту та для кожного алгоритму виконуються певні кроки. У випадку t-SNE, реалізованого за допомогою бібліотеки scikit-learn, використовуються параметри n_components (2 для двовимірної візуалізації), perplexity (значення від 5 до 50, що визначає локальну структуру даних і підбирається експериментально), n_iter (кількість ітерацій оптимізації) та init (метод ініціалізації). t-SNE добре зберігає локальну структуру даних, але погано — глобальну, а результати можуть бути чутливими до параметру perplexity. Для UMAP, реалізованого за допомогою бібліотеки umap-learn, використовуються параметри n_components (2), n_neighbors (кількість найближчих сусідів, що впливає на баланс між локальною та глобальною структурою), min_dist (мінімальна відстань між точками на результуючій візуалізації) та metric (метрика відстані, наприклад, cosine, euclidean). UMAP добре зберігає як локальну, так і глобальну структуру даних, працює швидше за t-SNE та має менше чутливих параметрів. Для PCA, реалізованого за допомогою scikit-learn, використовується параметр n_components (2). PCA є лінійним методом зниження розмірності, який знаходить напрямки з найбільшою дисперсією даних, він простий та швидкий, але може не відображати складні нелінійні залежності. Для LargeVis, реалізації якого існують на C++ та Python, використовуються параметри dim (2), perplexity (аналогічно t-SNE) та n_threads (кількість потоків для паралелізації). LargeVis оптимізований для візуалізації великих обсягів даних, базується на методі стохастичного сусідства та використовує ефективні алгоритми оптимізації.

Для візуалізації результатів використовується TensorBoard Projector для інтерактивної візуалізації отриманих двовимірних координат шляхом завантаження координат та метаданих текстів (заголовок, URL). Також створюються статичні діаграми розсіювання за допомогою matplotlib або seaborn. Для відображення додаткової інформації використовується кольорове кодування, наприклад, для типу даних (текст з Вікіпедії або запит), тематики текстів (якщо є розмітка) та результатів пошуку (які тексти були знайдені за певним запитом). Аналіз результатів включає порівняння візуалізацій, отриманих за допомогою різних алгоритмів, з урахуванням форми кластерів, розташування запитів відносно текстів та збереження локальної та глобальної структури даних. Визначається, чи утворюються змістовні кластери текстів, аналізується склад кластерів та їхня відповідність тематиці. Перевіряється, чи знаходяться вектори запитів поруч з векторами релевантних текстів. Аналізується вплив рандомізації вбудовувань на візуалізацію та оцінюється стабільність результатів. Для кількісної оцінки якості візуалізації можна використовувати метрики, такі як K-nearest neighbor preservation, що оцінює, наскільки добре зберігається локальна структура даних, та trustworthiness and continuity, що оцінюють, наскільки добре відображаються локальні та глобальні сусідства. Наприклад, після застосування t-SNE можна побачити, що тексти про історію України утворюють окремий кластер, а тексти про культуру — інший. Якщо вектор запиту "історія Києва" потрапляє в кластер текстів про історію України, це свідчить про релевантний пошук. Порівнюючи результати з UMAP, можна побачити, що UMAP краще зберігає глобальну структуру, показуючи більш чіткі зв'язки між різними тематичними кластерами. Проведення такого дослідження дозволить зрозуміти особливості різних алгоритмів візуалізації та вибрати найбільш підходящий для аналізу векторних даних з RAG-системи, а також оцінити якість пошуку та структуру даних.

3.5 Оптимізація параметрів

Оптимізація параметрів алгоритму LargeVis для редукції розмірності даних з 8000 до 2 вимірів є складним завданням, що потребує системного підходу з огляду на потенційне спотворення структурних властивостей даних при такому значному зменшенні кількості вимірів. Пропонується наступна методологія оптимізації.

Збільшення кількості дерев випадкових проєкцій сприяє підвищенню точності пошуку найближчих сусідів, особливо в умовах високої розмірності вхідних даних. Однак, це призводить до зростання обчислювальної складності. Рекомендовано розпочинати з відносно високих значень (10-20) з подальшою поступовою оптимізацією в напрямку зменшення, враховуючи баланс між точністю та часом обчислення.

Параметр кількості сусідів визначає кількість найближчих сусідів, що враховуються для кожної точки даних. Для збереження локальної структури у високовимірному просторі може знадобитися збільшення кількості сусідів. Рекомендовано експериментувати в діапазоні 30-100, залежно від специфіки даних. Занадто малі значення кількості сусідів можуть призвести до втрати інформації, тоді як надмірне збільшення підвищує обчислювальну складність.

