МЕЛІТОПОЛЬСЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ БОГДАНА ХМЕЛЬНИЦЬКОГО

факультет інформатики, математики та економіки

кафедра інформатики і кібернетики

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «магістр»

на тему: «**Розробка інтелектуальної системи рекомендацій для інтернет-магазинів на основі поведінки користувачів**»

Виконав: здобувач вищої освіти
групи М314-і
спеціальність 122 Комп'ютерні науки
ОПП Комп'ютерні науки
Андреєв Андрії Валерійович

Керівник: доктор філософії,
Сіциліцин Юрій Олександрович

Рецензент, д.т.н., професор
кафедри «Комп'ютерні науки»
ТДАТУ імені Дмитра Моторного
Малкіна Віра Михайлівна

Запоріжжя – 2025 року

АНДРЕЄВ АНДРІЇ ВАЛЕРІЙОВИЧ

РОЗРОБКА ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ РЕКОМЕНДАЦІЙ ДЛЯ ІНТЕРНЕТ-МАГАЗИНІВ НА ОСНОВІ ПОВЕДІНКИ КОРИСТУВАЧІВ

Анотація. Кваліфікаційна робота присвячена розробці інтелектуальної системи рекомендацій для інтернет-магазинів на основі поведінки користувачів. Актуальність дослідження обумовлена зростанням обсягів інформації та кількості товарів у електронній комерції, що ускладнює процес вибору покупцями релевантних продуктів. Рекомендаційні системи дозволяють автоматично аналізувати поведінку користувачів і формувати персоналізовані пропозиції, підвищуючи ефективність онлайн-продажів, лояльність клієнтів та рівень конверсії.

У роботі проведено аналіз сучасних методів і алгоритмів рекомендаційних систем, зокрема контентних, колаборативних та гібридних підходів. Особлива увага приділена удосконаленню алгоритму DeepCoNN для роботи з текстовими відгуками користувачів і характеристиками товарів, що дозволяє формувати більш точні та персоналізовані рекомендації. Проведено порівняльний аналіз ефективності запропонованого підходу з іншими популярними алгоритмами.

Розроблена система має модульну структуру, яка включає збір та обробку даних, генерацію рекомендацій, оцінку точності та інтеграцію з інтерфейсом інтернет-магазину. Практична користь полягає у можливості інтеграції системи в реальні інтернет-магазини для підвищення точності рекомендацій та поліпшення досвіду користувачів.

Запропоновані результати можуть бути використані для подальшого розвитку систем персоналізованих рекомендацій та впровадження адаптивних алгоритмів, які враховують зміни вподобань користувачів і нові тенденції ринку.

Ключові слова: алгоритм персоналізації, інтелектуальна система, метрики точності, модель глибокого навчання

ANDREIEV ANDRII

DEVELOPMENT OF AN INTELLIGENT RECOMMENDATION SYSTEM FOR ONLINE STORES BASED ON USER BEHAVIOR

Abstract. The qualification work is devoted to the development of an intelligent recommendation system for online stores based on user behavior. The relevance of the study is due to the growth of information volumes and the number of products in e-commerce, which complicates the process of choosing relevant products by buyers. Recommendation systems allow you to automatically analyze user behavior and form personalized offers, increasing the efficiency of online sales, customer loyalty and conversion rate.

The work analyzes modern methods and algorithms of recommendation systems, in particular content, collaborative and hybrid approaches. Particular attention is paid to improving the DeepCoNN algorithm for working with text user reviews and product characteristics, which allows you to form more accurate and personalized recommendations. A comparative analysis of the effectiveness of the proposed approach with other popular algorithms is conducted.

The developed system has a modular structure that includes data collection and processing, recommendation generation, accuracy assessment and integration with the online store interface. The practical benefit lies in the possibility of integrating the system into real online stores to increase the accuracy of recommendations and improve the user experience.

The proposed results can be used for further development of personalized recommendation systems and implementation of adaptive algorithms that take into account changes in user preferences and new market trends.

Keywords: personalization algorithm, intelligent system, accuracy metrics, deep learning model

ЗМІСТ

ВСТУП.....3
РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ РОЗРОБКИ РЕКОМЕНДАЦІЙНИХ СИСТЕМ.....6
1.1. Актуальність задачі надання рекомендацій.....6
1.2. Аналіз методів надання персоналізованих рекомендацій.....8
1.3. Етапи процесу рекомендацій та використання рекомендаційних систем для е-комерції.....11
1.4. Аналіз існуючих сервісів з надання рекомендацій.....13
РОЗДІЛ 2 МЕТОДИ І МОДЕЛІ РОЗРОБКИ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ РЕКОМЕНДАЦІЙ ДЛЯ ІНТЕРНЕТ-МАГАЗИНІВ НА ОСНОВІ ПОВЕДІНКИ КОРИСТУВАЧІВ.....17
2.1. Моделі глибоких нейронних мереж для формування рекомендацій.....17
2.2. Методи навчання глибоких нейронних мереж.....27
2.3. Критерії якості задачі надання рекомендацій.....29
2.4. Пропонована модифікація нейромережевої моделі формування рекомендацій та її обґрунтування.....32
РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МОДЕЛЕЙ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ФОРМУВАННЯ РЕКОМЕНДАЦІЙ.....39
3.1. Дослідження ефективності запропонованого методу.....39
3.2. Аналіз отриманих результатів.....46
ВИСНОВКИ.....59
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....61

ВСТУП

Актуальність дослідження. У сучасних умовах стрімкого розвитку електронної комерції персоналізація взаємодії з користувачами стає ключовим фактором конкурентоспроможності інтернет-магазинів. Покупці очікують індивідуальний підхід, зручність, швидкий пошук товарів та релевантні пропозиції. Рекомендаційні системи, що аналізують поведінку користувачів, дозволяють суттєво підвищити ефективність онлайн-продажів, збільшити конверсію, середній чек та лояльність клієнтів.

