Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Покращення «cold-start» пошуку товарів в e-commerce
за допомогою семантичних ідентифікаторів
(тема)

Виконав:
здобувач другого року навчання,
групи СШІм-24-2
Олексій СЕНЮШКОВИЧ
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)
Освітня програма Системи штучного інтелекту
(повна назва освітньої програми)
Керівник проф. Ваган ТЕРЗІЯН
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Лариса ЧАЛА
(підпис) (власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти другий (магістерський)

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

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

Освітня програма Системи штучного інтелекту
(повна назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Сенюшковичу Олексію Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Покращення «cold-start» пошуку товарів в e-commerce за допомогою семантичних ідентифікаторів

затверджена наказом університету від 6 квітня 20 26 р. № 269Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 21 травня 20 26 р.

3. Вихідні дані до роботи наукові статті і журнали, дані з мережі Інтернет, документації Transformers, документація PyTorch, документація Python

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної галузі

2) Проєктування системи

3) Програмна реалізація

4) Аналіз результатів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	06.04.2026	виконано
2	Проведення аналізу теми	06.04.2026	виконано
3	Визначення основних напрямків дослідження	10.04.2026	виконано
4	Огляд існуючих підходів в рекомендаційних	11.04.2026	виконано
	системах		
5	Проєктування системи	12.04.2026 – 15.04.2026	виконано
6	Вибір технологій	16.04.2026	виконано
7	Програмна реалізація	17.04.2026 – 10.05.2026	виконано
8	Аналіз результатів	12.05.2026 – 14.05.2026	виконано
9	Оформлення пояснювальної записки	15.05.2026 – 16.05.2026	виконано
10	Нормоконтроль	17.05.2026	виконано
11	Захист кваліфікаційної роботи	21.05.2026	

Дата видачі завдання 6 квітня 20 26 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Ваган ТЕРЗІЯН
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 60 с., 8 рис., 8 табл., 1 дод., 22 джерела.

ЕЛЕКТРОННА КОМЕРЦІЯ, ЕМБЕДДИНГИ, НЕЙРОННІ МЕРЕЖІ, РЕКОМЕНДАЦІЙНІ СИСТЕМИ, СЕМАНТИЧНІ ІДЕНТИФІКАТОРИ, E-COMMERCE, RANKING, RECOMMENDER SYSTEMS, RETRIEVAL, SEMANTIC ID, TWO-TOWER.

Об’єкт дослідження – рекомендаційні системи в електронній комерції.

Предмет дослідження – методи побудови рекомендаційних систем з використанням семантичних ідентифікаторів товарів, сформованих на основі ембеддингів та нейронних мереж.

Мета роботи – дослідження та реалізація підходу до побудови рекомендаційної системи для e-commerce із використанням semantic ID з метою покращення якості рекомендацій та часткового подолання проблеми cold start.

Дослідження генеративного пошуку на основі Semantic-ID для системи рекомендацій в електронній комерції, зосереджене на режимі холодного старту (cold-start) та протестоване відносно базової моделі на каталогах промислового й публічного корпусів для характеристики того, де компактні дискретні представлення елементів допомагають, а де виявляються неефективними.

ABSTRACT

Master’s thesis contains: 60 pp., 8 fig., 8 tabl., 1 ann., 22 references.

EMBEDDINGS, E-COMMERCE, NEURAL NETWORKS, SEMANTIC ID, SEMANTIC IDENTIFIERS, RANKING, RECOMMENDER SYSTEMS, RETRIEVAL, TWO-TOWER.

The object of the research is recommender systems in e-commerce.

The subject of the research is methods for building recommender systems using semantic product identifiers generated using embeddings and neural networks.

The purpose of the work is to study and implement an approach to building an e-commerce recommender system using Semantic ID to improve recommendation quality and partially address the cold-start problem.

The study investigates Semantic-ID-based generative retrieval for an e-commerce recommender system, focusing on the cold-start setting. The approach is evaluated against a baseline model on industrial and public catalogue corpora in order to characterize where compact discrete item representations are beneficial and where they prove ineffective.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8
Вступ 9
1 Аналіз предметної галузі 10
1.1 Опис предметної галузі 10
1.2 Огляд існуючих підходів 10
1.2.1 Колаборативна фільтрація 11
1.2.2 Фільтрація за вмістом 13
1.3 Обмеження існуючих підходів 14
1.4 Постановка задачі дослідження 16
2 Проєктування системи 17
2.1 Опис системи 17
2.2 Вибір технологій 19
3 Програмна реалізація 23
3.1 Векторизація товарів 23
3.1.1 Побудова каталогу 23
3.1.2 Збагачення таксономії 23
3.1.3 Контентний ембеддинг 24
3.1.4 Вирівнювання та нормалізація 25
3.1.5 Центрування як методологічне вдосконалення 25
3.2 Варіаційний автоенкодер із залишковою квантизацією 26
3.2.1 Архітектура 26
3.2.2 Ініціалізація 27
3.2.3 Цільова функція навчання 28
3.2.4 Механізми стабільності 29
3.2.4 Від навченої моделі до Semantic ID 30
3.3 Генеративна рекомендаційна система 30
3.3.1 Представлення послідовностей 31
3.3.2 Модель 31

3.3.3 Обмежене декодування 32
3.3.4 Оцінювання 33
4 Результати експериментів 34
4.1 Конфігурація експериментів 34
4.1.1 Датасети 34
4.1.2 Вибірки 35
4.1.3 Базова модель 35
4.1.4 Метрики 36
4.1.5 Навчання та апаратне забезпечення 37
4.2 Основні результати 37
4.3 Аналіз метрик 38
4.3.1 Більша ширина променевого пошуку 39
4.3.2 Міжкорпусне порівняння 40
4.3.3 Висновок 41
4.4 Геометрія контентних ембеддингів 42
4.4.1 Діагностика 42
4.4.2 Процедура центрування 43
4.4.3 Вплив на пошук 44
4.4.4 Висновок 45
4.5 Приклади 45
4.5.1 У чому перемагає генеративна рекомендаційна система 50
4.5.2 У чому перемагає щільна базова модель 51
4.5.3 Синтез 52
Висновки 53
Перелік джерел посилання 56
Додаток А Відомість кваліфікаційної роботи 60

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AE – Autoencoder – автоенкодер;

BERT – Bidirectional Encoder Representations from Transformers – двоспрямовані кодувальні подання з трансформерів;

DCG – Discounted Cumulative Gain – дисконтоване сукупне здобуття;

GPU – Graphics Processing Unit – графічний процесор;

InfoNCE – Information Noise-Contrastive Estimation – оцінка контрастності інформації та шуму;

K-means – K-means clustering algorithm – алгоритм кластеризації К середніх;

L2 – L2 (Euclidean) norm – евклідова норма;

MSE – Mean Squared Error – середньоквадратична помилка;

NDCG – Normalised Discounted Cumulative Gain – нормалізоване дисконтоване сукупне здобуття;

RQ – Residual Quantisation – залишковий квантизатор;

RQ-VAE – Residual-Quantisation Variational Autoencoder – варіаційний автоенкодер з залишковою квантизацією;

Semantic ID – Semantic Identifier – семантичний ідентифікатор;

TIGER – Transformer Index for GEnerative Recommenders – трансформер індекс для генеративних рекомендаторів;

T5 – Text-to-Text Transfer Transformer – трансформер для переносу знань текст-в-текст;

VAE – Variational Autoencoder – варіаційний автоенкодер.

ВСТУП

За останні 20 років популярність e-commerce платформ виросла у десятки разів. Якщо на початку 21-го століття ринок роздрібної e-commerce торгівлі у США становив 26 млрд доларів США у 2000 році, то у 2024 році він уже досяг 1.2 трлн доларів США [1].

Велику, якщо не більшу, роль в цьому зіграв розвиток рекомендаційних систем. Рекомендаційна система – це система яка автоматично визначає, які товари, послуги або одиниці контенту можуть бути найбільш релевантними для конкретного користувача. Такі системи застосовуються не лише в e-commerce, а й у стримінгових сервісах, соціальних мережах, новинних платформах, музичних застосунках та онлайн-освіті.

У цій роботі досліджується підхід, у якому ембединги товарів використовуються для побудови дискретних семантичних ідентифікаторів (Semantic ID), що забезпечують компактне та змістовне представлення інформації про товар. Такий підхід дає змогу застосовувати трансформерні архітектури [5] в задачах рекомендації, оскільки вони ефективно працюють із послідовностями дискретних токенів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Опис предметної галузі

Електронна комерція є однією з найпоширеніших сфер застосування сучасних інформаційних технологій. Вона охоплює процеси продажу товарів і послуг через онлайн-платформи, де користувач отримує доступ до великої кількості товарів, їх характеристик, відгуків та можливості здійснення покупки без фізичної присутності в магазині. Однією з основних особливостей e-commerce є дуже великий обсяг доступних товарів, через що користувачу часто складно самостійно знайти найбільш релевантні для себе варіанти. Саме тому важливу роль у цій сфері відіграють рекомендаційні системи.

Рекомендаційні системи в e-commerce призначені для персоналізації взаємодії користувача з платформою та підвищення релевантності запропонованих товарів. Вони аналізують дані про товари, дії користувачів, їхні інтереси та попередню поведінку, щоб рекомендувати ті об’єкти, які з найбільшою ймовірністю зацікавлять користувача. Використання таких систем дозволяє не лише покращити досвід користувача, а й підвищити ключові бізнес-показники, зокрема ймовірність покупки, середній чек та рівень утримання клієнтів. У сучасних e-commerce системах рекомендаційні моделі є важливою складовою інтелектуальної обробки даних та підтримки прийняття рішень.

1.2 Огляд існуючих підходів

В рекомендаційних системах є два основні методи: колаборативна фільтрація (collaborative filtering), фільтрація вмісту (content filtering). Колаборативна фільтрація – це коли система формує рекомендації на основі взаємодій багатьох користувачів з товарами, наприклад переглядів, кліків,

покупок чи оцінок, і рекомендує те, що було релевантним для схожих користувачів [3]. Фільтрація вмісту, або фільтрація за контентом, це метод, коли система використовує характеристики самого товару, наприклад назву, опис, категорію, бренд, зображення або інші атрибути, і рекомендує об’єкти, схожі на ті, з якими користувач вже взаємодіяв раніше.

Сучасні рішення в великих компаніях, таких як Amazon, Netflix, Prom.ua, включають в собі комбінацію цих підходів, використовуючи як вміст документу, так і поведінкові дані користувачів. Часто така система будується у два етапи: retrieval і ranking. На етапі retrieval із великого каталогу швидко відбирається невелика множина кандидатів, а на етапі ranking ці кандидати вже більш точно оцінюються та впорядковуються. Для retrieval часто використовують document embeddings та архітектуру two-tower, у якій одна модель формує векторне представлення користувача або запиту, а інша – товару, що дозволяє ефективно шукати найбільш релевантні об’єкти у великому каталозі.

1.2.1 Колаборативна фільтрація

Колаборативна фільтрація (рисунок 1.1) є одним із базових підходів до побудови рекомендаційних систем і ґрунтується на аналізі взаємодій користувачів з об’єктами без необхідності явного використання змістовних характеристик самих об’єктів. Основна ідея цього підходу полягає в тому, що користувачі зі схожою поведінкою в минулому, ймовірно, матимуть схожі вподобання і в майбутньому. Аналогічно, об’єкти, з якими взаємодіють схожі групи користувачів, можуть вважатися подібними між собою. У межах колаборативної фільтрації зазвичай виділяють два основні напрями: user-based підхід, у якому рекомендації формуються на основі пошуку схожих користувачів, та item-based підхід, у якому визначається схожість між самими об’єктами. Як джерело даних при цьому можуть використовуватись явні взаємодії, наприклад оцінки товарів або фільмів, а

також неявні сигнали, такі як перегляди, кліки, додавання до кошика, час взаємодії чи факти придбання.