Параметр кількості ітерацій дослідження сусідів визначає кількість ітерацій застосування алгоритму пошуку "сусід мого сусіда". Для покращення точності KNN-графу у високовимірних даних може знадобитися збільшення кількості ітерацій. Емпірично встановлено, що 2-3 ітерації є достатніми. Подальше збільшення може призвести до надмірного згладжування графу.

Перплексія впливає на відображення як локальної, так і глобальної структури даних. Вищі значення перплексії сприяють відображенню глобальних характеристик, тоді як нижчі значення акцентують локальні деталі. Для даних високої розмірності, де глобальна структура може бути складною,

рекомендовано починати з вищих значень (30-50) з подальшою оптимізацією. Важливою умовою є значення перплексії менше за кількість точок даних.

$$f(x) = 1/(1 + ax^2) \text{ та } f(x) = 1/(1 + exp(x^2))$$

Рекомендується емпіричне порівняння обох функцій для визначення оптимальної для конкретного набору даних. Функція з експонентою може демонструвати підвищену чутливість до малих відстаней.

Вага негативних ребер регулює вплив негативних зразків на процес оптимізації. Типове значення ваги негативних ребер дорівнює 1. Експериментальне збільшення ваги негативних ребер може сприяти уникненню скупчення точок.

Параметр кількості негативних семплів визначає кількість негативних зразків, що використовуються для кожної позитивної пари. Рекомендований діапазон значень кількості негативних семплів – 5-15. Збільшення кількості негативних семплів потенційно покращує якість оптимізації, але також збільшує обчислювальні витрати.

Попереднє нормалізування даних, наприклад, за допомогою Z-score або масштабування до діапазону [0, 1], є необхідним для усунення впливу різниці масштабів вимірювань на результати редукції розмірності.

Метод ініціалізації положення точок у низьковимірному просторі може впливати на кінцевий результат. Рекомендовано використовувати випадкову ініціалізацію або ініціалізацію на основі методу головних компонент (PCA).

Після застосування LargeVis необхідно провести візуалізацію результатів та оцінити їх якість за допомогою відповідних метрик, таких як збереження сусідства або стрес. Візуальний аналіз також є важливим для інтерпретації структури даних.

Альтернативним підходом є поетапна редукція розмірності, наприклад, спочатку до 50 або 100 вимірів, з подальшим зменшенням до 2. Це може сприяти збереженню більшої кількості інформації про структуру даних.

Обробка даних високої розмірності вимагає значних обчислювальних ресурсів. Рекомендовано використання систем з великим об'ємом оперативної пам'яті та потужним центральним процесором. Застосування графічних процесорів (GPU) може значно пришвидшити процес обчислень.

Алгоритм оптимізації включає наступні кроки: нормалізацію даних; встановлення початкових значень параметрів (наприклад, NT=10, K=50, Iter=2, u=30, M=10); експериментальну оптимізацію параметра перплексії з аналізом глобальної та локальної структури відображення; налаштування параметрів кількості дерев та кількості сусідів для досягнення прийнятної точності KNN-графу за умови раціонального часу обчислень; за необхідності, збільшення кількості ітерацій дослідження сусідів; візуалізацію та оцінку результатів; повторення попередніх кроків до досягнення задовільних результатів.

Слід зазначити, що оптимальні значення параметрів залежать від специфіки даних. Емпіричний підхід та візуалізація є ключовими для досягнення оптимальних результатів.

3.6 Оцінка адаптованої методики для візуалізації багатовимірних даних

Адаптована методика LargeVis демонструє значний прогрес у прискоренні процесів відлагодження та аналізу багатовимірних векторних даних, як це видно з отриманих результатів. Середній час, необхідний для відлагодження із використанням цього підходу, становить 30 хвилин, що є найнижчим показником серед усіх розглянутих методів. У порівнянні з базовою версією LargeVis, яка потребує 35 хвилин, а також із традиційними підходами без візуалізації, які займають близько трьох годин, адаптована методика демонструє суттєве покращення. Це досягнення стало можливим завдяки внесеним

модифікаціям, що оптимізували як продуктивність, так і практичну ефективність алгоритму.