Попри наявність великої кількості інструментів для рекомендацій, значна частина з них потребує складної інтеграції, не враховує детальні поведінкові характеристики користувачів або є закритими комерційними рішеннями з високою вартістю. Це ускладнює їх використання малим та середнім бізнесом. У зв'язку з цим виникає потреба у створенні гнучкої, інтелектуальної та доступної системи рекомендацій, яка працює на основі аналізу реальної поведінки користувачів. Саме це визначає актуальність обраної теми.

Об'єкт дослідження – інтелектуальна система рекомендацій для інтернет-магазинів.

Предмет дослідження – методи та алгоритми персоналізації товарних пропозицій на основі поведінкових даних користувачів. Мета дослідження – розробити програмний додаток, що формує персоналізовані рекомендації товарів, враховуючи дії й інтереси користувачів інтернет-магазину.

Завдання дослідження

1. Проаналізувати сучасні підходи до побудови рекомендаційних систем та методи персоналізації в електронній комерції.
2. Розробити модель інтелектуальної системи рекомендацій, здатної формувати пропозиції на основі поведінкових даних користувачів.
3. Порівняти існуючі рішення з бізнес-вимогами щодо автоматизації процесу рекомендацій.
4. Вибрати інструменти та технології для реалізації системи з

урахуванням її гнучкості, масштабованості та продуктивності.

5. Розробити та реалізувати систему рекомендацій, протестувати її та оцінити отримані результати.

6. Підготувати користувацьку інструкцію, що описує порядок роботи та налаштування системи.

Теоретичне значення полягає у вивченні алгоритмів рекомендаційних систем, методів збору та обробки поведінкових даних, а також у дослідженні сучасних технологій створення систем персоналізації. Особлива увага приділяється контентним, колаборативним і гібридним методам, а також моделям глибокого навчання, зокрема DeepCoNN та його модифікаціям.

Практичне значення полягає у створенні інтелектуальної системи рекомендацій, яка автоматично аналізує поведінку користувачів, формує персоналізовані товарні пропозиції та може бути інтегрована в інтернет-магазин без суттєвих змін його структури. Розроблена система має модульну архітектуру, що включає:

- модуль збору даних;
- модуль обробки та нормалізації даних;
- модуль генерації рекомендацій на основі Collaborative Filtering, Content-Based Filtering та DeepCoNN;
- модуль оцінки точності рекомендацій;
- модуль інтеграції з інтерфейсом інтернет-магазину.

Апробація дослідження

Основні положення та результати дослідження були обговорені під час внутрішніх консультацій на кафедрі, а також використані при підготовці презентаційних матеріалів до захисту кваліфікаційної роботи. Запропоновані підходи та рішення отримали позитивні відгуки наукового керівника та були рекомендовані для подальшого використання й удосконалення.

Структура роботи. Дипломна робота складається зі вступу, трьох розділів, висновків і списку використаних джерел.

Перший розділ присвячено огляду предметної області, аналізу підходів і

алгоритмів, що лежать в основі рекомендаційних систем.

Другий розділ містить обґрунтування вибору технологій, опис архітектури та функціональних модулів системи.

Третій розділ охоплює реалізацію програмного забезпечення, опис алгоритмів роботи та процес інтеграції з інтерфейсом інтернет-магазину.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ РОЗРОБКИ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

1.1. Актуальність задачі надання рекомендацій

Сьогодні рекомендаційні системи відіграють надзвичайно важливу роль у роботі багатьох великих технологічних компаній. Вони допомагають персоналізувати контент на таких платформах, як YouTube, Spotify, Facebook, Netflix, IMDb та багатьох інших. Завдяки цим системам користувачі отримують пропозиції фільмів, музики, книг чи інших матеріалів, які максимально відповідають їхнім інтересам і вподобанням. Крім того, рекомендаційні механізми дають можливість відкривати нові цікаві проєкти та продукти, про які користувачі могли раніше й не знати, що робить досвід взаємодії з платформою більш насиченим та індивідуалізованим.

Важливим етапом розвитку рекомендаційних систем став конкурс Netflix Prize, який компанія Netflix оголосила у 2006 році. Завдання полягало в тому, щоб перевершити існуючу систему Cinematch за показником RMSE на 10%. Для учасників був наданий унікальний на той час датасет, що містив понад 100 мільйонів оцінок користувачів. Незважаючи на видиму простоту завдання, значного покращення вдалося досягти лише через три роки. При цьому переможний алгоритм так і не впровадили у виробництво, оскільки його реалізація виявилася надто складною та ресурсомісткою у порівнянні з уже існуючою системою. Цей приклад показує, що навіть потужні алгоритми потребують оптимізації для реального використання.

Рекомендаційні системи мають велике значення і в електронній комерції. На таких платформах, як Amazon, вони виконують роль віртуального консультанта, автоматично пропонуючи користувачам супутні або альтернативні товари. Це не лише допомагає бізнесу збільшувати прибутки,

але й значно полегшує вибір користувачам, особливо коли асортимент продукції дуже широкий і постійно оновлюється.

Такий підхід дозволяє економити час покупців і одночасно підвищувати ефективність роботи компанії.

На перший погляд може здатися, що головна мета рекомендаційних систем полягає виключно у збільшенні доходів компаній. Однак користь від таких систем отримують і самі користувачі. Сучасний ринок пропонує величезну кількість товарів та контенту, які постійно оновлюються. Без систем рекомендацій покупцям складно швидко знайти те, що максимально відповідає їхнім потребам і вподобанням. Завдяки рекомендаціям вони отримують саме ті продукти, які дійсно їм цікаві, що робить процес вибору більш комфортним і ефективним.