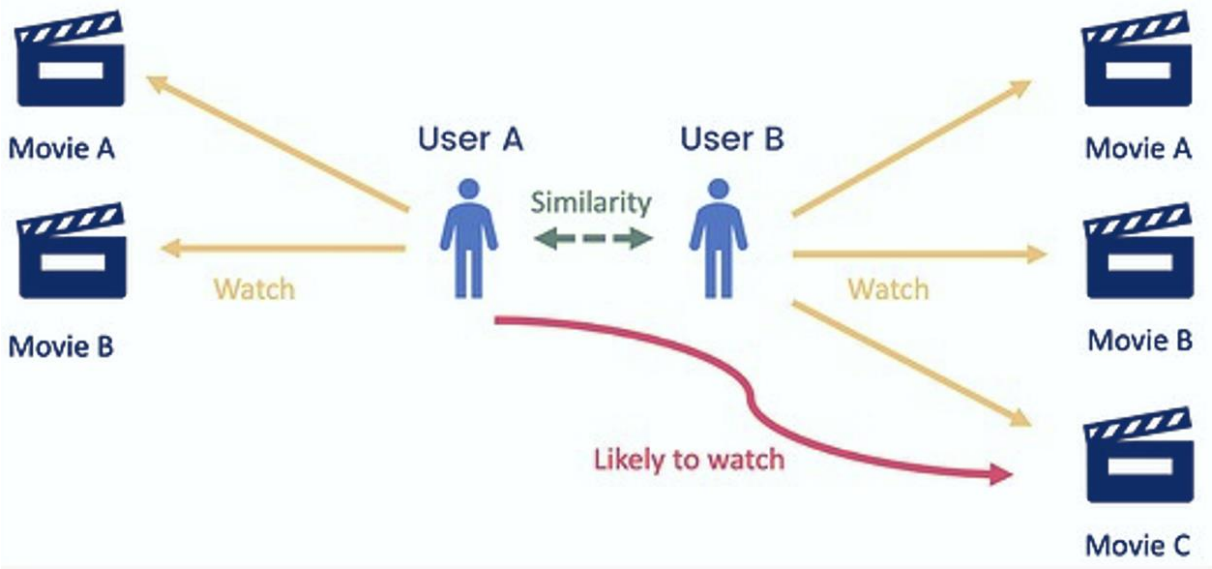


Рисунок 1.1 – Схема роботи колаборативної фільтрації

З технічної точки зору колаборативна фільтрація часто реалізується у вигляді матриці взаємодій користувачів та об’єктів, де рядки відповідають користувачам, а стовпці – товарам або іншим одиницям рекомендації. На основі цієї матриці можуть обчислюватися міри схожості, наприклад cosine similarity, Pearson correlation або інші метрики близькості, після чого система прогнозує ймовірність взаємодії користувача з новим об’єктом.

Однак у реальних задачах така матриця зазвичай є дуже розрідженою, оскільки кожен користувач взаємодіє лише з невеликою частиною доступних об’єктів. Тому на практиці часто застосовуються методи матричної факторизації (такі як SVD), зокрема розклад матриці взаємодій на латентні векторні представлення користувачів та об’єктів [4]. Це дозволяє виявляти приховані фактори вподобань і більш ефективно працювати з великими обсягами даних.

У сучасних системах цей підхід також може бути розширений за допомогою нейронних мереж, які навчаються будувати латентні

представлення користувачів та об’єктів і прогнозувати релевантність на основі їхньої взаємодії у спільному векторному просторі.

1.2.2 Фільтрація за вмістом

Фільтрація за вмістом (рисунок 1.2) є підходом до побудови рекомендаційних систем, у якому рекомендації формуються на основі характеристик самих об’єктів та історії взаємодії конкретного користувача з ними [7].

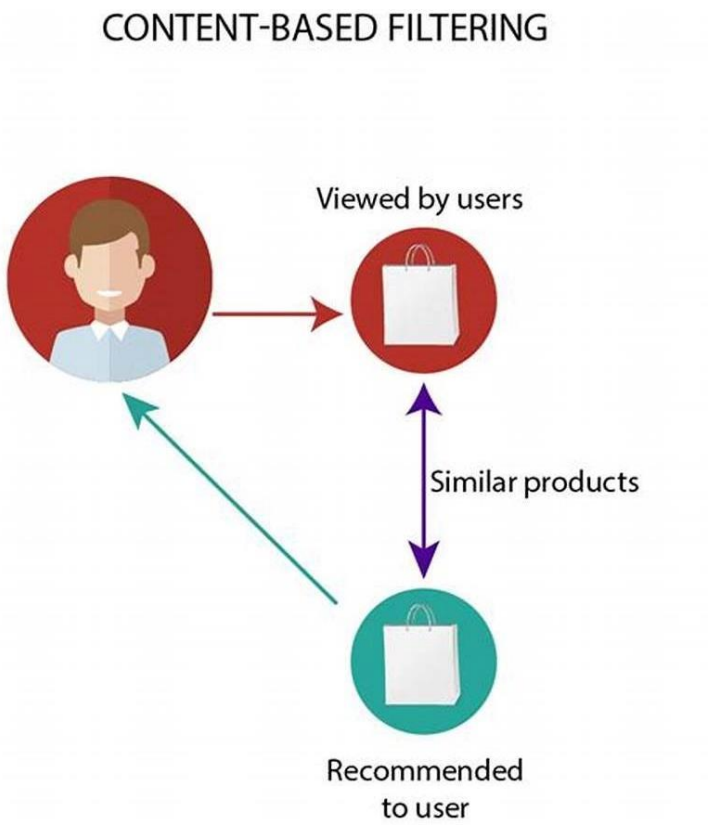


Рисунок 1.2 – Схема роботи фільтрації за вмістом

Основна ідея цього методу полягає в тому, що якщо користувач раніше взаємодіяв з певними товарами або одиницями контенту, то йому можуть бути рекомендовані об’єкти з подібними властивостями. На відміну

від колаборативної фільтрації, цей підхід не потребує даних про поведінку інших користувачів і зосереджується на аналізі ознак об’єктів, таких як назва, опис, категорія, бренд, ключові слова, числові характеристики, зображення або інші атрибути. Такий підхід особливо корисний у випадках, коли інформація про взаємодії користувачів є обмеженою, але доступний детальний опис самих об’єктів.

З технічної точки зору фільтрація за вмістом передбачає побудову векторного представлення об’єктів на основі їхніх ознак. Для текстових описів можуть використовуватися класичні методи, такі як bag-of-words або TF-IDF, а також сучасніші підходи на основі ембедингів, отриманих за допомогою нейронних мереж. Далі на основі об’єктів, з якими користувач вже взаємодіяв, формується профіль користувача, що відображає його інтереси у просторі ознак. Рекомендація нових об’єктів здійснюється шляхом обчислення схожості між профілем користувача та векторними представленнями об’єктів, наприклад за допомогою cosine similarity. У сучасних рекомендаційних системах цей підхід часто поєднується з глибокими моделями, які дозволяють враховувати не лише явні атрибути товарів, а й складні семантичні зв’язки між ними, зокрема через використання текстових, візуальних або мультимодальних ембедингів.

1.3 Обмеження існуючих підходів

Попри високу практичну цінність рекомендаційних систем, як колаборативна фільтрація, так і фільтрація за вмістом мають низку суттєвих обмежень. Однією з найбільш відомих проблем є cold start – ситуація, коли система має недостатньо даних для формування якісних рекомендацій. Ця проблема виникає у випадку нових користувачів, для яких ще відсутня історія взаємодій, а також у випадку нових товарів, які ще не отримали достатньо переглядів, кліків або покупок. Для колаборативної фільтрації ця проблема є особливо критичною, оскільки якість рекомендацій

безпосередньо залежить від наявності достатньої кількості поведінкових даних. У свою чергу, для методів фільтрації за вмістом проблема нових об’єктів є менш вираженою за наявності якісного опису товару, однак проблема нових користувачів усе одно залишається актуальною.

Ще одним суттєвим недоліком колаборативної фільтрації є проблема розрідженості даних та пов’язаний із нею ефект «довгого хвоста». У реальних e-commerce системах значна частина товарів взаємодіє лише з невеликою кількістю користувачів, тоді як основний обсяг взаємодій зосереджується навколо обмеженої кількості популярних позицій. У результаті алгоритм краще навчається на популярних товарах і має менше інформації про рідкісні або нові об’єкти, що призводить до менш якісних рекомендацій для товарів із «довгого хвоста». Це може зменшувати різноманітність рекомендацій і посилювати концентрацію уваги на вже популярних об’єктах. Крім того, колаборативна фільтрація схильна до так званого ефекту popularity bias, коли система частіше рекомендує ті товари, які й без того мають велику кількість взаємодій.

Фільтрація за вмістом також має власні обмеження. Основна проблема полягає в тому, що схожість за ознаками не завжди означає корисність рекомендації з практичної точки зору. Наприклад, якщо користувач уже придбав певний товар, повторна рекомендація дуже схожого або навіть того самого товару часто не несе цінності. У багатьох випадках для користувача більш корисною є рекомендація сумісного або доповнюючого товару, а не просто найбільш схожого. Наприклад, після придбання смартфона більш доречно рекомендувати чохол, захисне скло або зарядний пристрій, ніж інший майже ідентичний смартфон. Таким чином, фільтрація за вмістом добре працює для пошуку схожих об’єктів, але сама по собі недостатньо ефективно враховує відмінність між релевантністю, новизною, корисністю та комплементарністю рекомендацій.

Таким чином, жоден із базових підходів не є універсальним, а їх практичне використання потребує врахування специфіки предметної

області, типу даних, бізнес-цілей та очікувань користувача. Це зумовлює актуальність розробки гібридних рекомендаційних систем, здатних поєднувати переваги різних методів та частково компенсувати їхні недоліки.

1.4 Постановка задачі дослідження

У кваліфікаційній роботі необхідно реалізувати рекомендаційну систему, яка дозволятиме формувати рекомендації з використанням семантичних ідентифікаторів (Semantic ID [11]), побудованих за допомогою нейронної мережі, з метою покращення якості рекомендацій та часткового подолання проблеми cold start.

Для досягнення поставленої мети необхідно виконати такі завдання:

– проаналізувати існуючі системи і рішення з кодом у відкритому доступі;

– ознайомитись з науковими статтями по semantic ID;

– реалізувати код нейронної мережі для створення semantic ID та використання їх для рекомендацій;

– навчити модель нейронної мережі на існуючому корпусі товарів та історичних даних користувачів для покращення рекомендацій;

– проаналізувати результати і покращити систему, змінюючи параметри або підходи.

2 ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Опис системи

Система, що розробляється в цьому проєкті, включає:

– пайплайн підготовки даних (data preparation pipeline), який створює дедуплікований каталог товарів і хронологічно розділені послідовності взаємодій користувачів на основі необроблених експортів сесій та продуктів у форматі CSV;

– сервіс контентного ембеддингу (content embedding service), який перетворює текстові та категоріальні метадані кожного елемента каталогу в щільне векторне представлення за допомогою попередньо навченого текстового енкодера;

– варіаційний автоенкодер із залишковою квантизацією (RQ-VAE [3]), який стискає контентний ембеддинг кожного елемента в короткий кортеж дискретних токенів фіксованої довжини [4] – семантичний ідентифікатор елемента (Semantic ID);

– крок матеріалізації семантичних ідентифікаторів (Semantic-ID materialization), який кодує кожен елемент каталогу, детерміновано вирішує колізії кодових слів за допомогою стабільного суфіксного токена та експортує як таблицю кодів для кожного елемента, так і префіксне дерево (trie) всіх валідних ідентифікаторів;

– конструктор датасетів послідовностей (sequence dataset builder), який об'єднує потік взаємодій на рівні подій із семантичними ідентифікаторами елементів для створення навчальних прикладів;

– генеративний рекомендаційний трансформер (TIGER [2]), який авторегресійно прогнозує семантичний ідентифікатор наступного елемента на основі історії сесії користувача, при цьому променевий пошук (beam search) під час декодування обмежений префіксним деревом семантичних ідентифікаторів, що гарантує генерацію лише валідних елементів каталогу;

– щільну двовежеву базову модель пошуку (dense two-tower retrieval baseline) з порівнянною кількістю параметрів, яка навчена на тих самих послідовностях взаємодій із контрастивною цільовою функцією всередині батчу (in-batch contrastive objective) і використовується як система для порівняння;

– модуль оцінювання (evaluation module), який оцінює як генеративну, так і щільну моделі на одній і тій самій відкладеній хронологічній вибірці (held-out split), виводячи метрики пошуку top-K для повної тестової вибірки (test set), для цілей холодного старту (cold-start targets) та для зрізів лише за префіксами семантичних ідентифікаторів.

Архітектуру системи наведено на рисунку 2.1

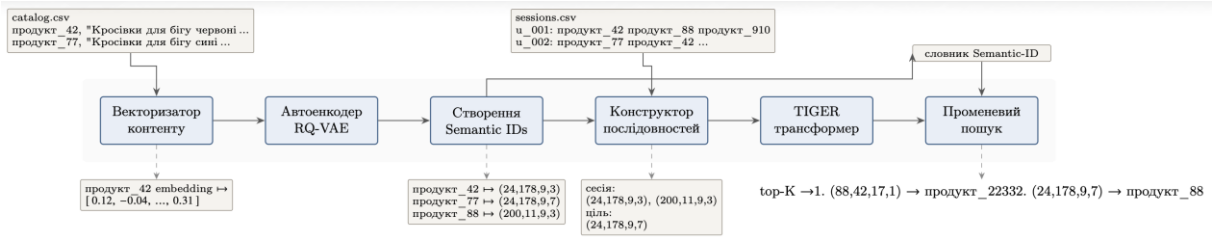


Рисунок 2.1 – Архітектура системи

У цій роботі для створення контентних ембеддингів використовуються два попередньо навчені текстові енкодери: внутрішній енкодер текстів товарів, що розгортається через MLflow для корпусу електронної комерції (e-commerce), та модель sentence-transformers/sentence-t5-base [7] з платформи HuggingFace для публічного корпусу Amazon Reviews 2023. Сам по собі RQ-VAE складається з трьох частин:

– енкодер прямого поширення (feed-forward encoder), який відображає вхідний ембеддинг у низьковимірний латентний простір;

– залишковий векторний квантизатор (residual vector quantizer), який послідовно квантизує латентний вектор та його залишки за допомогою кількох навчених словників (codebooks);

– симетричний декодер прямого поширення (feed-forward decoder), який відновлює вхідні дані з кумулятивного квантизованого латентного вектора.