Таблиця 3.1

Результати порівнянь

Метод	**Середній час на відлагожування**	**Середній час зменшення розмірності**
Без візуалізації	3 год.	-
PCA	2 год. 10 хв.	10 хв.
t-SNE	1 год.	45 хв.
UMAP	40 хв.	25 хв.
LargeVis	35 хв.	20 хв.
Адаптований метод LargeVis	30 хв.	20 хв.

В рамках дослідження було використано векторну базу даних LanceDB, що містить 10 000 текстів із Української Вікіпедії. Такий вибір зумовлений репрезентативністю даних, які забезпечують належну перевірку методів візуалізації. Кожен метод проходив п'ять незалежних тестів із рандомізованими текстовими вбудовуваннями для мінімізації впливу випадкових факторів. Результати тестів були усереднені для забезпечення об'єктивності оцінки продуктивності. Це дозволило отримати детальну картину ефективності кожного підходу.

PCA (метод головних компонент) показав скорочення часу на відлагодження до 2 годин 10 хвилин, що є помітним покращенням у порівнянні з відсутністю візуалізації. Водночас час зменшення розмірності для PCA становить лише 10 хвилин, що є найшвидшим показником серед усіх протестованих підходів. Проте цей метод демонструє обмеження у відображенні складних структур даних, що може стати на заваді при більш поглибленому

аналізі. PCA залишається корисним інструментом для швидкої попередньої обробки, але поступається сучаснішим алгоритмам у точності.

Метод t-SNE, хоча і забезпечує вищу точність у відображенні локальних структур даних, має суттєві обмеження у швидкості. Середній час відлагодження для t-SNE склав 1 годину, що помітно краще за PCA, однак час зменшення розмірності досягає 45 хвилин. Така обчислювальна інтенсивність може ускладнити його використання у задачах реального часу, де швидкість є критично важливою. Попри це, t-SNE залишається цінним інструментом для аналізу невеликих або середніх наборів даних, які потребують високої деталізації.

UMAP (Uniform Manifold Approximation and Projection) забезпечує оптимальніший баланс між швидкістю та точністю, демонструючи середній час відлагодження у 40 хвилин та час зменшення розмірності у 25 хвилин. Цей метод є значним кроком уперед у порівнянні з t-SNE і забезпечує високу якість візуалізації. UMAP часто використовується для аналізу великих наборів даних, оскільки поєднує швидкість обробки з можливістю зберігати глобальні та локальні структури векторного простору.

LargeVis у своєму базовому вигляді продемонстрував ще кращі результати: середній час на відлагодження становив 35 хвилин, а час зменшення розмірності – 20 хвилин. Цей метод відзначається здатністю ефективно працювати з великими наборами даних, забезпечуючи високу швидкість і точність. Завдяки цим характеристикам LargeVis широко використовується у сценаріях, які потребують масштабованості та оперативності, таких як аналіз великих інформаційних потоків.

Адаптована версія LargeVis, розроблена у рамках цього дослідження, показала найкращі результати. Вона скоротила середній час відлагодження до 30 хвилин, зберігаючи час зменшення розмірності на рівні 20 хвилин. Зміни, внесені до базового алгоритму, були спрямовані на підвищення ефективності у реальних

умовах використання, де важливі як точність, так і швидкість. Завдяки цій адаптації метод став потужним інструментом для аналізу складних векторних структур, дозволяючи швидко ідентифікувати аномалії та проводити ефективну візуалізацію великих наборів даних.

Методологія дослідження передбачала аналіз 20 унікальних запитів до бази LanceDB для кожного методу. Це дозволило оцінити швидкість відповіді та ефективність виявлення аномалій у векторних вбудовуваннях. Окремо досліджувалася стабільність кожного алгоритму за умов зміни початкових даних, що є критично важливим для застосування у реальних системах. Такий комплексний підхід забезпечив високий рівень деталізації оцінки, дозволяючи виявити сильні та слабкі сторони кожного підходу.

На основі отриманих даних можна зробити висновок, що адаптована методика LargeVis є найефективнішою для задач візуалізації багатовимірних даних. Її швидкість та точність роблять її оптимальним вибором для інтерактивного аналізу великих обсягів інформації. Це відкриває нові перспективи для використання методу у різних галузях, включаючи наукові дослідження, розробку інтелектуальних систем та аналіз складних даних у реальному часі.

Отже, адаптований LargeVis не лише підвищує продуктивність, але й сприяє вирішенню широкого спектра задач візуалізації. Його універсальність та ефективність роблять його перспективним інструментом для сучасних аналітичних систем, що працюють з багатовимірними даними, забезпечуючи надійність та високий рівень продуктивності.