Щороку у сфері рекомендаційних систем проходять дві важливі події, де науковці представляють свої новітні розробки: конференція ACM Conference Series on Recommender Systems та міжнародне змагання RecSys Challenge. Ці заходи дозволяють обмінюватися досвідом, представляти нові алгоритми та обговорювати сучасні тенденції розвитку рекомендаційних механізмів.

Актуальність цієї теми постійно зростає, адже рекомендаційні механізми можна інтегрувати майже у будь-яку платформу чи бізнес-модель. Водночас існують суттєві виклики, які залишаються предметом активних досліджень. Серед них виділяють:

проблему холодного старту, коли новим користувачам або новим продуктам складно запропонувати якісні рекомендації через брак даних;

проблему масштабованості, яка полягає в необхідності забезпечити ефективну роботу системи при постійному зростанні кількості користувачів, товарів і взаємодій.

Розробка алгоритмів, здатних ефективно вирішувати ці проблеми, залишається відкритим і надзвичайно важливим завданням у науково-практичній сфері. Успішне вирішення цих задач дозволяє створювати більш

точні, персоналізовані та ефективні рекомендаційні системи, які підвищують якість взаємодії користувачів із сервісами та сприяють розвитку бізнесу.

1.2. Аналіз методів надання персоналізованих рекомендацій

Рекомендації продуктів можна умовно розділити на два основні типи: персоналізовані та неперсоналізовані.

Неперсоналізовані рекомендації досить прості. Зазвичай вони пропонують користувачу найпопулярніші товари або ті, що відповідають загальним бізнес-цілям. Зрозуміло, що такі рекомендації не завжди бувають корисними для конкретного користувача. Тому їх найчастіше використовують як один із елементів більш складної системи, де поєднують кілька алгоритмів, щоб підвищити точність пропозицій.

Персоналізовані рекомендації, навпаки, враховують індивідуальні вподобання користувачів. Їх поділяють на чотири основні типи: Content-based, Case-based, Collaborative filtering і гібридні підходи, які поєднують два або більше з цих методів. Перші три ми розглядаємо зараз, а гібридні підходи будуть детально описані у пункті 2.3, де ми обговорюватимемо сучасні алгоритми рекомендаційних систем.

Колаборативна фільтрація – це підхід, коли користувачів об’єднують за схожістю їхніх вподобань. Ідея досить проста: якщо люди з подібними смаками оцінили певні товари позитивно, інші користувачі з цієї ж групи, ймовірно, теж їх оцінять. Метод ґрунтується на трьох припущеннях: по-перше, люди з подібними інтересами роблять схожі вибори; по-друге, їхні вподобання залишаються відносно стабільними; по-третє, можна робити прогноз про майбутній вибір користувача на основі його попередніх оцінок і взаємодій [2].

Щоб визначити схожість користувачів, у колаборативній фільтрації використовуються спеціальні математичні формули. Вони дозволяють зрозуміти, наскільки схожі вподобання двох людей, і на основі цього

формувати рекомендації. Саме завдяки цьому підходу система може пропонувати товари, які дійсно будуть цікаві конкретному користувачу.

$$\mathrm{sim}(i,j) = \frac{\sum_{u \in U_{ij}} (r_{u,i} - \overline{r}_u)(r_{u,j} - \overline{r}_u)}{\sqrt{\sum_{u \in U_{ij}} (r_{u,i} - \overline{r}_u)^2} \sqrt{\sum_{u \in U_{ij}} (r_{u,j} - \overline{r}_u)^2}}$$

де $u \in U$ – множина користувачів, $i \in I$ – множина об'єктів,(r_ui, u, i)$\in D$ – множина подій або діяльність, яку виконують над об'єктами.

Для того, щоб рекомендаційні системи працювали ефективно, потрібна велика кількість даних. Особливо це стосується колаборативної фільтрації. Якщо користувачі оцінюють мало товарів або майже не взаємодіють із платформою, система просто не встигає зрозуміти їхні вподобання. У результаті з'являються нерелевантні або навіть дивні рекомендації. Саме через це колаборативна фільтрація найбільше страждає від так званої проблеми «холодного старту».

Методи колаборативної фільтрації діляться на два основні підходи: user-based та item-based.

У user-based підході система визначає схожість між користувачами. Наприклад, якщо користувач А та користувач Б мають схожі оцінки на книги та фільми, система може запропонувати А ті товари, які сподобалися Б.

У item-based підході оцінюється схожість між товарами. Наприклад, якщо користувач купив смартфон певної моделі, система може запропонувати йому аксесуари або інші смартфони з подібними характеристиками.

Схожість користувачів або товарів обчислюється різними способами, наприклад через метрики відстані або кореляцію, і це дозволяє системі точніше формувати рекомендації. Сучасні системи часто комбінують

обидва підходи в гібридних моделях, щоб одночасно враховувати схожість і між користувачами, і між товарами.

Другий тип рекомендацій — контентно-орієнтовані системи (content-based). Тут пропозиції формуються на основі характеристик самих товарів: їхніх властивостей, складу чи опису. Наприклад, якщо користувач купує книгу жанру фантастики, система буде пропонувати йому інші фантастичні книги або твори того ж автора. Такий метод добре працює, коли про продукт відомо багато, а про користувача — мало. Проте є і недоліки: система може потрапити в інформаційну «бульбашку» — вона буде пропонувати тільки те, що схоже на те, що користувач вже оцінював, і різноманітність рекомендацій зменшується.

Також тут теж є проблема холодного старту: новий користувач повинен взаємодіяти з кількома товарами, щоб система зрозуміла його вподобання.