Усі частини навчаються спільно з оптимізацією цільових функцій реконструкції та регуляризації квантизатор. Генеративна рекомендаційна система реалізована як авторегресійний трансформер типу decoder-only, словник якого є об'єднанням елементів словників (codebooks) RQ-VAE та невеликого набору структурних токенів; щільна базова модель використовує трансформерну «вежу запитів» (query tower) для елементів історії та MLP «вежу елементів» (item tower) для тих самих контентних ембеддингів.

2.2 Вибір технологій

Реалізація написана мовою програмування Python, організована як пакет зі структурою src/ під керуванням pyproject.toml і адмініструється за допомогою інструменту розв'язання залежностей uv (uv resolver). Попередньо навчені енкодери завантажуються через бібліотеки HuggingFace transformers та sentence-transformers, а також доповнюються внутрішнім завантажувачем на базі MLflow для пропрієтарного енкодера продуктів. Усі компоненти, що навчаються – RQ-VAE, трансформер TIGER та двовежева базова модель – реалізовані на PyTorch [20].

Числові артефакти (матриці ембеддингів, підрахунки частоти елементів, корисне навантаження префіксного дерева) серіалізуються за допомогою відображень пам'яті NumPy (NumPy memory-maps) та JSON. Запуски навчання відстежуються централізовано на власному (self-hosted) сервері трекінгу MLflow, який фіксує гіперпараметри, криві навчання та

валідації, а також метадані контрольних точок (checkpoints). Кожен етап пайплайну представлений як незалежний CLI-скрипт, що керується конфігураційним файлом JSON на основі датакласів (dataclasses) у директорії configs/. Це забезпечує слабку зв'язаність етапів через артефакти на диску та дозволяє повторно запускати окремі фази без переобчислення всіх минулих етапів. Можливість побудови каталогу, генерації контентних ембеддингів, навчання та аудиту RQ-VAE, матеріалізації семантичних ідентифікаторів, створення датасету послідовностей, навчання як генеративної, так і щільної моделей, а також їх оцінювання на однакових вибірках (splits) була реалізована та перевірена наскрізно (end-to-end) на двох корпусах.

TIGER («Transformer Index for GEnerative Recommenders») був представлений Google / Google DeepMind 8 травня 2023 року.Основна пропозиція полягає в заміні пошуку найближчих сусідів на основі ембеддингів – стандартного механізму в промислових рекомендаційних системах щонайменше з 2016 року – на авторегресійну генерацію ідентифікаторів елементів. Автори конструюють семантично значущі кортежі ідентифікаторів, які називаються Semantic ID (семантичні ідентифікатори), і навчають Transformer-модель типу sequence-to-sequence прогнозувати Semantic ID наступного елемента на основі історії користувача. Вони повідомляють про покращення результатів порівняно з сильними базовами моделями послідовних рекомендацій (sequential-recommendation baselines) на бенчмарках Amazon Reviews та демонструють кращу узагальнювальну здатність для умов холодного старту (cold-start generalization).

TIGER об’єднує два попередні напрями:

– автоенкодери із залишковою квантизацією для дискретного представлення. Компонент RQ-VAE запозичений майже безпосередньо з роботи Lee, Kim, Kim, Cho, and Han, «Autoregressive Image Generation using Residual Quantization», CVPR 2022 (Kakao Brain), де було показано, що

рекурсивне квантизування вектора за допомогою кількох словників (codebooks) забезпечує експоненціально більший ефективний кодовий простір. Це дозволяє трансформеру виконувати авторегресію над отриманим стеком кодів, використовуючи значно менше токенів на елемент, ніж у звичайному плосковому VQ-VAE;

– генеративний пошук за ідентифікаторами документів. У попередніх роботах із генеративного пошуку інформації (наприклад, DSI, NCI) трансформери вже навчали виводити ідентифікатори документів. Внесок TIGER полягає в застосуванні цього підходу до рекомендаційних систем (де немає природних ідентифікаторів, а каталоги налічують від мільйонів до мільярдів елементів) та у залученні конструкції Semantic-ID з RQ-VAE.

Архітектура містить чотири логічні етапи:

– кожен елемент каталогу пропускається через заморожений попередньо навчений текстовий енкодер для обробки його назви, опису та метаданих категорії, що дає один щільний вектор на елемент. В оригінальній статті використовується Sentence-T5, проте цей вибір позиціонується як взаємозамінний, що і було показано в цій кваліфікаційній роботі;

– побудова Semantic-ID за допомогою RQ-VAE. Невеликий варіаційний автоенкодер із залишковою квантизацією навчається на цих ембеддингах. Його «вузьким місцем» (bottleneck) є залишковий векторний квантизатор: латентний вектор квантизується за першим словником (codebook), залишок – за другим, наступний залишок – за третім і так далі для фіксованої кількості рівнів. Декодер відновлює вхідні дані з кумулятивного квантизованого латентного вектора. Після навчання кожен елемент каталогу кодується у фіксованої довжини кортеж цілочисельних індексів кодових слів – його Semantic ID. За конструкцією префікс іде від загального до конкретного (coarse-to-fine): елементи зі спільним першим кодом є семантично схожими на найзагальнішому знайменшому рівні, елементи зі спільними першими двома кодами – на детальнішому рівні й так

далі. Колізії кодових слів (коли різні елементи хешуються в один і той самий кортеж) усуваються за допомогою детермінованого суфіксного токена;

– потік взаємодій користувачів перевиражається в просторі Semantic-ID: сесія із відвідуваних товарів стає конкатенацією кодових кортежів цих елементів, а ціллю для прогнозування наступного елемента стає кодовий кортеж наступного товару. Стандартна Transformer-модель типу sequence-to-sequence навчається на цьому корпусі за допомогою крос-ентропії для наступних кодових токенів;

– декодування з обмеженим променевим пошуком. Під час роботи моделі в продакшені, трансформер виконує променевий пошук (beam search) за словником кодових токенів, але кожен крок розширення променя маскується попередньо обчисленим префіксним деревом (trie) валідних префіків Semantic ID, побудованим на основі каталогу. Оскільки кожен листок цього дерева відповідає наявному елементу, модель не може галюцінувати ідентифікаторами поза каталогом, а топ-K декодованих кортежів детерміновано відображаються назад у топ-K кандидатів-елементів.

Таким чином, механізмом, який відрізняє TIGER від щільної пошукової моделі (dense retriever), є авторегресійна факторизація ідентичності елемента за семантичними кодами від загального до конкретного плюс обмежене декодування – тобто генерація, а не звичайний пошук (look-up). Описані в статті переваги зосереджені на елементах холодного старту, оскільки раніше не бачений елемент все одно має валідний Semantic ID (присвоєний за допомогою RQ-VAE під час матеріалізації), який декодер може згенерувати, навіть якщо модель жодного разу не бачила його під час навчання.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Векторизація товарів

Пайплайн векторизації перетворює необроблені операційні логи в єдину щільну матрицю, в якій кожен рядок є контентним ембеддингом одного елемента каталогу. Ця матриця є вхідним інтерфейсом для кожного наступного етапу системи, тому її правильна та відтворювана побудова є передумовою для будь-якого змістовного порівняння на подальших етапах.

3.1.1 Побудова каталогу

Необробленими вхідними даними є плоскі CSV-експорти появ продуктів на рівні сесій, де кожен рядок фіксує одну появу продукту в межах сесії користувача. Багато продуктів з'являються тисячі разів протягом вікна експорту з незначними відхиленнями в метаданих назви, ціни або категорії між рядками. Перший етап дедуплікує ці появи за ідентифікатором продукту, вибирає канонічний запис для кожного елемента за правилом «перемагає найновіший» (most-recent-wins) і створює чистий каталог. Для підтримки корпусів масштабом у десятки мільйонів елементів цей етап реалізовано як один потоковий прохід по вихідному файлу, а не як з'єднання в пам'яті (in-memory join), і він видає аудиторський маніфест, що підсумовує кількість оброблених вихідних рядків і збережених унікальних елементів.

3.1.2 Збагачення таксономії

Необроблені записи містять лише непрозорі числові ідентифікатори категорій. Етап збагачення таксономії об'єднує їх із підготовленим словником категорій і розкладає кожну кінцеву категорію на її ієрархічний шлях. Отримані рівні ієрархії не використовуються самим енкодером;

замість цього вони підтримують подальші діагностичні аналізи – вимірювання того, чи мають елементи зі спільним префіксом Semantic-ID також спільний префікс категорії, оцінювання повноти (recall) для кожної категорії та декомпозицію зрізу холодного старту. Збагачення залишається незалежним від побудови каталогу, тому зміни в таксономії можна застосовувати без повторного потокового читання необроблених даних.

3.1.3 Контентний ембеддинг

Кожен елемент у каталозі кодується один раз за допомогою попередньо навченого текстового енкодера, що створює щільне представлення фіксованої розмірності. У цій роботі використовуються два енкодери, по одному на корпус. Для внутрішнього корпусу електронної комерції ми використовуємо пропрієтарний енкодер текстів товарів, який був попередньо навчений на словнику каталогу компанії; для корпусу Amazon Reviews ми використовуємо загальнодоступний енкодер речень, застосований до того самого текстового з'єднання, що й в оригінальній статті про TIGER (назва товару, магазин, основна категорія, повний ієрархічний шлях категорії). Вибір енкодера вважається взаємозамінним: обидва завантажуються через єдиний інтерфейс, і відрізняється лише специфічний для корпусу адаптер форматування тексту.

При тридцяти мільйонах елементів кодування не може бути реально виконане за один транзакційний прохід. Реалізація обробляє каталог батчами і записує проміжний результат як послідовність незалежних шардів, кожен з яких супроводжується маніфестом прогресу. Така конструкція стійка до переривань – процес, примусово завершений через брак пам'яті (out-of-memory event), тимчасовий збій GPU або втручання оператора, можна запустити знову, і він відновить роботу з останнього завершеного шарда без повторного кодування раніше виконаної роботи. У масштабах цього дослідження ця властивість є операційно необхідною, а не

просто зручною: запуск без можливості відновлення втрачає кілька годин часу GPU при кожному збої.

3.1.4 Вирівнювання та нормалізація

Шарди, створені під час кодування, видаються в тому порядку, в якому оброблялися їхні вихідні рядки, що загалом не збігається з канонічним порядком каталогу, необхідним для подальших споживачів. Заключний етап підготовки збирає шарди в єдину безперервну матрицю ембеддингів, переставлену відповідно до порядку в каталозі, застосовує L2-нормалізацію по рядках, щоб подальші косинусні подібності зводилися до скалярних добутків, і записує дані про енкодер, промпт, вихідний маніфест та застосовані трансформації у супутній (sidecar) файл метаданих. Починаючи з цього моменту, кожен наступний етап системи – варіаційний автоенкодер із залишковою квантизацією, щільна базова модель та будь-які допоміжні діагностичні інструменти – споживає лише цей підготовлений артефакт і каталог, і не залежить від того, як саме вони були створені.

3.1.5 Центрування як методологічне вдосконалення

Для однієї з конфігурацій в експериментальному дослідженні перед передачею ембеддингів до RQ-VAE додається додаткова трансформація. Було виявлено, що вихідні ембеддинги двовежевої моделі, які використовуються як вхідні дані для RQ-VAE, мають сильно виражену компоненту спільного середнього – приблизно дев'яносто вісім відсотків їхньої міжрядкової дисперсії вловлюється одним вектором зсуву (offset vector). Така анізотропія є непомітною для щільного пошуковика, який є інваріантним до постійних зсувів при косинусній подібності, але вона негативно взаємодіє із залишковою векторною квантизацією: перший словник (codebook) витрачає більшу частину своєї ємності на відтворення

спільного зсуву, а не на розрізнення елементів. Щоб вирішити цю проблему, ми оцінюємо зсув лише на підмножині елементів навчального періоду (train-period subset) – зберігаючи властивість хронологічної вибірки щодо відсутності витоків, – віднімаємо його від кожного рядка каталогу та повторно нормалізуємо. Центровані ембеддинги надаються подальшим споживачам за тим самим контрактом артефактів, що й нецентровані, тому перемикання між ними не потребує змін у коді на етапі споживання. Емпірично це вдосконалення забезпечило приблизно десятивідсоткове покращення повноти пошуку (retrieval recall) у масштабі тридцяти мільйонів елементів.

3.2 Варіаційний автоенкодер із залишковою квантизацією

RQ-VAE – це компонент, який перетворює неперервний контентний ембеддинг у короткий кортеж дискретних токенів – Semantic ID (семантичний ідентифікатор) елемента. Його конструкція відповідає формулюванню, представленому в оригінальній статті про TIGER, а варіанти реалізації орієнтовані на стабільність в умовах важкохвостих (heavy-tailed) та анізотропних розподілів ембеддингів, які зустрічаються в реальних даних каталогів.