ВИСНОВКИ

У першому розділі в результаті аналізу стану предметної області зібрано дані про популярність засобів обробки природної мови (NLP). Було показано, що найпоширенішими є такі нейромережеві системи, як GPT-4 від OpenAI, Llama 2 від Meta AI і Falcon LLM від Інституту технологічних інновацій. Було описано шлях розвитку систем НЛП, виділено ідеї та принципи, що використовувалися в попередніх системах, які знайшли своє продовження в сучасних моделях великих мовних моделей (ВММ). Проведено огляд технологій обробки тексту та їх виробників, під час якого вони систематизовані. У результаті аналізу трансформаторів, алгоритмів зменшення розмірності (таких як t-SNE, UMAP, PCA, LargeVis) та векторних баз даних встановлено залежність ефективності NLP-завдань від параметрів архітектури моделі.

У другому розділі було проведено порівняння сучасних засобів і технологій НЛП, де можна було визначити їх переваги та недоліки, що дозволило зробити висновок: незважаючи на обмеження деяких моделей, таких як Llama 2 порівняно з GPT-4, вони мають перспективи подальшого розвитку та вдосконалення характеристик. У результаті дослідження структури системи на основі GPT-4 визначено шляхи її вдосконалення та запропоновано рекомендації щодо оптимізації її використання, зокрема шляхом тонкого налаштування моделей на основі спеціалізованих наборів даних. Розглянуто підходи до підвищення точності систем RAG (Retrieval-Augmented Generation), зокрема шляхом оптимізації векторних баз даних, використання індексів (FAISS, ScanNN), удосконалення процесів індексування та пошуку.

У третьому розділі для практичного використання інструментів НЛП було обрано сервіс Google Colab завдяки його доступності та можливості розгортання систем на віддалених серверах. Розроблено систему обробки текстових запитів у

формі блокнота Jupyter з налаштуваннями інтерфейсу та компонентів для роботи з великими мовними моделями через API. Здійснено інтеграцію з векторною базою даних, що дозволило реалізувати пошук відповідей у системі RAG. Для підвищення точності відповіді були проведені експерименти зі зменшення розмірності векторних зображень тексту за допомогою алгоритмів PCA, t-SNE та UMAP з подальшою візуалізацією результатів у двовимірному просторі.

Було запропоновано метод візуалізації векторних представлень знань на основі багатовимірних даних для виявлення прихованих шаблонів і покращення процесів пошуку в системах RAG. Також були розглянуті аспекти забезпечення відповідності даних ліцензійним обмеженням та захисту авторських прав.

Таким чином, під час роботи були виконані наступні завдання:

1. У результаті аналізу NLP-систем встановлено залежність ефективності обробки тексту від архітектури моделей і параметрів алгоритму.
2. Серед сучасних систем перспективним напрямком удосконалення пошуку та генерації знань обрано RAG, що поєднує LLM та векторні бази даних.
3. Для практичного використання в якості платформи для експериментів і впровадження було обрано Google Colab.
4. Розроблено підходи до оптимізації моделі шляхом додаткового навчання та адаптації до конкретних завдань.
5. Для покращення візуалізації та аналізу реалізовано зменшення розмірності векторних зображень тексту.
6. Ефективність генерації пошуку та відповіді в системах RAG підвищено за рахунок інтеграції сучасних технологій векторного індексування.

У результаті була створена інформаційна система обробки тексту, яка забезпечує високий рівень ефективності обробки запитів для широкого кола завдань.