Третій тип — knowledge-based або експертні системи. Вони ґрунтуються на знаннях про предметну область, товари і користувачів. Наприклад, у виборі житла або автомобіля система може враховувати бюджет користувача, технічні характеристики та інші пріоритети. До таких систем належать:

– case-based — пропонує рішення на основі схожих попередніх випадків;

– demographic-based — враховує демографічні характеристики користувача, наприклад вік або місце проживання;

– utility-based — намагається максимізувати користь або цінність для користувача;

– critique-based — система враховує зауваження користувача до поточної рекомендації і пропонує новий варіант.

Основна перевага knowledge-based систем у тому, що вони дозволяють обробляти складні запити і враховувати специфічні вимоги користувача. Проте вони дуже спеціалізовані: створену систему важко

перенести на іншу сферу, а розробка потребує участі експерта та значних ресурсів.

Щоб компенсувати недоліки окремих підходів, у сучасних рекомендаційних системах часто використовують гібридні моделі.

Найпоширеніші способи комбінування алгоритмів такі:

– Колаборативні та контентні методи працюють окремо, а результати об’єднуються.

– Правила контентної фільтрації інтегруються в колаборативну модель.

– Елементи колаборативної фільтрації додаються до контентної методики.

Створюється єдина модель, яка одночасно враховує принципи обох підходів. Гібридні системи дозволяють зменшити проблеми холодного старту та ефект «бульбашки», одночасно підвищуючи точність і різноманітність рекомендацій

1.3. Етапи процесу рекомендацій та використання рекомендаційних систем для е-комерції

Етап збору інформації На першому етапі відбувається збір даних, необхідних для формування профілю користувача. Це можуть бути особисті атрибути користувача, його поведінка на платформі або характеристики контенту, з яким він взаємодіє. До моменту створення повного профілю система не здатна пропонувати релевантні рекомендації, тому важливо накопичити якомога більше інформації про користувача.

Для цього використовуються різні типи даних:

Явний зворотний зв’язок (explicit feedback) — користувач сам оцінює товари або контент (наприклад, ставить рейтинг чи залишає відгук). Точність рекомендацій напряму залежить від кількості наданих оцінок. Цей метод вважається найбільш надійним, але потребує додаткових дій від користувача.

Неявний зворотний зв’язок (implicit feedback) — система визначає вподобання користувача на основі його поведінки: переглядів, історії покупок, часу, проведеного на сторінках, кліків тощо. Цей метод не вимагає активної участі користувача, проте точність рекомендацій може бути нижчою.

Гібридний зворотний зв’язок — поєднує явний та неявний підходи. Наприклад, неявні дані можуть перевіряти об’єктивність явних оцінок, або явні відгуки пропонуються лише за бажанням користувача. Такий підхід дозволяє мінімізувати недоліки кожного методу і підвищити ефективність системи

Етап навчання На цьому етапі рекомендаційна система «вчиться» робити передбачення на основі зібраних даних. Використовуються алгоритми навчання, які дозволяють визначати вподобання користувача та прогнозувати, що йому може бути цікаво. Конкретні методи навчання залежать від типу системи: колаборативна фільтрація, контентно-орієнтовані підходи, гібридні моделі тощо.

Етап формування рекомендацій Після навчання система на основі отриманих знань пропонує користувачу ті елементи, які, ймовірно, його зацікавлять. Користувач отримує рекомендації, а його подальша поведінка або оцінки використовуються для корекції та оптимізації наступних пропозицій. Це дозволяє системі динамічно вдосконалювати свої рекомендації та підвищувати їх точність з часом.

Збільшення продажів Одна з головних переваг рекомендаційних систем у комерційних проєктах — це можливість збільшувати продажі. Наприклад, коли система показує «часто купують разом з цим товаром», користувачі часто додають додаткові товари у кошик. Зазвичай ці речі дійсно відповідають потребам покупців.

Більша активність користувачів Це важливо не лише для магазинів. Наприклад, платформи на кшталт YouTube заробляють на рекламі, тому їм

вигідно показувати користувачам максимально цікавий і релевантний контент.

Це змушує людей довше залишатися на сайті і активніше взаємодіяти з платформою.

Розширення вибору товарів Рекомендації допомагають знайти речі або послуги, які складно помітити без додаткових підказок. Важливо показувати не тільки популярні товари, а й менш відомі, які підходять конкретному користувачу.

Підвищення лояльності Користувачі більше довіряють платформам, які враховують їхні вподобання. Персоналізовані рекомендації роблять сайт зручнішим, що підвищує задоволеність та бажання повертатися знову.

Краще розуміння аудиторії На основі історії поведінки користувачів системи можуть прогнозувати, що буде користуватися попитом у майбутньому, планувати закупівлі та допомагати у маркетингових та аналітичних завданнях.

Особливості роботи рекомендаційних систем у e-commerce Зворотний зв’язок може бути двох типів:

Явний – коли користувач оцінює товар (наприклад, від 1 до 5) або позначає «сподобалось / не сподобалось».

Неявний – коли система збирає інформацію про поведінку користувача: історію покупок, переглядів, відгуки тощо.

Явні оцінки корисні для точних рекомендацій, але користувачі рідко їх залишають. Тому сучасні системи намагаються максимально використовувати неявні сигнали, щоб рекомендації були кращими.

1.4. Аналіз існуючих сервісів з надання рекомендацій

Як уже зазначалося раніше, сьогодні рекомендаційні системи практично

скрізь у бізнесі. Далі наведу кілька прикладів, як вони використовуються на практиці:

– IMDB – це сайт про фільми та серіали, де можна знайти все: від акторів та режисерів до рецензій і оцінок користувачів. Рекомендації тут персоналізовані: система пропонує фільми та серіали, ґрунтуючись на твоїх попередніх оцінках і смаках людей, які схожі на тебе. Тобто, якщо хтось з подібними вподобаннями вже оцінив фільм високо, тобі його теж порадять.