3.2.1 Архітектура

Модель складається з трьох частин, що навчаються спільно. Латентний енкодер відображає кожен вхідний ембеддинг через невелику повнозв'язну мережу прямого поширення (feed-forward network), звужуючись через послідовність прогресивно менших прихованих шарів до низьковимірного латентного вектора. Опціонально латентний вектор піддається L2-нормалізації на межі енкодера; це змушує всю подальшу квантизацію працювати на одиничній гіперсфері та стабілізує геометрію

словників проти дрейфу масштабів вхідних даних. Потім залишковий векторний квантизатор обробляє цей латентний вектор на послідовних рівнях: спочатку латентний вектор квантизується за першим словником (codebook), залишок між латентним вектором і кодом квантизується за другим словником, наступний залишок – за третім і так далі для фіксованої кількості рівнів. Вибрані індекси кодових слів на кожному рівні складають Semantic ID елемента. Нарешті, симетричний декодер – мережа прямого поширення, розмірності шарів якої є зворотними до розмірностей енкодера – відновлює вхідний ембеддинг із кумулятивного квантизованого латентного вектора.

Самі словники (codebooks) зберігаються як тензори параметрів, що підлягають навчанню, по одному на кожен рівень квантизатора. Елементи словника утримуються на тій самій гіперсфері, що й латентний вектор, коли діє латентна нормалізація, і зчитуються під час навчання та інференсу через єдиний інтерфейс, так що будь-яка трансформація нормалізації, застосована в енкодері, дзеркально відображається на стороні словника.

3.2.2 Ініціалізація

Добре відомо, що випадкова ініціалізація словників погано взаємодіє з навчанням векторної квантизації: випадково розташований словник дає майже рівномірні ймовірності призначення на першому кроці, з яких градієнту важко вирватися, і значна частина кодових слів часто «відмирає» під час ранніх епох. Щоб уникнути цього, для словників застосовується «гарячий старт» (warm-start) за допомогою кластерів K-means, обчислених на великій випадковій підвибірці вхідних ембеддингів – перший рівень на самому латентному векторі, кожен наступний рівень на залишках, отриманих на попередніх рівнях. Крім того, модель підтримує фазу розігріву автоенкодера (warm-up phase), під час якої енкодер і декодер навчаються протягом певної кількості кроків в обхід квантизатора; лише

після того, як втрати реконструкції (reconstruction loss) стабілізуються в цьому режимі, підключається квантизатор. Обидва типи ініціалізації є опціональними, але були збережені в робочій конфігурації після емпіричної перевірки, яка показала, що вони зменшують дисперсію між випадковими початковими значеннями (random seeds) і скорочують час до збіжності.

3.2.3 Цільова функція навчання

Модель навчається шляхом мінімізації зваженої комбінації двох функцій втрат. Перша – це доданок реконструкції (3.1), який порівнює вихід декодера з оригінальним вхідним ембеддингом, поєднуючи компонент середньоквадратичної помилки з компонентом косинусної відстані; косинусний доданок гарантує, що модель буде покарана за помилки напрямку, а також за помилки величини, що важливо, оскільки всі подальші споживачі ембеддингів використовують косинусну подібність.

$$\mathcal{L}_{\text{recon}}(x, \hat{x}) = w_{\text{MSE}} \|x - \hat{x}\|_2^2 + w_{\cos}\left(1 - \frac{\langle x, \hat{x}\rangle}{\|x\|\|\hat{x}\|}\right), \tag{3.1}$$

де x – вхідний вектор;

$\hat{x}$ – реконструйований вектор;

$\langle x, \hat{x}\rangle$ – векторний добуток двох векторів;

$||x||$ – L2-норма вектора.

Друга – це термін регуляризації квантизатора (3.2) – втрата відповідності (commitment loss) на кожному рівні між виходом енкодера та вибраними кодовими словами, підсумована за всіма рівнями квантизатора, – яка притягує латентний вектор до наявних кодових слів і спрямовує оновлення словників до латентного розподілу. Відносне зважування цих двох доданків є головним важелем, що контролює, чи поводиться модель більше як точний автоенкодер (висока вага

реконструкції, словники недовикористовуються), чи як агресивний квантизатор (висока вага відповідності, якість реконструкції погіршується). Конфігурація, яка використовується в цій роботі, однаково зважує обидва доданки, що емпірично забезпечує стабільне навчання без колапсу жодної зі сторін.

$$\mathcal{L}_{\text{RQ}} = \sum_{l=1}^{L}(\|\text{sg}[r_l] - e_l\|_2^2 + \beta\|r_l - \text{sg}[e_l]\|_2^2), \quad (3.2)$$

де sg – це stop-gradient оператор;

β – commitment coefficient.

3.2.4 Механізми стабільності

За нашим досвідом, основними режимами збоїв при навчанні RQ-VAE є не повільна збіжність, а раптова дивергенція (розбіжність) втрат і прихований колапс словників. Два механізми захищають від цього. Захист від дивергенції (divergence guard) відстежує рухоме значення втрат навчання і перериває запуск, якщо воно перевищує настроюване кратне значення від свого нещодавнього мінімуму протягом певної кількості кроків; це дозволяє зафіксувати режим неконтрольованого зростання втрат, характерний для надто агресивної швидкості навчання (learning rate) або погано обумовленої ваги відповідності (commitment weight), перш ніж тривалий запуск буде змарновано. Рання зупинка (early stopping) за втратами реконструкції на валідаційній вибірці з невеликим порогом мінімальної дельти припиняє навчання, щойно покращення виходить на плато, що є типовим критерієм завершення на практиці. Статистика використання словників – перплексія, кількість активних кодів і коефіцієнти зменшення залишків на кожному рівні – фіксується протягом усього навчання та аналізується після його завершення для виявлення «мертвих» кодів та неефективних глибоких рівнів.

3.2.4 Від навченої моделі до Semantic ID

Після завершення навчання модель використовується в режимі «тільки кодування» (encode-only), щоб присвоїти кожному елементу каталогу його Semantic ID. Ембеддинг кожного елемента пропускається через енкодер і, що критично важливо, через будь-яку латентну нормалізацію, яка була активною під час навчання, перш ніж потрапити до залишкового квантизатора; пропуск кроку нормалізації на цьому етапі призводить до того, що експортовані Semantic ID не збігаються з кодами, на яких навчалася модель, і це відомий режим збою, від якого ми явно захищаємося в реалізації. Отриманий кортеж індексів кодових слів і є Semantic ID. Оскільки два різні елементи каталогу можуть відображатися в один і той самий кортеж, для усунення колізій додається детермінований суфіксний токен – елементи, які мають спільний базовий кортеж, упорядковуються за стабільним хешем їхнього ідентифікатора, і їм присвоюються послідовні значення суфікса, – що забезпечує ін'єктивне відображення елементів каталогу на розширені Semantic ID. На основі цього відображення будується та серіалізується префіксне дерево (trie) валідних ідентифікаторів; це префіксне дерево є структурою, до якої звертається генеративна рекомендаційна система під час декодування, щоб обмежити свій променевий пошук (beam search) лише ідентифікаторами з каталогу.

3.3 Генеративна рекомендаційна система

Генеративна рекомендаційна система – це авторегресійний трансформер типу decoder-only, навчений на послідовностях токенів Semantic-ID, який під час декодування поєднується з обмеженим променевим пошуком (beam search) під керівництвом префіксного дерева (trie) Semantic-ID.

3.3.1 Представлення послідовностей

Кожна сесія користувача перевиражається як конкатенація ідентифікаторів Semantic ID відвіданих елементів у хронологічному порядку з додаванням за потреби невеликого словника структурних токенів (межа послідовності, роздільник між елементами, падінг). Позиції кодових токенів розмежовуються між рівнями квантизатора – індекс кодового слова, отриманий на другому залишковому рівні, перебуває в іншій області словника моделі, ніж той самий індекс, отриманий на першому рівні, – завдяки чому модель може вивчати специфічну для кожного рівня семантику токенів замість їх змішування. Таким чином, навчальний корпус є потоком прикладів фіксованої форми, в яких контекстне вікно токенів нещодавньої історії поєднується з кортежем Semantic-ID наступного елемента взаємодії як цільовою мішенню для прогнозування.

3.3.2 Модель

Трансформер являє собою стек блоків причинного самоуваги (causal self-attention) з попередньою нормалізацією (pre-norm), які працюють над єдиним потоком токенів, із навченим позиційним ембеддингом та спільним вхідним/вихідним ембеддингом токенів. Маючи розмір у кілька мільйонів параметрів у конфігураціях, використаних у цій роботі, він є навмисно компактним – достатньо великим, щоб увібрати структуру граматики кодів Semantic-ID, але достатньо малим, щоб один повний прохід навчання по корпусу взаємодій із тридцяти мільйонів елементів завершувався протягом одного GPU-дня. Навчання оптимізує стандартну цільову функцію крос-ентропії (3.3) для наступного токена зі згладжуванням міток (label smoothing).

$$\mathcal{L}_{\text{TIGER}} = -\sum_{i=1}^{T} \log P_\theta\,(t_i | t_{<i}, h), \quad (3.3)$$

де h – історія проглянутих товарів користувача, представлених у виді списку кортежей Semantic ID;

t – наступний товар у вигляді кортежу Semantic ID.

Варіант конфігурації (навчання з «маскованим суфіксом», masked-suffix) виключає детермінований суфіксний токен-розв'язувач колізій із функції втрат. Це відокремлює відновлювані, семантично значущі префіксні токени – які модель в принципі може передбачити з історії – від суфіксного токена, який визначається довільним детермінованим хешем і не несе корисного сигналу для навчання. Цей варіант забезпечив стабільне покращення повноти (recall) в масштабі чотирьох словників (codebooks) і використовується як робочий режим навчання для тридцяти мільйонів елементів.

3.3.3 Обмежене декодування

Під час інференсу модель отримує як умову історію сесії, і від неї вимагається авторегресійно згенерувати кортеж Semantic-ID прогнозованого наступного елемента. Необмежений променевий пошук часто створював б послідовності токенів, які не відповідають жодному елементу каталогу – оскільки Semantic ID є розрідженими в комбінаторному просторі токенів, – тому пошук обмежується префіксним деревом (trie), побудованим під час матеріалізації Semantic-ID. На кожному кроці декодування префіксне дерево повертає набір індексів токенів, які розширюють поточний префікс до валідного ідентифікатора каталогу; логіти недозволених токенів маскуються мінус нескінченністю перед вибором променя. Таким чином, кожен завершений промінь за своєю конструкцією відповідає наявному елементу, а префіксне дерево транслює топ-K променів назад у топ-K ранжування каталогу.

3.3.4 Оцінювання

Оцінка відкладеної тестової вибірки (test set), взятої з хронологічно найпізнішого періоду потоку взаємодій, виконується шляхом запуску обмеженого променевого пошуку для кожної сесії та порівняння прогнозованого топ-K з істинним наступним елементом (ground-truth). Основними звітними метриками є Recall@K (3.4) та NDCG@K (3.5). Ця сама тестова вибірка також оцінюється відносно щільної двовежевої базової моделі з використанням її власного пошуку найближчих сусідів, завдяки чому обидві системи порівнюються на однакових сесіях та однакових відкладених елементах. Крім того, наводяться два діагностичні варіанти метрики: зріз холодного старту (cold-start slice), який обмежує цільові елементи лише тими, що не зустрічалися під час навчання, ізолюючи узагальнювальну здатність, яка й була початковою мотивацією для створення TIGER; та зріз за префіксом-K (prefix-K slice), який зараховує промінь як влучання щоразу, коли він збігається з цільовим елементом за першими K токенами Semantic-ID, що дозволяє відокремити внесок префікса коду (від загального до конкретного) від внеску суфікса-розв'язувача колізій.

$$\text{Recall@}K = \frac{1}{N}\sum_{n=1}^{N} 1\left(t_n \in \left\{t_n^{(1)}, \dots, t_n^{(K)}\right\}\right), \tag{3.4}$$

$$\text{NDCG@}K = \frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} \frac{1\left(t_n = \hat{t}_n^{(k)}\right)}{\log_2(k+1)}, \tag{3.5}$$

де t_n – цільовий товар, з яким взаємодіяв користувач.

4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ

4.1 Конфігурація експериментів

Цей розділ визначає корпуси, базову модель для порівняння, протокол оцінювання та бюджет навчання, за яких було отримано всі результати в цьому розділі.

4.1.1 Датасети

У дослідженні використовуються два корпуси, які охоплюють режими розміру каталогу та щільності сесій, найбільш релевантні для промислових рекомендацій.

Перший, який надалі називається session_products, є внутрішнім датасетом prom.ua, що містить приблизно 31 мільйон унікальних продуктів і близько 75 мільйонів навчальних прикладів прогнозування наступного товару, отриманих з приблизно 19 мільйонів сесій користувачів. Сесії є щільними в часі – більшість із них містять кілька подій протягом кількох хвилин, – а розподіл частоти елементів має важкий хвіст із невеликою «головою» високопопулярних товарів і довгим «хвостом» елементів, що з'являються один або два рази.

Другий корпус, Amazon Reviews 2023 – Beauty and Personal Care (підмножина 5-core), містить приблизно двісті 7 тисяч елементів і 6.5 мільйонів подій, з яких близько 5.25 мільйонів формують навчальну вибірку. Траєкторії окремих користувачів є розрідженими, з інтервалами від місяців до років між подіями, а розподіл елементів є порівняно плоским. Таким чином, ці два корпуси доповнюють один одного за обома осями, які в нещодавніх роботах були визначені як ключові фактори впливу ефективності генеративного пошуку: потужність (cardinality) каталогу та щільність сесій.