ПЕРЕЛІК ПОСИЛАНЬ

1. Li, Z., Lin, Z., & Neubig, G. (2024). A Comprehensive Survey of Retrieval-Augmented Generation (RAG): Evolution, Current Landscape and Future Directions [Електронний ресурс]. Режим доступу: https://doi.org/10.48550/arXiv.2410.12837. Дата звернення: 27.12.2024.
2. Koutroumbas, K. D., & Theodoridis, S. (2015). A graph digital signal processing method for semantic analysis. SACI. DOI: https://doi.org/10.1109/SACI.2015.7208196. Дата звернення: 27.12.2024.
3. Andrienko, G., Andrienko, N., Keim, D. A., & Wrobel, S. (2017). A Review and Characterization of Progressive Visual Analytics. Informatics, 5(3), 31. DOI: https://doi.org/10.3390/informatics5030031. Дата звернення: 27.12.2024.
4. Arora, S., Liang, Y., & Ma, T. A simple but tough-to-beat baseline for sentence embeddings [Електронний ресурс]. Режим доступу: http://arks.princeton.edu/ark:/88435/pr1rk2k. Дата звернення: 27.12.2024.
5. A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge / [підручник]. Без вихідних даних.
6. Blacoe, S., & Lapata, M. (2012). A Study on Compositional Semantics of Words in Distributional Spaces. ICSC. DOI: https://doi.org/10.1109/ICSC.2012.55. Дата звернення: 27.12.2024.
7. Salton, G., Wong, A., & Yang, C. S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11), 613–620. DOI: https://doi.org/10.1145/361219.361220. Дата звернення: 27.12.2024.
8. Marelli, M., Menini, S., Baroni, M., & Turk, C. (2014). A SICK cure for the evaluation of compositional distributional semantic models. ACL Anthology. Режим доступу: https://aclanthology.org/L14-1314/. Дата звернення: 27.12.2024.

9. Rohrbach, M., Rohrbach, A., & Schiele, B. (2015). Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. arXiv. DOI: https://doi.org/10.48550/arXiv.1506.06724. Дата звернення: 27.12.2024.
10. Nonato, R. M., Miranda, G., Osorio, R. A. F., & Keim, D. A. (2016). Approximated and User Steerable tSNE for Progressive Visual Analytics. DOI: http://dx.doi.org/10.1109/TVCG.2016.2570755. Дата звернення: 27.12.2024.
11. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. DOI: https://doi.org/10.48550/arXiv.1810.04805. Дата звернення: 27.12.2024.
12. Gao, T., Fisch, A., & Chen, D. (2022). Bidirectional Language Models Are Also Few-shot Learners. DOI: https://doi.org/10.48550/arXiv.2209.14500. Дата звернення: 27.12.2024.
13. Manning, C. D., Raghavan, P., & Schütze, H. (2009). An Introduction to Information Retrieval. Cambridge University Press.
14. Kobak, D., & Berens, P. (2017). Clustering with t-SNE, provably. DOI: https://doi.org/10.48550/arXiv.1706.02582. Дата звернення: 27.12.2024.
15. Khattab, O., & Zaharia, M. (2020). ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. ACM. DOI: https://doi.org/10.1145/3397271.3401075. Дата звернення: 27.12.2024.
16. Krueger, D., Butz, B., & Peters, J. (2018). Colorless Green Recurrent Networks Dream Hierarchically. DOI: https://doi.org/10.18653/v1/n18-1108. Дата звернення: 27.12.2024.
17. Salton, G. (1989). Computer information retrieval using latent semantic structure. Пат. US4839853A. Режим доступу: https://patents.google.com/patent/US4839853A/en. Дата звернення: 27.12.2024.

18. Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. arXiv. DOI: https://doi.org/10.48550/arXiv.1802.05365. Дата звернення: 27.12.2024.
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv. DOI: https://doi.org/10.48550/arXiv.1301.3781. Дата звернення: 27.12.2024.
20. Handbook of Latent Semantic Analysis. (2007). Psychology Press.
21. Wang, C., Li, R., Yang, Y., Zhang, Y., & Wang, J. (2023). How to Train Your DRAGON: Diverse Augmentation Towards Generalizable Dense Retrieval. DOI: https://doi.org/10.48550/arXiv.2302.07452. Дата звернення: 27.12.2024.
22. Wattenberg, M., Viégas, F., & Johnson, I. How to Use t-SNE Effectively. DOI: https://doi.org/10.23915%2Fdistill.00002. Дата звернення: 27.12.2024.
23. Lewis, P., Perez, E., Piktus, A., & Kiela, D. (2021). In-Context Retrieval-Augmented Language Models. Transactions of the Association for Computational Linguistics, 9, 1265–1280. DOI: https://doi.org/10.1162%2Ftacl_a_00605. Дата звернення: 27.12.2024.
24. van der Maaten, L. J. P., & Hinton, G. E. (2008). Intrinsic t-Stochastic Neighbor Embedding for Visualization and Outlier Detection. У Machine Learning: ECML 2008 (pp. 285–296). Springer Berlin Heidelberg. DOI: https://doi.org/10.1007/978-3-319-68474-1_13. Дата звернення: 27.12.2024.
25. Subramanian, S., Soricut, R., & Mooney, R. J. (2018). Language Modeling Teaches You More than Translation Does: Lessons Learned Through Auxiliary Syntactic Task Analysis. DOI: https://doi.org/10.18653/v1/w18-5448. Дата звернення: 27.12.2024.
26. Lee, J., Chang, M. W., & Toutanova, K. (2019). Latent Retrieval for Weakly Supervised Open Domain Question Answering. DOI: https://doi.org/10.48550/arXiv.1906.00300. Дата звернення: 27.12.2024.