– Netflix – потоковий сервіс, який не тільки показує популярні фільми, а й пропонує схожі до того, що ти вже дивився. Тут працює item-based колаборативна фільтрація: система порівнює схожість відео між собою. Також Netflix враховує твої оцінки й вподобання, щоб підбирати контент під твій смак.

– LinkedIn – соцмережа для професійного спілкування. Тут система радить людей для контактів, вакансії, групи та компанії, які можуть бути цікавими. Все працює на основі колаборативної фільтрації з використанням великих даних через Apache Hadoop.

– Spotify – музичний сервіс, який аналізує сотні характеристик пісень і виконавців, щоб формувати плейлисти з урахуванням твого смаку. Оцінки «сподобалося / не сподобалося» теж враховуються — система підлаштовується під тебе.

– YouTubc – відсохостинг, який використовує два типи рскомсндацій: неперсоналізовані (трендові відео) і персоналізовані (з урахуванням твоєї історії переглядів).

– **Steam** – платформа для комп'ютерних ігор. Система підбирає ігри на основі того, що ти вже грав, а також враховує популярність, рейтинги, відгуки друзів і топи продажів.

– **Amazon** – величезний онлайн-магазин. Тут система пропонує товари, схожі на ті, що вже купували люди з подібними покупками. Це реалізовано через item-to-item колаборативну фільтрацію.

– **Gifts** When You Need (1-800-Flowers.com) – сервіс-консьєрж із штучним інтелектом. Ти вводиш дані про людину, якій хочеш подарунок, а система підбирає варіанти, виходячи з того, що купували інші схожі користувачі.

– North Face – компанія радить куртки, враховуючи, де ти живеш, який сезон, твої гендерні особливості та інші параметри.

Отже, прикладів того, як використовуються рекомендаційні системи, багато. Вони бувають різні: колаборативні, контентно-орієнтовані, гібридні та навіть спеціалізовані під конкретні бізнес-завдання. Головне — допомагають користувачеві знайти те, що йому реально цікаво, а бізнесу — збільшити продажі і лояльність.

Отже, ми проаналізували, наскільки актуальною є задача надання рекомендацій продуктів користувачам, а також розглянули історію розвитку рекомендаційних систем — від 1994 року до сучасності. Крім того, були наведені приклади використання таких систем у бізнесі та проаналізовано їхній вплив на ефективність комерційних і некомерційних сервісів.

Розглянуто основні підходи до побудови рекомендаційних систем:

Неперсоналізовані рекомендації – ґрунтуються на популярності товарів або на тому, наскільки вони відповідають цілям бізнесу;

Персоналізовані рекомендації, які поділяються на:

на основі колаборативної фільтрації;

на основі контенту;

на основі знань (експертні системи);

Гібридні підходи, які поєднують різні методики, щоб мінімізувати недоліки окремих підходів.

Було встановлено, що кожен метод має свої плюси та мінуси, тому на практиці часто використовують гібридні моделі.

Також досліджено сучасні тенденції розвитку рекомендаційних систем, зокрема:

– застосування методів машинного навчання для підвищення точності та швидкості персоналізації;

– використання аналізу відгуків користувачів для поліпшення рекомендацій;

– прогнозування рейтингу переходів та контекстно-орієнтовані рекомендації;

– застосування автокодерів для побудови рекомендацій і реалізації

– послідовних рекомендацій.

На основі проведеного аналізу були сформовані завдання, цілі та конкретні кроки щодо розробки рекомендаційної системи, які стали вихідною точкою для наступних розділів роботи.

РОЗДІЛ 2 МЕТОДИ І МОДЕЛІ РОЗРОБКИ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ РЕКОМЕНДАЦІЙ ДЛЯ ІНТЕРНЕТ-МАГАЗИНІВ НА ОСНОВІ ПОВЕДІНКИ КОРИСТУВАЧІВ

2.1. Моделі глибоких нейронних мереж для формування рекомендацій

У цьому розділі ми переходимо від теорії до практики і розглядаємо, як насправді створюється інтелектуальна система рекомендацій для інтернет-магазинів, яка враховує поведінку користувачів. Після того, як ми проаналізували сучасні підходи і методи рекомендацій, потрібно показати, як їх можна реалізувати у вигляді працюючої системи. Тут ми детально пояснюємо, чому обрано саме такий технологічний стек, як побудована нейромережева модель і як обробляються дані, щоб рекомендації були швидкими, точними і максимально персоналізованими.

Система працює не просто з оцінками чи покупками користувачів. Вона вивчає історію взаємодій – перегляди, кліки, покупки – а також тексти відгуків і характеристики товарів. Наприклад, якщо користувач залишив відгук, де похвалив легкість використання конкретного гаджета, система враховує ці слова при прогнозі, які ще товари йому можуть сподобатися. Щоб нейромережа могла аналізувати такі дані, ми спершу проводимо *Textual Preprocessing*: токенізацію, нормалізацію, видалення зайвих слів і перетворення текстів у числові вектори за допомогою Word2vec. Це дозволяє системі “розуміти” семантичне значення слів і знаходити схожі контексти у відгуках інших користувачів.

Ми обрали бібліотеку *PyTorch*, бо вона дає свободу експериментувати з різними архітектурами нейромереж, легко змінювати шари, тестувати гіперпараметри і швидко запускати навчання моделей на GPU. Архітектура моделі побудована так, щоб інтегрувати різні типи даних: поведінку користувачів, характеристики товарів і тексти відгуків. Особливо корисним

виявився *Attention-шар*, який дозволяє виділяти найважливіші ознаки у текстах, враховуючи, що не всі слова або відгуки однаково важливі для прогнозу.

Уявімо, що користувач часто купує спортивний одяг і залишає відгуки з коментарями про якість матеріалу. Модель за допомогою Attention-шару може підсвітити ключові слова з цих відгуків і запропонувати нові товари, схожі за характеристиками, навіть якщо користувач раніше їх не переглядав. Завдяки такому підходу система не просто повторює минулі покупки, а передбачає потенційні інтереси.