4.1.2 Вибірки

Ці два корпуси відрізняються своїми протоколами розподілу на вибірки, в кожному випадку дотримуючись конвенції, встановленої для цього корпусу в попередніх роботах. Для session_products потік взаємодій розділяється хронологічно за охопленням днів приблизно у співвідношенні 80/10/10, так що валідаційний та тестовий періоди наступають суворо пізніше в часі, ніж навчальний період. Цей протокол запобігає часовому витоку даних і, крім того, виявляє нетривіальний режим холодного старту – приблизно чверть подій тестового періоду пов'язана з товарами, які не мали взаємодій у навчальному періоді. Для Amazon Beauty PC використовується стандартний протокол leave-last-out (виключення останнього), в якому остання взаємодія кожного користувача становить тестову ціль, а передостання формує валідаційну ціль. Для обох корпусів вибірки зафіксовані у записах на рівні окремих подій, створених конструктором даних, і не перетасовуються на жодному з наступних етапів.

4.1.3 Базова модель

Базовою моделлю для порівняння є щільний двовежевий пошуковик із порівнянною з генеративною моделлю кількістю параметрів. Його вежа запитів є трансформерним енкодером над нещодавною історією ембеддингів елементів користувача, застосованим до тих самих векторів контенту, які використовуються як вхідні дані для RQ-VAE. Його вежа елементів є невеликим багатошаровим перцептроном над ембеддингом елемента. Виходи обох веж піддаються L2-нормалізації та порівнюються за допомогою скалярного добутку, а модель навчається з контрастивною функцією втрат (InfoNCE) всередині батчу, використовуючи інші елементи в кожному міні-батчі як негативні приклади. Базова модель узгоджується за кількістю параметрів із генеративною рекомендаційною системою з

точністю до десяти відсотків, тому будь-яка різниця в якості пошуку зумовлена архітектурою та схемою представлення, а не потужністю моделі. Під час інференсу вихід вежі елементів попередньо обчислюється для кожного елемента каталогу, і ранжування зводиться до запиту top-K скалярного добутку відносно цієї матриці.

4.1.4 Метрики

Усі результати наводяться як Recall@10 та NDCG@10 що є стандартними метриками в літературі з послідовних рекомендацій. Recall@K – це частка тестових сесій, у яких істинний наступний елемент (ground-truth) з'являється в будь-якому місці серед top-K передбачень. NDCG@K додатково винагороджує влучання, які знаходяться вище в рейтингу. Ці дві метрики, взяті разом, дозволяють відокремити питання «чи взагалі модель передбачила правильний товар з яким взаємодіяв користувач» від «чи вивела вона цей товар на початку списку».

Також наводяться два діагностичні обмеження цих самих метрик. Зріз холодного старту обмежує оцінювання тестовими прикладами, цільовий елемент яких мав нуль взаємодій протягом навчального періоду; це вимірює узагальнювальну здатність для дійсно не бачених раніше елементів – режим, який і був початковою мотивацією для генеративного пошуку за Semantic ID. Зріз за префіксом-K (prefix-K slice), визначений лише для генеративної рекомендаційної системи, зараховує промінь як влучання щоразу, коли він збігається з Semantic ID цільового елемента за першими K токенами, незалежно від того, чи збігається повний ідентифікатор (включаючи суфікс-розв'язувач колізій).

Порівняння Recall для префікса-K та повного ID дозволяє відокремити внесок префікса коду (від загального до конкретного) від внеску суфікса і є основним діагностичним інструментом.

4.1.5 Навчання та апаратне забезпечення

Обидві моделі навчаються з оптимізатором AdamW на одному GPU з 24 гігабайт відеопам’яті, корпус електронної комерції обробляється у змішаній точності (mixed precision), щоб було менше завантаження на оперативну- та відео- пам’ять. Двовежева базова модель збігається за один навчальний прохід по всьому потоку взаємодій на session_products, генеративна рекомендаційна система використовує той самий бюджет в один прохід на цьому корпусі. На меншому корпусі Amazon Beauty PC обидві моделі навчаються протягом фіксованої кількості епох по всьому потоку подій. Усі запуски відстежуються на власному сервері MLflow, який зберігає покрокові криві навчання та валідації, гіперпараметри моделі та отримані результуючі файли JSON оцінювання, на основі яких побудовано таблиці в цьому розділі.

4.2 Основні результати

Головне міжкорпусне порівняння наведено в таблиці 4.1. Обидва корпуси оцінюються за протоколом, визначеним у розділі 4.1.

Таблиця 4.1 – Порівняння метрик TIGER та базової моделі

Корпус	TIGER		Базова модель		Різниця
	Recall @10	NDCG @10	Recall @10	NDCG @10	
Session products	0.0190	0.0100	0.0648	0.0305	0.29×
Session products (cold start)	0.0233	0.0122	0.0795	0.0373	0.29×
Amazon Beauty	0.0141	0.0073	0.0028	0.0013	5.12×

З таблиці 4.1 безпосередньо випливають два спостереження:

– на корпусі електронної комерції з 30 мільйонами елементів щільна базова модель перевершує генеративну рекомендаційну систему приблизно в три з половиною рази за метрикою Recall@10, і це співвідношення є

практично однаковим як на популярних елементах («голові»), так і на зрізах холодного старту – тобто розрив не зосереджений в якомусь одному режимі популярності цільових товарів;

– на корпусі Amazon Beauty PC з 2 тисячами елементів відносний порядок змінюється на протилежний: генеративна рекомендаційна система перевершує щільну базову модель у п'ять разів на тих самих архітектурах, тих самих рецептах навчання та тому самому шляху виконання коду. Єдиними змінними, які змінюються між двома оцінюваннями, є сам корпус та вибір контентного енкодера, зумовлений цим корпусом.

Ця зміна порядку є центральним емпіричним висновком цього розділу. Сама по собі вона є сильним запереченням для будь-якого трактування нижчої ефективності TIGER на session_products як універсальної архітектурної плати за генеративний пошук порівняно зі щільним пошуком; та сама архітектура рішуче перемагає на другому корпусі. Тому природним наступним питанням є не те, яка архітектура є внутрішньо сильнішою, а швидше те, які властивості корпусу визначають переможця, і решта розділу побудована навколо трьох взаємодоповнюючих діагностичних методів, які розкладають, на що витрачається або де втрачається Recall моделі TIGER у кожному режимі. Двовежева базова модель на Amazon Beauty PC додатково демонструє чітку неповну збіжність (under-convergence), але якісний напрямок цієї зміни є настільки значним, що його навряд чи вдасться змінити лише за допомогою кращого налаштування базової моделі.

4.3 Аналіз метрик

Корисним способом дослідити, на що витрачається повнота (recall) генеративної рекомендаційної системи, є розкладання повного передбачення Semantic-ID на внесок його семантично значущого префікса (від загального до конкретного) та внесок детермінованого

суфікса-розв'язувача колізій, який додається під час матеріалізації. Метрика префікса-K, визначена в розділі 4.1, ізолює перший компонент, зараховуючи промінь як влучання щоразу, коли він збігається з Semantic ID цільового елемента за першими K токенами. Таким чином, розрив між повнотою для префікса-K та повнотою для повного ідентифікатора вимірює частку збігів префіксів у промені, яку етап суфікса не зміг перетворити на влучання на рівні окремих елементів.

У таблиці 4.2 наведено декомпозицію для генеративної рекомендаційної системи на session_products при робочій ширині променя.

Таблиця 4.2 – Recall@10 на session_products при послідовних довжинах префікса

Довжина префіксу	Recall @10
Prefix-3	0.0382
Prefix-4	0.0236
Повністю (prefix-4 + токен суфіксу)	0.0190

Якщо дивитися вниз по таблиці, повнота для префікса-3 приблизно вдвічі перевищує повноту для повного ідентифікатора: модель розміщує правильні перші три токени Semantic-ID в межах першої десятки у 3.82% тестових сесій, але лише 1.90% цих сесій завершуються правильним повним ідентифікатором після четвертого токена та суфікса-розв'язувача колізій. Приблизно половина збігів префікса-3 у промені втрачається при переході від префікса до конкретного елемента.

4.3.1 Більша ширина променевого пошуку

Щоб перевірити, чи можна суфіксний податок пов'язати з недостатнім обсягом пошуку, а не з властивістю самих кодів, декодер було повторно оцінено при десятикратній робочій ширині променя на діагностичній підвибірці з двох тисяч сесій. Результат наведено в таблиці 4.3.

Таблиця 4.3 – Метрики при зміні ширини променя на діагностичній підвибірці

Ширина променя	Recall @10	Recall @100	Prefix-1	Prefix-2	Prefix-3	Prefix-4
Beam=10	0.0173	–	–	–	0.039	0.024
Beam=100	0.0165	0.0320	0.459	0.160	0.064	0.038

Показник Recall@10 статистично не змінився при ширшому промені, що узгоджується зі стандартним спостереженням, згідно з яким широкі промені зберігають шляхи з низькою апріорною ймовірністю, які витісняють лідерів фінального ранжування. Діагностична цінність ширшого пошуку полягає в нижніх рівнях: верхня межа того, що могли б забезпечити коди – присутність повного prefix-4 у будь-якому місці променя зростає лише з 2.36% до 3.80% при десятикратному збільшенні променя, а сам Recall@100 досягає лише 3.20%. Співвідношення між Recall@100 та частотою присутності повного prefix-4 у промені, що становить приблизно 0.85, визначає умовну ефективність етапу розв'язання суфікса: якщо правильний prefix-4 взагалі є в промені, суфіксний розв'язувач колізій відновлює його як влучання на рівні елемента приблизно в п'яти випадках із шести. Таким чином, основним режимом збою є не розв'язувач колізій, а самі коди: навіть при розширенні бюджету пошуку на порядок правильний префікс-4 відсутній у промені у понад дев'яносто шести відсотках тестових сесій.

Таким чином, повнота обмежена кодами, а не декодуванням.

4.3.2 Міжкорпусне порівняння

Та сама декомпозиція, обчислена на Amazon Beauty PC, демонструє структурно іншу картину, наведену в таблиці 4.4. Повнота для prefix-3 та повнота для повного ідентифікатора збігаються в межах 5% на меншому корпусі, на відміну від двократного розриву, що спостерігався на

session_products. Суфіксний податок, з практичної точки зору, відсутній на Amazon Beauty PC і поглинає половину доступної повноти на корпусі електронної комерції.

Таблиця 4.4 – Міжкорпусне порівняння суфіксного податку

Корпус	Розмір каталогу	Prefix-3 R@10	R@10	Різниця
Session products	30.9 мільйонів	0.0382	0.0190	2.01×
Amazon Beauty	207 тисяч	0.0148	0.0141	1.05×

Дві взаємодоповнювальні причини узгоджуються з цією різницею:

– Комбінаторна: з каталогом у двісті тисяч елементів чотирьох-токенний простір Semantic-ID є розріджено заповненим і колізії трапляються рідко, тому етап суфікса майже завжди є тривіальним.

– Геометрична: менший і плоскіший каталог забезпечує багатшу дискримінацію на одну комірку на найглибшому рівні квантизатора, залишаючи менше елементів у кожній окремій комірці prefix-4 для усунення неоднозначності між ними. Самої лише першої причини достатньо, щоб передбачити величину розриву.

4.3.3 Висновок

Декомпозиція дає два висновки, які разом пояснюють значну частину міжкорпусної зміни порядку, описаної в розділі 4.2. Ефективна верхня межа генеративної рекомендаційної системи на корпусі визначається насамперед якістю її кодів Semantic-ID, а не бюджетом пошуку під час декодування. Штраф, пов'язаний із розв'язанням колізій на найглибшому рівні квантизатора, сам по собі є функцією від потужності каталогу, і він є основною вартістю генеративного пошуку в масштабі 30 мільйонів елементів – приблизно половина повноти (Recall), яку несуть у собі коди, втрачається при переході від префікса до конкретного елемента. На корпусі,

що на два порядки менший, ця вартість фактично зникає. Висновок для ширшого аналізу розриву полягає в тому, що зусилля з покращення повноти генеративної рекомендаційної системи на великих каталогах слід спрямовувати на самі коди – або шляхом розширення чи поглиблення словників (codebooks), щоб кожна комірка prefix-4 містила менше елементів, або шляхом повного обходу суфіксного етапу через гібридний пошук, який використовує префікс як генератор грубих кандидатів. Обидва ці напрями розглядаються в розділі висновків.

4.4 Геометрія контентних ембеддингів

RQ-VAE отримує на вхід щільний ембеддинг для кожного елемента каталогу, тому якість семантичних ідентифікаторів (Semantic ID), які він створює, обмежена зверху дискримінативною структурою цього розподілу ембеддингів. Попередня діагностика виходів вежі елементів навченої двовежевої моделі – варіанту ембеддингів, що використовувався як вхідні дані для RQ-VAE протягом усього ланцюжка на 30 млн елементів – виявила, що цей розподіл є геометрично дегенерованим у спосіб, який несприятливо взаємодіє саме із залишковою квантизацією, і мотивує застосування єдиної трансформації артефакту ембеддингів, вплив якої на повноту (recall) на наступних етапах є єдиною абляцією в цьому розділі.