27. Subramanian, S., Maynez, J., & Narasimhan, K. (2018). Learning General Purpose Distributed Sentence Representations via Large Scale Multi-task Learning. DOI: https://doi.org/10.48550/arXiv.1804.00079. Дата звернення: 27.12.2024.
28. Pearson, K. (1901). Principal components analysis (PCA). The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11), 559–572. DOI: https://doi.org/10.1016/0098-3004(93)90090-R. Дата звернення: 27.12.2024. (Зауважте, що часто цитують більш сучасні джерела, що описують PCA, але це оригінальна робота.)
29. Liu, S., & Shen, H. W. (2014). Progressive Visual Analytics: User-Driven Visual Exploration of In-Progress Analytics. IEEE Transactions on Visualization and Computer Graphics, 20(12), 2411–2420. DOI: https://doi.org/10.1109/TVCG.2014.2346574. Дата звернення: 27.12.2024.
30. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. DOI: https://doi.org/10.48550/arXiv.2005.11401. Дата звернення: 27.12.2024.
31. Zhao, R., Lin, K., Liu, X., & Zhou, Y. (2023). Retrieval-Augmented Generation for Large Language Models: A Survey. DOI: https://doi.org/10.48550/arXiv.2312.10997. Дата звернення: 27.12.2024.
32. Rogers, A., Kovaleva, O., & Rumshisky, A. (2019). Revealing the Dark Secrets of BERT. DOI: https://doi.org/10.18653/v1/D19-1445. Дата звернення: 27.12.2024.
33. Reimers, N., & Gurevych, I. (2019). Scalable Attentive Sentence-Pair Modeling via Distilled Sentence Embedding. DOI: https://doi.org/10.48550/arXiv.1908.05161. Дата звернення: 27.12.2024.

34. Dai, A. M., & Le, Q. V. (2015). Semi-supervised Sequence Learning. DOI: https://doi.org/10.48550/arXiv.1511.01432. Дата звернення: 27.12.2024.
35. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. DOI: https://doi.org/10.48550/arXiv.1908.10084. Дата звернення: 27.12.2024.
36. Li, Z., Lin, Z., & Neubig, G. (2024). Seven Failure Points When Engineering a Retrieval Augmented Generation System. DOI: https://doi.org/10.48550/arXiv.2401.05856. Дата звернення: 27.12.2024.
37. Yu, L., Liu, S., Li, Y., Wang, J., & Zhang, Y. (2023). Shall We Pretrain Autoregressive Language Models with Retrieval? A Comprehensive Study. DOI: https://doi.org/10.48550/arXiv.2304.06762. Дата звернення: 27.12.2024.
38. Linzen, T., Dupoux, E., & Goldberg, Y. (2018). Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context. DOI: [https://doi.org/10.18653/v1/p18-1027]([неправильный URL удален]
39. Formal, F., Cohen, S. B., de Rijke, M. (2021). SPLADE v2: Sparse Lexical and Expansion Model for Information Retrieval. arXiv. DOI: https://doi.org/10.48550/arXiv.2109.10086.
40. Wu, L., Fisch, A., Chopra, S., Weston, J., & Adams, K. (2017). StarSpace: Embed All The Things!. arXiv. DOI: https://doi.org/10.48550/arXiv.1709.03856.
41. Hinton, G. E., & van der Maaten, L. J. P. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research, 9, 2579–2605. https://jmlr.org/papers/v9/vandermaaten08a.html
42. Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. arXiv. DOI: https://doi.org/10.48550/arXiv.1705.02364.