Мета цього розділу – створити технологічну базу для програмної реалізації моделі, щоб потім можна було оцінити її ефективність і порівняти з існуючими підходами. В результаті система повинна пропонувати користувачу релевантні товари, підвищувати його залученість і стимулювати продажі, роблячи взаємодію з інтернет-магазином максимально персоналізованою та комфортною.

На рис.2.1 показано архітектуру гібридної моделі колаборативної фільтрації та глибинного навчання Neu-CF

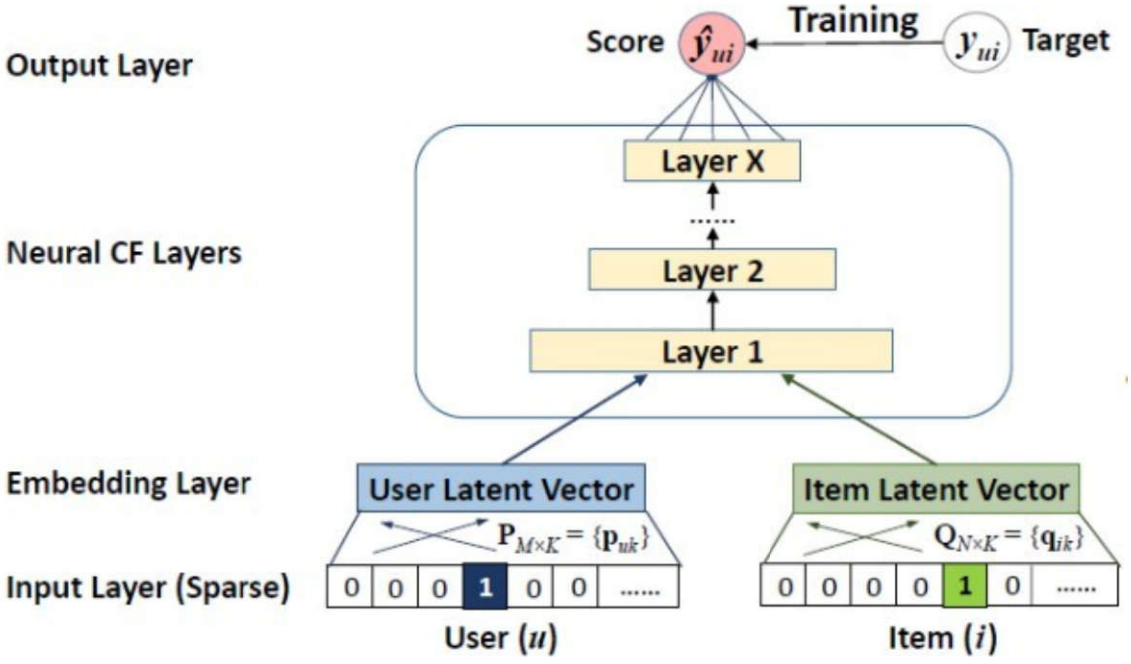


Рис.2.1 – Архітектура моделі Neu-C

Результат моделі Neu-CF:

$$\hat{y}_{ui} = f(P^T v_U, Q^T v_I \mid P, Q, \Theta)$$

де P, Q – матриці для прихованих факторів відповідно користувача і об'єкта. Функція f представляє багатошарову нейронну мережу, тому її можна описати наступним чином:

$$f(P^T v_U, Q^T v_I) = \phi_{\text{out}}\Big(\phi_L(\dots \phi_2(\phi_1(P^T v_U, Q^T v_I))\dots)\Big)$$

де phi_{out}, phi_x визначають взаємозв'язок вихідного шару з x-м шаром.

На рис.2,2 показано гібридну модель Neu-MF.

Останній прихований шар виконує поєднання (Concatenation) результатів двох підмоделей, які представлені виходами шарів GMF Layer та MLP Layer 1, 2, …, X.