4.4.1 Діагностика

Спільний компонент у розподілі ембеддингів елементів навченої двовежевої моделі було оцінено шляхом обчислення емпіричного вектора середнього значення для всіх елементів каталогу та вимірювання його норми відносно норм ембеддингів окремих елементів. Результат наведено у верхній частині таблиці 4.5. Вектор середнього має норму 0.98 на корпусі, де вектори окремих елементів піддані L2-нормалізації до одиничної

довжини: приблизно дев'яносто вісім відсотків дисперсії рядків уловлюється одним спільним зсувом (offset), характерним для кожного елемента. Як прямий наслідок, косинусна подібність між двома випадково вибраними елементами становить приблизно 0.96, тобто весь каталог займає невелику зону, замість того щоб охоплювати всю одиничну сферу.

Такий анізотропний розподіл є інваріантним щодо операцій щільної базової моделі. Косинусна подібність між запитом та елементом не змінюється під впливом постійного адитивного зсуву, застосованого однаково до всіх елементів, оскільки цей зсув вносить однаковий скалярний добуток у кожне порівняння і нівелюється при ранжуванні. Як наслідок, двовежевий пошуковик видає однакове впорядкування найближчих сусідів як зі спільним середнім, так і без нього.

Залишкова векторна квантизація, навпаки, є чутливою до зсуву. Перший словник RQ-VAE підганяється під розподіл латентного вектора і неминуче виділяє значну частину своєї ємності на домінантний напрямок – у цьому випадку на спільне середнє. Ефективні елементи словника на першому рівні стають незначними, у першому залишку домінує той самий напрямок, і модель приходить до майже некорисної ієрархії, в якій перший рівень нездатний розрізняти елементи. Емпіричною ознакою цього збою, що спостерігавалася в ранніх запусках на 30 млн елементів, де нецентровані ембеддинги використовувалися як вхідні дані для RQ-VAE, є колапс використання словника першого рівня та префікс Semantic-ID, який не несе інформації про ідентичність елемента. Діагностика та режим збою тісно пов'язані: геометрія, яка є непомітною для щільного пошуковика, є саме тим, що ламає квантизатор.

4.4.2 Процедура центрування

Рішення полягає в тому, щоб відняти спільне середнє перед передачею ембеддингів до RQ-VAE (таблиця 4.5). Щоб зберегти властивість

хронологічного розподілу на навчальній, валідаційній та тестовій вибірці. Для відсутності часових витоків, середнє оцінюється лише для елементів, які мали принаймні одну взаємодію у навчальному періоді – приблизно двадцять мільйонів із тридцяти одного мільйона елементів каталогу – а потім уніфіковано застосовується до всіх елементів каталогу під час підготовки. Центровані ембеддинги повторно піддаються L2-нормалізації, щоб подальші функції втрат на основі косинусної подібності залишалися на одиничній сфері. Трансформація надається за тим самим контрактом артефактів, що й нецентровані ембеддинги, тому перемикання між ними не потребує змін у коді ні в RQ-VAE, ні в будь-якому наступному споживачі.

Таблиця 4.5 – Геометрична статистика розподілу ембеддингів елементів двовежевої моделі до та після центрування

Статистика	Нецентровані вектори	Центровані вектори
Норма середнього	0.98	0.0004
Косинусна схожість випадкових векторів	0.96	0.002

4.4.3 Вплив на пошук

У таблиці 4.6 наведено наскрізний ефект на Recall@10 на тестовій вибірці для 30 млн елементів при порівнянні трьох варіантів вхідних даних RQ-VAE. Перший рядок – це оригінальний рецепт TIGER, що використовує попередньо навчений контентний ембеддинг безпосередньо; другий використовує центрований вихід двовежевої моделі як вхідні дані для RQ-VAE. Приріст, зумовлений центруванням, становить приблизно десять відсотків як на популярних елементах («голові»), так і на зрізах холодного старту, і є найбільшим окремим методологічним покращенням повноти генеративної рекомендаційної системи, зафіксованим у цій ітерації. Приріст для холодного старту має порівнянну величину з приростом для популярних елементів, що узгоджується з діагностичною інтерпретацією: центрування

покращує геометрію всіх елементів уніфіковано, а не переважно тих, що були знайдені під час навчання.

Таблиця 4.6 – Recall@10 залежно від вхідного ембеддингу RQ-VAE

Вхід RQ-VAE	Recall @10	Cold-start Recall@10
Контентні ембедінги	0.0173	0.0210
Центрований вихід із двох-веженної моделі	0.0190 (+9.5%)	0.0233 (+10.7%)

4.4.4 Висновок

Результат центрування узагальнюється за межі конкретного енкодера, використаного тут. Будь-який розподіл ембеддингів, який має слугувати вхідними даними для RQ-VAE, слід перевіряти на наявність анізотропної спільної компоненти перед навчання квантизатора; ця перевірка є дешевою, режим збою є прихованим (втрати навчання та реконструкції виглядають нормально, тоді як отримані Semantic ID є неінформативними), а коригувальна трансформація є однорядковим кроком підготовки, який нічого не коштує під час інференсу. Цей результат також дає інформацію про те, що відображає ширший розрив порівняно зі щільною базовою моделлю: принаймні десять відсотків розриву на session_products були артефактом подачі в RQ-VAE розподілу, який квантизатор був структурно нездатний спожити, а не властивістю самого генеративного пошуку.

4.5 Приклади

Порівняння в розділі 4.2 відображає агреговану якість пошуку, а діагностика в розділах 4.3 та 4.4 пояснює, на що витрачається або де втрачається повнота (recall) генеративної рекомендаційної системи відносно кодів, які вона має в розпорядженні. Жоден із цих поглядів нічого не говорить про те, які саме сесії кожна модель обробляє добре. Щоб вирішити

це питання, обидві системи оцінили однакові 5000 сесій з тестів популярних товарів, а передбачення для кожної сесії були збережені на диск для попарного аналізу. Порівняння в цьому розділі обчислюються на цій узгодженій вибірці.

Аналіз розкладає тестову сукупність за чотирма осями:

– якість ранжування за умови влучання;

– виведення холодних елементів та елементів із хвоста;

– семантична зв'язність прогнозованого списку;

– чутливість до довжини історії сесії.

Також додатково наведені тестові приклади та передбачення моделі.

Історія сесії містила два зарядні пристрої: універсальний кабельний зарядний пристрій VIVO 2A та швидкий зарядний пристрій потужністю 20 Вт із двома портами USB-A / Type-C для моделі VIVO V20 (рисунок 4.1).

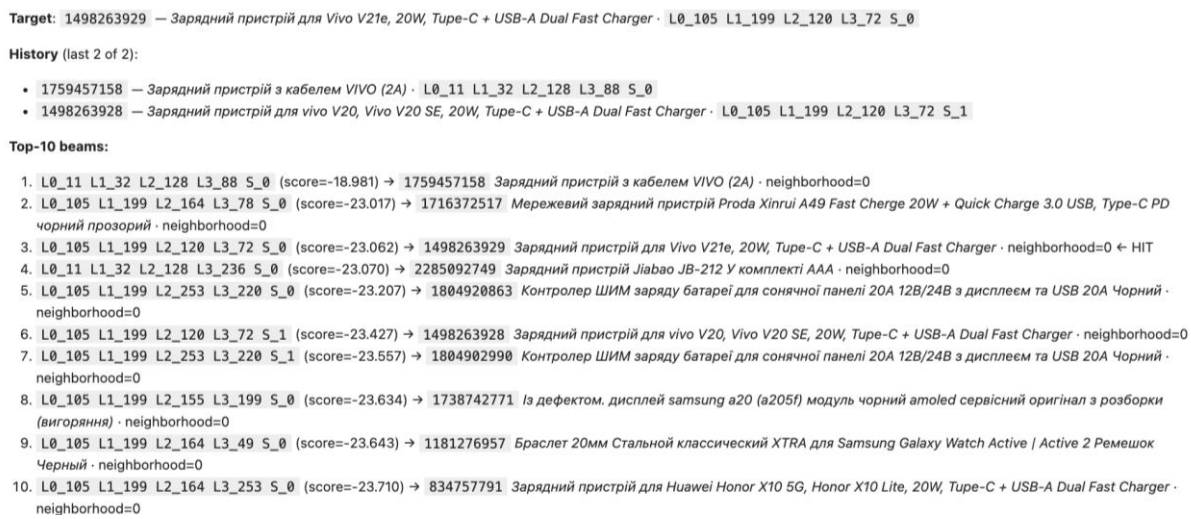


Рисунок 4.1 – Приклад №1 з зарядними пристроями

Цільовим елементом був швидкий зарядний пристрій потужністю 20 Вт із двома портами для VIVO V21e – та сама категорія аксесуарів, та сама потужність, той самий фізичний інтерфейс, але інший варіант моделі телефону. У результаті модель помістила точний цільовий елемент на позицію 3 у першій десятці променів, при цьому два вищих за рангом

промені були сусідніми швидкими зарядними пристроями в тій самій комірці Semantic-ID. Шість із десяти променів – це швидкі зарядні пристрої потужністю 20 Вт; три – це безпосередньо зарядні пристрої лінійки VIVO, включаючи цільовий елемент.

На рисунку 4.2 показано, що історія сесії містила шість чайників зі свистком із нержавіючої сталі бренду Kamille різних розмірів (1 л, 2.7 л, 3 л) та кольорів (білий, сірий, чорний). Цільовим елементом був чайник Kamille об'ємом 3 л із нержавіючої сталі з нейлоновою ручкою. Дев'ять із десяти променів моделі в першій десятці – це чайники Kamille (або суміжні чайники зі свистком із нержавіючої сталі) розміром 2.0–2.8 л; промінь на позиції 1 – це чайник Kamille об'ємом 2 л із того самого сімейства продуктів, що й цільовий. Точного цільового SKU немає в першій десятці, але кожне передбачення в промені належить до того самого бренду, типу продукту та класу розміру.

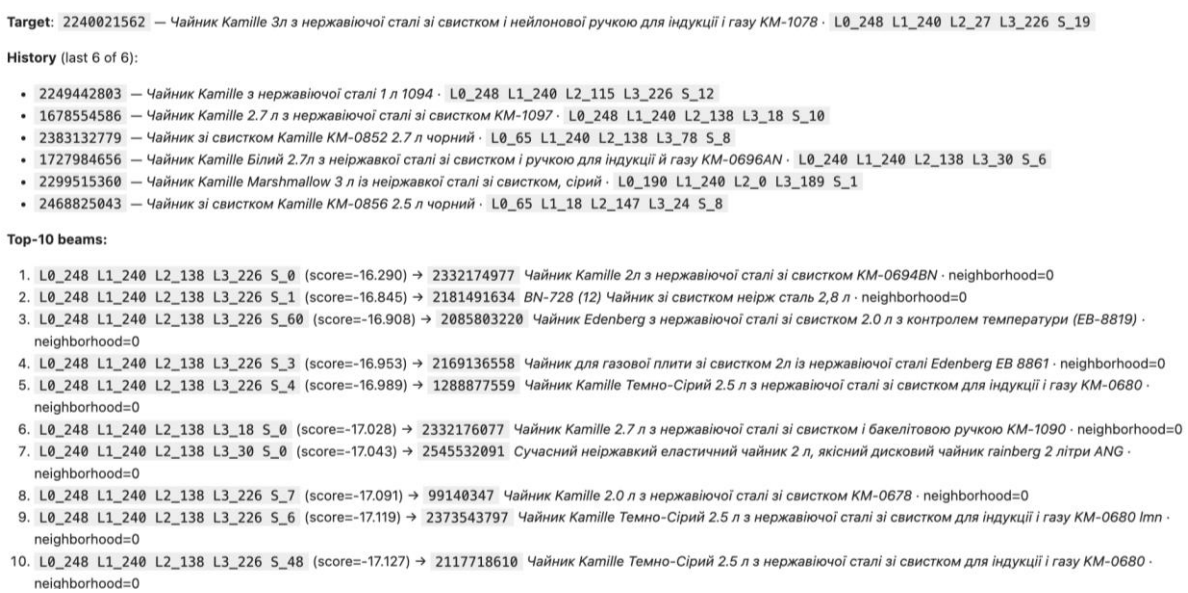


Рисунок 4.2 – Приклад №2 з чайниками Kamille

Історія сесії містила лише один елемент: літню шину CST Adreno AD-R9 розміром 245/50 ZR18. Цільовим елементом був наступний за порядком ідентифікатор продукту в тому самому сімействі товарів – літня шина CST

Adreno AD-R9 розміром 245/40 ZR18. Ідентифікатори елементів у каталозі 1472583144 та 1472583145: той самий бренд, однаковий малюнок протектора, сусідній розмір. Передбачення моделі на позиції 1 виявилося самим елементом з історії, повернутим без змін. Решта першої десятки була заповнена суміжними літніми шинами інших брендів і розмірів, але цільовий SKU був відсутній (рисунок 4.3).