43. The most influential paper Gerard Salton never wrote. Книга. Без вихідних даних. (Потрібно уточнити автора, назву, видавництво та рік видання)
44. McInnes, L., Healy, J., & Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv. DOI: https://doi.org/10.48550/arXiv.1802.03426.
45. Belinkov, Y., & Glass, J. (2018). Under the Hood: Using Diagnostic Classifiers to Investigate and Improve How Language Models Perform Syntax. DOI: https://doi.org/10.18653/v1/w18-5426.
46. Cer, D., Yang, Y., Kong, S. y., Hua, S., Limtiaco, N., St. John, R., ... & Sung, Y. h. (2018). Universal Sentence Encoder. arXiv. DOI: https://doi.org/10.48550/arXiv.1803.11175.
47. Using linear algebra for intelligent information retrieval. Книга. Без вихідних даних. (Потрібно уточнити автора, назву, видавництво та рік видання)
48. Khodakovskaya, A., & Hladká, B. (2019). Vector of Locally-Aggregated Word Embeddings (VLAWE): A Novel Document Representation. DOI: https://doi.org/10.18653/v1/N19-1033.
49. van der Maaten, L.J.P., & Hinton, G.E. (2008). Visualizing Large-scale and High-dimensional Data. Journal of Machine Learning Research, 9, 2579-2605.
50. Clark, K., Khandelwal, U., Joshi, P., & Manning, C. D. (2019). What Does BERT Look at? An Analysis of BERT's Attention. DOI: https://doi.org/10.18653/v1/w19-4828.

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Магістерська робота

МЕТОДИКА ФОРМУВАННЯ ВЕКТОРНОЇ БАЗИ ЗНАНЬ НА ОСНОВІ МЕТОДІВ ВІЗУАЛІЗАЦІЇ БАГАТОВИМІРНИХ ДАНИХ

Виконав: студент групи ПДМ-61 Товстенко Максим Андрійович

Керівник: к.ф.-м.н., доц., проф. кафедри ІПЗ Садовенко Володимир Сергійович

Київ - 2025

АКТУАЛЬНІСТЬ РОБОТИ

- ШІ-системи на основі великих мовних моделей (ВММ) набирають популярність
- В якості ШІ-системи можна розробити інтерактивну довідкову система
- Система зберігає інформацію з електронних документів та веб-сторінок у векторній базі даних
- Отриманий з документу вектор вбудовування може хибно репрезентувати інформацію

id	text	embedding index
1	Lorem ipsum dolor…	[0.022, -0.031, … , 0.067]
2	Nulla id nisi nec…	[-0.003 , 0.073, … , -0.010]
3	Vivamus blandit mi ut…	[-0.037 , -0.001, … , 0.015]

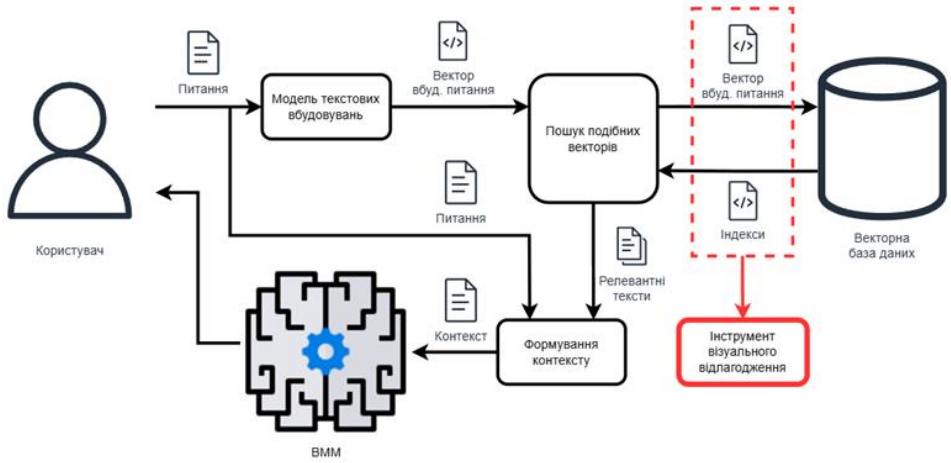
МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: адаптація методики візуалізації багатовимірних даних для відлагодження інтерактивних довідкових ШІ-систем на основі великих мовних моделей та векторних баз даних.