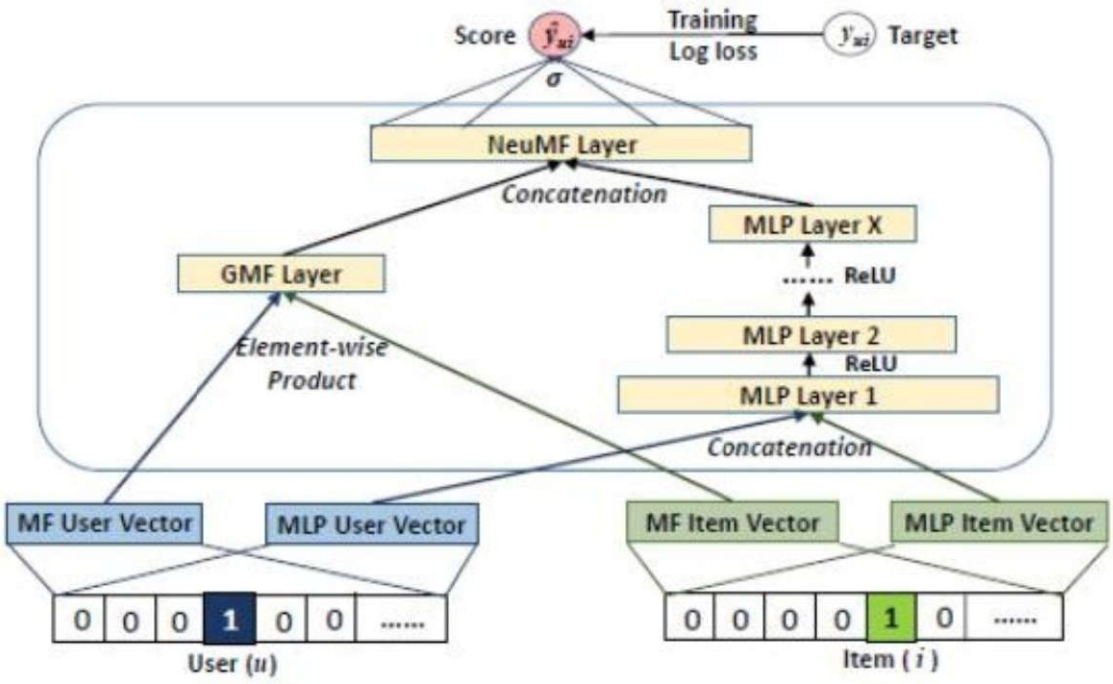


Рисунок 2.2 - Архітектура моделі Neu-MF

LightGCN – це графова нейронна мережа, яку створили, щоб спростити Neural Graph Collaborative Filtering (NGCF), і при цьому вона навіть точніша. Головна фішка LightGCN у тому, що тут немає матриць ваг і складних нелінійних функцій активації. Тобто навчаються тільки початкові ембедінги для кожного вузла графа. Через це навчання йде набагато швидше.

Працює LightGCN досить просто: він «збирає» інформацію від сусідніх вузлів і поширює її по графу. Це означає, що ембедінги кожного вузла оновлюються на основі того, що відбувається з його сусідами. Такий підхід дозволяє мережі краще розуміти структуру графа і давати більш точні рекомендації.

$$x_i^{(k+1)} = \sum_{j \in N(i)} \frac{1}{\sqrt{|N(i)|}\sqrt{|N(j)|}} x_j^{(k)}$$

Де:

$x_i^{(k)}$ — вектор ознак вузла iii на kkk-му шарі (ітерації).

$N(i)$ — множина сусідів вузла iii (можливо включно з самим вузлом iii, якщо враховуємо self-loop).

$|N(i)|$ — кількість сусідів вузла iii.

$\frac{1}{\sqrt{|N(i)|}\sqrt{|N(j)|}}$ значение узла jjj на шаге kkk. Це гарантує що внесок сусідів з різною кількістю зв’язків масштабується, щоб уникнути занадто великих або малих значень (це symmetrically normalized adjacency matrix).

LightGCN – графова нейронна мережа, яка працює досить просто і зрозуміло. Вона оновлює представлення вузла, беручи до уваги інформацію від його сусідів у графі. Проблема полягає в тому, що при цьому мережа повністю відкидає початкові дані про користувача або об’єкт. Це може призвести до втрати індивідуальних вподобань користувача або особливостей

об’єкта. Щоб уникнути цього, деякі дослідження [5] пропонують комбінувати початкове і агреговане представлення вузла через підсумовування або середнє.

Таким чином, мережа одночасно зберігає особисту інформацію вузла і враховує контекст сусідів.

У 2017 році з’явилася Neural Collaborative Filtering (NCF) [6]. Ідея дуже проста: нейронна мережа здатна навчитися будь-яким залежностям між користувачами і товарами, які раніше моделювалися лише через матричну факторизацію. NCF створює ембедінги для користувачів та об’єктів, а потімбагатошаровий перцептрон «вчиться» знаходити складні взаємозв’язки між ними. Тобто замість того, щоб обмежуватися лінійною матрицею, модель здатна враховувати нелінійні залежності, які набагато точніше відображають реальну поведінку користувачів.

DeepFM поєднує два підходи одночасно: факторизацію матриць і глибоку нейронну мережу. Всі категоріальні ознаки перетворюються у формат one-hot і проходять через спільний шар ембедінгів. Потім обчислюються взаємні добутки ембедінгів і векторів, які пройшли через нейронку, а результат об’єднується і подається на сигмоїду для отримання ймовірності рекомендації. Модель DLRM спрощує DeepFM, прибираючи обов’язковий перцептрон для ембедінгів, що робить її швидшою та менш ресурсоємною.

Для роботи з неповними вподобаннями користувачів застосовують Neural Tangent Kernel [7], а модель DISTILL-CF дозволяє виділяти найбільш важливі ознаки з великих розріджених матриць взаємодій. Тут використовується диференційована вибірка Гамбеля, яка дозволяє ефективно обробляти неоднорідні дані і масштабуватися для великих наборів інформації.

У [2] запропоновано модифікований метод колаборативної фільтрації для електронної комерції, який працює в три етапи:

Відбір надійних даних, де оцінюють користь відгуку;

Обчислення подібності користувачів через коефіцієнт Пірсона;

Інтеграція подібності у вдосконалений алгоритм нахилу один.

Для рекомендацій новин у [8] була створена модель CHAMELEON. Вона складається з двох частин: згорткова мережа аналізує текст статей і виділяє суттєві ознаки, а LSTM моделює послідовність кліків користувача. Це дозволяє передбачати, які статті будуть цікавими для конкретного користувача, і вирішує проблему холодного старту.

Метод PACE ([9]) поєднує колаборативну фільтрацію та напівавтоматичне навчання (SSL). Це дозволяє прогнозувати вподобання користувачів і одночасно враховувати контекстні точки інтересу (POI), що робить рекомендації більш персоналізованими.

Для послідовних рекомендацій у [10] використовується BERT4Rec, який враховує елементи послідовності як з минулого, так і з майбутнього. Це дає можливість передбачати дії користувача більш точно, ніж класичні однонаправлені моделі.

Модель DRR ([11]) представляє рекомендацію як послідовність рішень, використовуючи навчання з підкріпленням і схему «актор-критик». Вона здатна враховувати динаміку поведінки користувачів і довгострокові відгуки, що робить систему гнучкою і адаптивною.