Target: `1472583145` — *Літні шини CST Adreno AD-R9 245/40 ZR18 97Y XL* · `L0_81 L1_226 L2_15 L3_226 S_0`

History (last 1 of 1):

- `1472583144` — *Літні шини CST Adreno AD-R9 245/50 ZR18 104Y XL* · `L0_81 L1_110 L2_98 L3_226 S_0`

Top-10 beams:

1. `L0_81 L1_110 L2_98 L3_226 S_0` (score=-17.107) → `1472583144` *Літні шини CST Adreno AD-R9 245/50 ZR18 104Y XL* · neighborhood=0
2. `L0_81 L1_110 L2_121 L3_226 S_0` (score=-21.538) → `2424610714` *Шины літня Pirelli Scorpion 235/55 R18 100V Demo* · neighborhood=0
3. `L0_81 L1_110 L2_98 L3_229 S_0` (score=-23.394) → `2435190167` *Всесезонні шини Pirelli Scorpion Zero All Season 235/55 R19 101T Elect AO+* · neighborhood=0
4. `L0_81 L1_110 L2_87 L3_107 S_0` (score=-24.323) → `1756264695` *Літні шини Michelin Latitude Tour HP 235/55 R18 100V* · neighborhood=0
5. `L0_81 L1_110 L2_106 L3_248 S_0` (score=-24.437) → `1505510473` *Шини Б/У зимова 235/50 R19 PIRELLI SOTTOZERO WINTER 210 SERIE II 5mm* · neighborhood=0
6. `L0_81 L1_110 L2_87 L3_195 S_0` (score=-24.680) → `2342115422` *БУ літні шини Continental ContiSportContact 5 235/55 R18 100V ContiSeal 2019 Португалія* · neighborhood=0
7. `L0_81 L1_110 L2_121 L3_195 S_0` (score=-24.746) → `2492810281` *Літні шини Continental CrossContact H/T 235/55 R18 100V* · neighborhood=0
8. `L0_81 L1_110 L2_87 L3_13 S_0` (score=-24.748) → `2445871330` *Шины Б/У літня 235/55 R19 CONTINENTAL CROSSCONTACT LX SPORT 6mm* · neighborhood=0
9. `L0_81 L1_110 L2_87 L3_164 S_0` (score=-24.940) → `1908451206` *Літні шини б/у 235/55 R19 Continental ContiSportContact 5* · neighborhood=0
10. `L0_81 L1_110 L2_34 L3_195 S_0` (score=-25.020) → `2299661475` *Шини літні 255/60/18 Starmaxx Incurro A/T ST440 112H* · neighborhood=0

Рисунок 4.3 – Приклад №3 з літніми шинами CST Adreno

Одна із основних гіпотез перед початком цього дослідження була в тому, що коди, як вивчає RQ-VAE, несуть в собі якусь семантичну суть від загального до конкретного. Це підтвердилось, і на рисунку 4.4 можна побачити, що код першого рівня з індексом 79 сувмістив у собі усі смартфони.

item_id	title	L0	L1	L2
2444587359	Мобільний телефон realme 12 Plus 5G 8/256GB Pioneer Green mayak	79	180	37
2499418789	Смартфон 6.6" TCL 405 2/32Gb 4G 2-SIM 8 ядер Android 12 Dark Grey Уцінка	79	214	209
2456592638	TECNO Pova 6 12/256Gb NFC (Comet Green), Новий спритний смартфон для роботи, Helio G99 Ultra, 2.2 ГГц, AMOLEDGGPB	79	77	196
2440639558	Смартфон Samsung Galaxy S22 8/128GB Dual Bora Purple (SM-S901B)	79	214	251
2223912940	Смартфон Infinix GT20 Pro (X6871) 12/256GB Mecha Orange	79	199	28
2353252551	Смартфон Motorola Edge 50 NEO 8/256 Grisaille	79	116	248
2467724925	Poco C75 6.88" 8 GB RAM 256 GB ROM 5160 мА·год 120 Гц 50 Мп 4G NFC Android14 Gold	79	205	164
2452527639	ORIGINAL! Мобільний телефон Samsung Galaxy A16 LTE 4/128Gb Gray (SM-A165FZABEUC) - Якість! Гарантія! MegaTorg.com.ua	79	192	99
2526273716	Мобільний телефон Samsung Galaxy A05s (A057) 4/128 Green	79	93	205
2493923288	Apple Iphone 13 Pro 512Gb Sierra Blue Neverlock Оригінал!	79	157	232
2338417314	Смартфон Apple iPhone 14 Pro Max 256GB Gold (MQ9W3)	79	199	232
2148763502	Samsung Galaxy S20+ 5G SM-G986F-DS 12/128GB Cosmic Grey	79	55	94

Рисунок 4.4 – Прикладі товарів із кодом L0_79

На рисунку 4.5 показано, що коди L0_161 і L1_93, включають в собі зимове взуття бренду Adidas (де код L0_161 відповідає за Adias, а L1_93 за зимовий сезон.

item_id	title	L0	L1	L2
1998034592	Чоботи зимові водонепроникні Norfin KLONDAIK 2 (EVA/TPR + 2 зачепи, вкладки) -40° / р.43,16990-43	17	224	98
2495397931	Босоніжки жіночі рожеві замшеві на середньому підборі із закритим носом	17	242	218
2013278652	Жіночі Лосіни на хутрі Дуже теплі класичні легінси. Зимові лосини на хутрі	17	220	212
2366315767	Черевики жіночі демісезонні темно-синього кольору 191329S	17	92	231
2424205876	Premium! Жіночі бордові ботфорти Зимові чоботи теплі панчохи на хутрі Зима	17	105	85
2345484132	Зимові жіночі кросівки Adidas Campus Green Winter, Адідас кампус зелені хутро	17	165	80
2383782298	Жіночі лосини зимові з Термоефектом на хутрі р.52-56	17	248	177
886123356	Уги UGG Australia Bailey Button чоботи черевики зимові овчина цигейка. Оригінал. 37 р./23 см.	17	201	106
1980865122	Черевики дитячі зимові еко-нубук черевики 857М на шнурівці утеплені 25 розмір / устілка 15,4 см коричневі	17	197	16
1028278096	Замшеві зимові чобітки на невеликому підборі 3,5 см	17	2	101
2211436789	Берці черевики зимові чоловічі на хутрі високі шкіряні тактичні утеплені з клапаном на товстій підошві -20°C	17	139	3
2043663612	Стильні та натуральні жіночі Зимові кросівки з хутром 💛💙🏆	17	111	4
2041579314	Чоботи дутики жіночі зимові сірі на танкетці Dreamstan 4348 42	17	61	249
2218570189	Жіночі Військові Черевики pixel сейф 38(VS)	17	244	82
2385061677	Зимові забави Чарівний гном Авт: Чхайло О. Вид: Торсінг	17	139	253
2219623655	Жіночі сенсорні рукавички Fashion Gloves теплі м'які сірі	17	112	66
1944569575	Берці робочі утеплені, зимове спецобвзуття, утеплене робоче взуття, берці з хутром Польща 40 р • промислових підприємств;, Зима, 47	17	102	130
1900262885	Черевики жіночі зимові	17	191	88

Рисунок 4.5 – Приклади товарів із кодом L0_161 і L1_93

При цьому, що цікаво, код L0_163 відповідає за бренд Nike, таким чином ми можемо очікувати, що комбінація L0_163 та L1_93 повинна відповідати за зимове взуття Nike, що і зображено на рисунку 4.6:

item_id	title	L0	L1	L2
2419532628	Зимові кросівки nike air force хутро замшеві форси Nike air force зимові кросівки,	163	93	216
335409979	Класичні Зимові черевики Дуже м'яка шкіра. GriSport (45р)	163	93	197
1334011996	Зимові кросівки Nike Huarache X Acronym City, кросівки найк хуарачи, зимові кросівки Nike Huarache Acronym	163	93	216
2416433855	Зимові черевики Nike ACG Running Grey модні зимові кросівки	163	93	120
1483327672	Чоловічі зимові сірі кросівки Nike Air Force Winter	163	93	66
1988919381	Зимові Чоловічі Кросівки Nike Air Force 1 Grey (Хутро) 42-43	163	93	99
2010546355	Зимові кросівки Nike air чорно-білі	163	93	235
1725457355	Чоловічі зимові кросівки Nike Air Max 90 Sneakerboot Mid Winter Green кросівки найк аір макс 90 Ultra shark	163	93	209
2387651704	Зимові шкіряні кросівки на хутрі Nike Air Force Winter	163	93	224
2403841507	Кросівки зимові Nike Air Jordan 1 Retro High Black White Red Fur 36	163	93	99
1701259428	Зимові кросівки Nike Huarache X Acronym City MID-Winter хутро 43	163	93	85
2392767907	Зимовий чоловічий пуховик Nike nsw Пуховик Теплий пуховик Пуховик найк NSW Оригінальний пуховик Nike	163	93	207
2033807833	Чоловічі зимові черевики Nike ACG Air Zoom Gaiadome Gore-Tex GTX Brown Winter Termo, коричневі кросівки найк асг аір зум shark	163	93	231
2451554605	Зимові кросівки Nike P-6000 Gore-Tex (Фліс - 21*)	163	93	168
1872940488	Urbanshop com ua Кросівки Nike Zoomx Zegama Beige Dh0625-601 РОЗМІР ЗАПИТУЙТЕ	163	93	157
2490687274	Зимові кросівки чорного кольору Nike ACG Gaiadome	163	93	235
1763678286	Зимові кросівки Nike Huarache X Acronym City, кросівки найк хуарачи, зимові кросівки Nike Huarache Acronym shark	163	93	216
2428372889	Кросівки зимові NIKE Air Jordan 1 Retro High Grey Black White Fur 36 41	163	93	99
1674633280	Зимові шкіряні кросівки на хутрі Nike Air Force Winter	163	93	224

Рисунок 4.6 – Приклади товарів із кодом L0_163 і L1_93

Рівень взаємодії кодів, які створює RQ-VAE є доволі глибоким, коди використовуються тільки у контексті вже існуючого префіксу (L1_93 наприклад, у контексті одягу та взутті включає в собі зимні товари, а в контексті техніки і електроніки – товари фірми Sigma)

4.5.1 У чому перемагає генеративна рекомендаційна система

Два режими показують стабільну перевагу генеративної моделі. Перший стосується якості влучання, коли воно досягнуто. У таблиці 4.7 наведено два взаємодоповнювальні показники: частка всіх влучань, які потрапляють на першу позицію ранжованого списку, та відношення NDCG@10 до Recall@10 для всього оцінювання. Обидва показники є вищими для генеративної рекомендаційної системи приблизно на дванадцять-сорок відсотків. Декодер із променевим пошуком (beam-search decoder) явно перераховує найбільш імовірні послідовності ідентифікаторів у порядку спадання спільної ймовірності, тому знайдена ціль зазвичай знаходиться ближче до початку променя; щільний пошуковик виконує невпорядкований запит найближчих сусідів по геометрії контенту, і схожі елементи часто чергуються на проміжних позиціях.

Таблиця 4.7 – Позапитові переваги генеративної рекомендаційної

Показник	TIGER	Базовий алгоритм	Δ
Частка top-1	0.103	0.074	+39 %
NDCG@10/Recall@10	0.492	0.477	+3 %
Частка холодних передбачень	0.300	0.264	+14 %
Частка популярних передбачених	0.003	0.001	3×

Другий режим стосується виведення елементів на крайніх полюсах розподілу популярності. Генеративна рекомендаційна система передбачає елементи, які мали нуль взаємодій у навчальному періоді, приблизно у тридцяти відсотках своїх слотів у першій десятці (top-10 slots) проти двадцяти шести відсотків у щільної базової моделі; симетрично, вона

піднімає на поверхню найпопулярніші елементи (понад тисячу появ у навчанні) у трьох із кожних тисячі слотів, тоді як щільна базова модель не виводить їх практично ніколи. Ці два ефекти разом вказують на те, що генеративна модель не колапсує в бік голови розподілу популярності до такої міри, як це робить щільна базова модель.

4.5.2 У чому перемагає щільна базова модель

Переваги щільної базової моделі зосереджені в протилежному напрямку. У таблиці 4.8 наведено два режими, в яких вона є явно сильнішою. По-перше, прогнозований список є семантично більш зв'язним: передбачення top-1 щільної базової моделі збігається з категорією першого рівня цільового елемента приблизно у шістдесяти п'яти відсотках випадків проти п'ятдесяти семи відсотків у генеративної рекомендаційної системи, з меншим, але стабільним розривом на порозі першої десятки (rank-ten cutoff). Це пряма ціна авторегресійного декодування: модель видає токени Semantic-ID по одному за раз залежно від історії, а не від контенту елемента-кандидата, тому високоімовірне продовження токенів може перетнути межу категорії, чого не зробив би щільний пошуковик, заякорений у геометрії контенту.