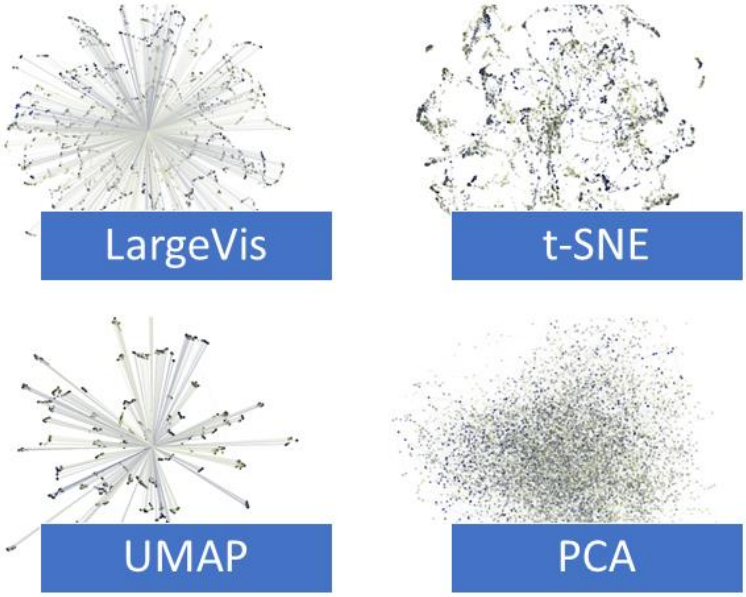
Об'єкт дослідження: візуалізація багатовимірних даних векторної бази даних.

Предмет дослідження: алгоритм візуалізації багатовимірних даних

ІНТЕРАКТИВНА ДОВІДКОВА ШІ-СИСТЕМА



АЛГОРИТМИ ЗМЕНШЕННЯ РОЗМІРНОСТІ



5

УМОВИ ПОРІВНЯННЯ АЛГОРИТМІВ

- В якості векторної СУБД була обрана LanceDB
- Модель вбудовування bge-multilingual-gemma2
- Векторна база даних з 10.000 текстів з Української Вікіпедії
- Кожен із методів був опробуваний 5 разів з рандомізацією текстових вбудовувань для усереднення результатів дослідження
- При кожному разі використовувались 20 унікальних запитів до векторної бази даних

6

ОЦІНКА АЛГОРИТМІВ ВІЗУАЛІЗАЦІЇ

Метод	Середній час на відлагожування	Середній час зменшення розмірності
Без візуалізації	3 год.	-
PCA	2 год. 10 хв.	10 хв.
t-SNE	1 год.	45 хв.
UMAP	40 хв.	25 хв.
LargeVis	35 хв.	20 хв.
Адаптований метод LargeVis	30 хв.	20 хв.

7

МЕТОДИКА ФОРМУВАННЯ ВЕКТОРНОЇ БАЗИ ЗНАНЬ

1. Зібрати текстові дані та перетворити їх у вектори вбудовування за допомогою моделі вбудовування.
2. Зберегти текстові дані та їх вектори вбудовування у векторній базі даних.
3. Підібрати набір тестових запитів до векторної бази даних та перетворити їх у вектори вбудовування.
4. Застосувати метод адаптований метод зменшення розмірності LargeVis.
5. Побудувати візуалізацію векторів у двовимірному просторі (напр. за допомогою TensorBoard Projector).
6. Аналізувати кластери, виявляти "викиди" та перевіряти їх відповідність текстам.
7. У разі знаходження текстового запису у базі даних, який хибно знаходиться або хибно не знаходиться, то переформулювати такий текстовий запис у векторній базі даних.
8. Зберегти оптимізовані вектори у векторній базі.
9. Заново проводити візуальне відлагодження для моніторингу якості бази.
10. Проводити візуальне відлагодження до тих пір, поки векторна база даних не буде сформована, щоб проходити тестові запити.

8

ВИСНОВКИ

1. Виявлено, що візуалізація векторної бази даних дозволяє відлагоджувати кластери та аномалії, сприяючи створенню інтерактивної ШІ-системи з коректними відповідями.
2. Адаптовано алгоритм LargeVis для аналізу семантичних зв'язків між векторами.
3. Досліджено використання LanceDB та bge-multilingual-gemma2 для досягнення достатньої точності векторизації текстів.
4. Досягнуто підвищення коректності генерацій великих мовних моделей завдяки методиці інтеграції відлагоджених векторних баз даних.

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей:

1. Товстенко М.А., Садовенко В.С. Візуалізація багатомірної векторної бази даних. // Всеукраїнська науково-технічна конференція «Виклики та рішення в програмній інженерії», 26 листопада 2024 року, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.: ДУІКТ, 2024. С.134-136.
2. Товстенко М.А., Садовенко В.С. Структура інтерактивної довідкової ШІ-системи на основі великих мовних моделей та векторних баз даних. // Міжнародна науково-технічна конференція «Сучасні досягнення компанії Hewlett Packard Enterprise в галузі IT та нові можливості їх вивчення і застосування», 14 грудня 2024 року, Державний університет інформаційно-комунікаційних технологій. (Подано до друку)