Таблиця 4.8 – Позапитові переваги щільної базової моделі

Показник	TIGER	Базовий алгоритм	Δ
Узгодженість категорії (top-1)	0.571	0.652	-14 %
Узгодженість категорії (top-10)	0.709	0.743	-5 %
Recall@10 для товарів з частотністю 11-20	0.0056	0.0427	7.6×

По-друге, якість щільної базової моделі деградує менш різко зі збільшенням довжини сесії. На сесіях із максимум двома елементами в історії обидві моделі перебувають, по суті, у своєму найкращому режимі; на сесіях з одинадцятьма-дев'ятнадцятьма елементами Recall генеративної

рекомендаційної системи падає приблизно на дві третини, тоді як у щільної базової моделі він знижується менш ніж наполовину. Найскладніший режим – цільові елементи із проміжною частотою, які спостерігалися від трьох до дев'яти разів під час навчання – відкриває шестикратний розрив на користь щільної базової моделі, що є найбільшим показником серед усіх режимів в узгодженому порівнянні.

4.5.3 Синтез

Вище оглянуті чотири режими описують чітку взаємодоповнюваність, а не однорідне ранжування. Переваги генеративної рекомендаційної системи, зосереджені на початку списку ранжування та на крайніх полюсах розподілу популярності, є саме тими режимами, в яких очікується найкраща робота декодера з променевим пошуком над вивченою граматикою токенів: він обирає один найбільш імовірний ідентифікатор і не має зсуву в бік щільного центру мас каталогу. Переваги щільної базової моделі, зосереджені в узгодженості категорій та в режимі довгої історії, де її вежа запитів може вбирати багатший контекст, є саме тими режимами, для яких структурно добре підходить пошук найближчих сусідів, заякорений у контенті. Жоден із наборів переваг не домінує над іншим у сукупності, і на узгодженій тестовій вибірці жодна з моделей не знаходить ті елементи, які знаходить інша, більш ніж у незначній частині сесій, де вона демонструє хороші результати.

Практичний висновок із цієї декомпозиції полягає в тому, що належною одиницею порівняння між двома системами є не єдиний агрегований показник Recall, а стратегія пошуку, обумовлена конкретним режимом. Наслідки для проєктування систем – зокрема, для гібридного пошуку, який використовує кожну модель у її сильному режимі – розглядаються в розділі висновків.

ВИСНОВКИ

Поставлене завдання кваліфікаційної роботи виконано в повному обсязі. У роботі було реалізовано конвеєр генеративної рекомендаційної системи TIGER від початку до кінця, що охоплює побудову контентних ембеддингів, автокодування із залишковою квантизацією, формування Semantic-ID, навчання авторегресійного трансформера та декодування рекомендацій за допомогою променевого пошуку. Розроблену систему було оцінено на двох корпусах різного масштабу та структури: внутрішньому корпусі сесій електронної комерції Prom.ua з приблизно тридцятьма мільйонами товарів та відкритому датасеті Amazon Reviews 2023 Beauty.

Основним результатом роботи є успішне відтворення генеративного підходу TIGER та його порівняння зі щільною двовежевою базовою моделлю пошуку. Отримані результати показали, що ефективність генеративної рекомендаційної системи суттєво залежить від властивостей корпусу. На великому корпусі електронної комерції щільна базова модель перевершила генеративну систему приблизно у 3,5 раза за метрикою Recall@10. Натомість на меншому корпусі Amazon генеративна система показала кращий результат і перевершила базову модель приблизно у 5 разів.

Проведений діагностичний аналіз показав, що в режимі великого каталогу основним обмеженням генеративної системи є не бюджет променевого пошуку, а дискримінативна здатність кодів Semantic-ID. Значна частина втрат повноти виникає на глибокому рівні квантизатора через необхідність розв’язання колізій. Також було виявлено анізотропний спільний компонент у просторі ембеддингів, який негативно впливає на залишкову квантизацію. Центрування ембеддингів без витоку даних дозволило покращити повноту рекомендацій приблизно на 10% як для популярних елементів, так і для елементів холодного старту.

Якісний аналіз показав, що жодна з двох систем не є універсально кращою в усіх режимах. Генеративна рекомендаційна система краще працює на початку ранжованого списку та частіше виводить елементи холодного старту й товари з довгого хвоста. Водночас щільна базова модель формує більш узгоджені за категоріями результати та є стабільнішою для сесій із довшою історією. Це свідчить про те, що генеративний пошук доцільно розглядати не як повну заміну щільного пошуку, а як доповнювальний механізм для побудови гібридних рекомендаційних систем.

У співвідношенні зі світовими аналогами виконана розробка відповідає сучасному напряму генеративного пошуку в рекомендаційних системах, представленому підходами TIGER, Semantic-ID-based retrieval та двовежевими моделями пошуку. На відміну від багатьох академічних робіт, у цій роботі підхід було додатково перевірено на корпусі промислового масштабу з десятками мільйонів товарів. У межах проведеного огляду не було виявлено відкритих вітчизняних аналогів, які б відтворювали генеративний пошук із Semantic-ID на каталозі такого масштабу.

Новими результатами роботи є експериментальне встановлення корпусозалежного характеру переваг генеративного та щільного пошуку, виявлення обмежень Semantic-ID у режимі великого каталогу, а також показ позитивного впливу центрування ембеддингів на повноту рекомендацій. Отримані результати можуть бути використані як основа для подальших досліджень гібридних рекомендаційних систем. На момент завершення роботи результати не були оформлені у вигляді винаходів, однак можуть бути підготовлені до публікації у вигляді статті або тез конференції.

Виконана робота пов’язана з науково-дослідним напрямом випускової кафедри, що охоплює методи штучного інтелекту, машинного навчання, нейронних мереж, обробки великих даних та інтелектуального аналізу інформації. Результати роботи також узгоджуються з прикладними

дослідженнями у сфері інтелектуальних інформаційних систем та рекомендаційних сервісів.

Матеріали роботи можуть бути використані під час розроблення та дослідження рекомендаційних систем для електронної комерції. Окремі частини роботи, зокрема огляд літератури, опис архітектури TIGER, реалізований конвеєр, експериментальний протокол і діагностичний аналіз, можуть бути використані в навчальному процесі університету під час вивчення дисциплін, пов’язаних із машинним навчанням, нейронними мережами, рекомендаційними системами та обробкою великих даних.

Подальший розвиток роботи доцільно спрямувати на побудову гібридної архітектури пошуку, у якій генеративна модель використовується для формування кандидатів у тих частинах каталогу, де вона має переваги, а щільна модель виконує переранжування. Також перспективними напрямами є триваліше навчання генеративного трансформера, використання більшої моделі та повторне тестування підходу з маскованим суфіксом на центрованих ембеддингах.

Підсумовуючи, розроблена система демонструє, що відтворення TIGER є можливим у промисловому масштабі, однак переваги генеративного пошуку суттєво залежать від властивостей даних. Практична цінність роботи полягає не лише в реалізації системи, а й у виявленні режимів, у яких генеративний підхід працює найкраще або, навпаки, втрачає якість порівняно зі щільним пошуком.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1) E-commerce 1999. United State Department of Commerce. URL: https://www.census.gov/content/dam/Census/library/publications/2001/econ/1999estatstext.pdf (date of access: 30.03.2026)

2) Recommender systems with generative retrieval / S. Rajput et al. *Advances in neural information processing systems 36*, New Orleans, Louisiana, USA, 10–16 December 2023. San Diego, California, USA, 2023. P. 10299–10315. URL: https://doi.org/10.52202/075280-0452 (date of access: 27.03.2026).

3) Autoregressive image generation using residual quantization / D. Lee et al. *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, New Orleans, LA, USA, 18–24 June 2022. 2022. URL: https://doi.org/10.1109/cvpr52688.2022.01123 (date of access: 30.03.2026).

4) Neural discrete representation learning / A. van den Oord et al. *Advances in Neural Information Processing Systems 30*, Long Beach, California, USA, 4–9 December 2017. Red Hook, New York, USA, 2017. P. 6306–6315. https://doi.org/10.48550/arXiv.1711.00937 (date of access: 02.04.2026).

5) Attention is all you need / A. Vaswani et al. *Advances in neural information processing systems 30*, Long Beach, California, USA, 4–9 December 2017. Red Hook, New York, USA, 2017. P. 5998–6008. URL: https://doi.org/10.48550/arXiv.1706.03762 (date of access: 28.03.2026).

6) Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer / C. Raffel et al. Journal of machine learning research. 2020. Vol. 21, No. 140. P. 1–67. URL: https://doi.org/10.48550/arXiv.1910.10683 (date of access: 05.04.2026).

7) Sentence-T5: scalable sentence encoders from pre-trained text-to-text models / J. Ni et al. *Findings of the association for computational linguistics*

(ACL 2022), Dublin, Ireland, 22–27 May 2022. Stroudsburg, Pennsylvania, USA, 2022. P. 1864–1874. URL: https://doi.org/10.48550/arXiv.2108.08877 (date of access: 07.04.2026).

8) Reimers N., Gurevych I. Sentence-BERT: sentence embeddings using siamese bert-networks. *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, Hong Kong, China. Stroudsburg, PA, USA, 2019. URL: https://doi.org/10.18653/v1/d19-1410 (date of access: 07.04.2026).

9) Distributed Representations of Words and Phrases and Their Compositionality / T. Mikolov et al. *Advances in Neural Information Processing Systems 26*, Lake Tahoe, Nevada, USA, 5–10 December 2013. Red Hook, New York, USA, 2013. P. 3111–3119. URL: https://doi.org/10.48550/arXiv.1310.4546 (date of access: 11.04.2026).

10) Barkan O., Koenigstein N. ITEM2VEC: neural item embedding for collaborative filtering. *2016 IEEE 26th international workshop on machine learning for signal processing (MLSP)*, Vietri sul Mare, Salerno, Italy, 13–16 September 2016. 2016. URL: https://doi.org/10.1109/mlsp.2016.7738886 (date of access: 14.04.2026).

11) Kang W.-C., McAuley J. Self-attentive sequential recommendation. *2018 IEEE international conference on data mining (ICDM)*, Singapore, 17–20 November 2018. 2018. URL: https://doi.org/10.1109/icdm.2018.00035 (date of access: 17.04.2026).

12) BERT4Rec: sequential recommendation with bidirectional encoder representations from transformer / F. Sun et al. *Proceedings of the 28th ACM international conference on information and knowledge management (CIKM 2019)*, Beijing, China, 3–7 November 2019. New York, New York, USA, 2019. P. 1441–1450. URL: https://doi.org/10.48550/arXiv.1904.06690 (date of access: 17.04.2026).

13) Transformer memory as a differentiable search index / Y. Tay et al. *Advances in neural information processing systems 35*, New Orleans, Louisiana, USA, 28 November – 9 December 2022. Red Hook, New York, USA, 2022. P. 21831–21843. URL: https://doi.org/10.48550/arXiv.2202.06991 (date of access: 19.04.2026).

14) Actions speak louder than words: trillion-parameter sequential transducers for generative recommendations / J. Zhai et al. *Proceedings of the 41st international conference on machine learning (ICML 2024)*, Vienna, Austria, 21–27 July 2024. PMLR, 2024. Vol. 235. URL: https://doi.org/10.48550/arXiv.2402.17152 (date of access: 21.04.2026).

15) Better generalization with semantic ids: a case study in ranking for recommendations / A. Singh et al. *RecSys '24: 18th ACM conference on recommender systems*, Bari Italy. New York, NY, USA, 2024. P. 1039–1044. URL: https://doi.org/10.1145/3640457.3688190 (date of access: 22.04.2026).

16) Unifying generative and dense retrieval for sequential recommendation / L. Yang et al. *arXiv* preprint arXiv:2411.18814. 2024. URL: https://doi.org/10.48550/arXiv.2411.18814 (date of access: 25.04.2026).

17) DAS: dual-aligned semantic ids empowered industrial recommender system / Y. Liu et al. *arXiv* preprint arXiv:2508.10584. 2025. URL: https://doi.org/10.48550/arXiv.2508.10584 (date of access: 27.04.2026).

18) Bridging language and items for retrieval and recommendation / Y. Hou et al. *arXiv* preprint arXiv:2403.03952. 2024. URL: https://doi.org/10.48550/arXiv.2403.03952 (date of access: 29.04.2026).

19) Sampling-bias-corrected neural modeling for large corpus item recommendations / X. Yi et al. *RecSys '19: thirteenth ACM conference on recommender systems*, Copenhagen Denmark. New York, NY, USA, 2019. URL: https://doi.org/10.1145/3298689.3346996 (date of access: 02.05.2026).

20) PyTorch: an imperative style, high-performance deep learning library / A. Paszke et al. *Advances in neural information processing systems 32*, Vancouver, British Columbia, Canada, 8–14 December 2019. Red Hook, New

York, USA, 2019. P. 8024–8035. URL: https://doi.org/10.48550/arXiv.1912.01703 (date of access: 05.05.2026).

21) Is generative recommendation the ChatGPT moment of RecSys? / Y. Meng. 2025. URL: https://www.yuan-meng.com/posts/generative_recommendation/ (date of access: 08.05.2026).

22) Industry 4.0 vs. industry 5.0: co-existence, transition, or a hybrid / M. Golovianko et al. *Procedia computer science*. 2023. Vol. 217. P. 102–113. URL: https://doi.org/10.1016/j.procs.2022.12.206 (date of access: 17.05.2026).