Серед інших сучасних моделей варто відзначити:

- FAT-DeepFFM ([8]) для прогнозу рейтингу кліків з увагою до ознак;
- FGCNN ([7]) для автоматичного створення ознак через CNN;
- Attentive Interaction Network ([12]) для контекстно-орієнтованих рекомендацій;
- MIND ([14]) – мультиінтересна мережа, яка представляє користувача через декілька векторів, щоб враховувати різні аспекти його інтересів;
- AutoInt ([15]) – автоматичне навчання ознак високого порядку з багатоголовою увагою.

Всі ці моделі демонструють, як сучасні рекомендаційні системи працюють із різними типами даних: текстовими оглядами, послідовностями кліків, контекстом користувача, багатьма інтересами та різними ознаками для точних і персоналізованих рекомендацій. Кожна з них намагається зробити

досвід користувача більш зрозумілим, адаптивним і точним, одночасно вирішуючи проблеми розріджених даних, холодного старту та складних взаємодій.

```
Model(
 (word2vec): Embedding(53775,
 64) (user_conv): TextCNN(
 (convs): ModulcList(
 (0): Conv2d(1, 100, kernel_size=(3, 64), stride=(1, 1), padding=(2, 0))
 )
 (fc): Linear(in_features=100, out_features=10, bias=True)
 (dropout): Dropout(p=0.6, inplace=False)
 )
 (item_conv):
 TextCNN( (convs):
 ModuleList(
 (0): Conv2d(1, 100, kernel_size=(3, 64), stride=(1, 1), padding=(2, 0))
 )
 (fc): Linear(in_features=100, out_features=10, bias=True)
 (dropout): Dropout(p=0.6, inplace=False))
(final): Sequential(
 (0): Linear(in_features=20, out_features=10, bias=True)
 (1): ReLU()
 (2): Dropout(p=0.6, inplace=False)
 (3): Linear(in_features=10, out_features=1, bias=True)
 )
 (fm): TorchFM(
 (lin): Linear(in_features=20, out_features=1, bias=True)
 )
 (dropout): Dropout(p=0.6,
 inplace=False) (relu): ReLU()
)
```

Рис.2.3 – Модель DeepCoNN формування рекомендацій, що враховує текстові огляди [16]

Опис моделі DeepCoNN для формування рекомендацій

Модель DeepCoNN (Deep Cooperative Neural Networks) призначена для побудови рекомендацій з урахуванням текстових оглядів користувачів і товарів. Вона поєднує нейронні мережі для обробки тексту та факторизаційні методи.

```
Model(
```

```
(word2vec): Embedding(53835, 64)
(user_embedding): Embedding(87273, 10)
(item_embedding): Embedding(13211, 10)
(user_conv): TextCNN(
(convs): ModuleList(
(0): Conv2d(1, 100, kernel_size=(3, 64), stride=(1, 1), padding=(2, 0)))
(fc): Linear(in_features=100, out_features=10, bias=True)
(dropout): Dropout(p=0.6, inplace=False))
(item_conv): TextCNN(
(convs): ModuleList(
(0): Conv2d(1, 100, kernel_size=(3, 64), stride=(1, 1), padding=(2, 0)))
(fc): Linear(in_features=100, out_features=10, bias=True)
(dropout): Dropout(p=0.6, inplace=False))

(attention_scorer_user): Sequential(
(0): Linear(in_features=20, out_features=10, bias=True)
(1): ReLU()
(2): Dropout(p=0.6, inplace=False)
(3): Linear(in_features=10, out_features=1,
bias=True)) (attention_scorer_item): Sequential(
(0): Linear(in_features=20, out_features=10, bias=True)
(1): ReLU()
(2): Dropout(p=0.6, inplace=False)
(3): Linear(in_features=10, out_features=1, bias=True))

(final): Sequential(
(0): Dropout(p=0.6, inplace=False)
(1):
Linear(in_features=10,
out_features=10,
bias=True)
(2): ReLU()
(3):
Linear(in_features=10,
out_features=1,
bias=True) )
(dropout): Dropout(p=0.6, inplace=False)
(sigmoid): Sigmoid()
(relu): ReLU())
```

Рис.2.4 – Модель NARRE формування рекомендацій, що враховує текстові огляди [17]

Модель NARRE призначена для прогнозування оцінок користувачів із урахуванням текстових оглядів, використовуючи механізм уваги (attention) для виділення найбільш інформативних відгуків.

DeepCoNN [16] – це глибока нейронна модель, яка одночасно вивчає поведінку користувачів і характеристики предметів через текстові відгуки. Її структура складається з двох паралельних мереж: одна аналізує відгуки користувачів, щоб зрозуміти їхні вподобання та поведінку, а інша – вивчає відгуки про товари, щоб визначити їхні властивості. На фінальному шарі обидві мережі об'єднуються, створюючи спільне латентне представлення користувачів і товарів. Таким чином, модель поєднує ідеї класичної матричної факторизації та сучасних нейронних мереж, забезпечуючи більш точне прогнозування рейтингів.

NARRE [17] пішла ще далі: вона оцінює значущість кожного окремого відгуку перед тим, як використати його у системі рекомендацій. Ідея дуже проста – не всі відгуки однаково корисні, деякі можуть навіть «забруднювати» модель. Тому NARRE застосовує механізм уваги, який присвоює кожному відгуку певну вагу, дозволяючи моделі більше покладатися на релевантні огляди, а менш інформативні відкинути.

HFT [18] використовує текстові відгуки як додатковий регуляризатор. Це означає, що модель не просто враховує оцінки користувачів, а порівнює їх із фактичним змістом відгуків, щоб зрозуміти, наскільки оцінка відповідає реальним характеристикам товару. Завдяки цьому система може точніше відображати справжні вподобання користувачів.

TransNets [19] – розширена версія DeepCoNN, яка вводить додатковий латентний рівень для кожної пари «користувач-товар». Під час навчання цей рівень наближається до прихованого представлення відгуку конкретного користувача про конкретний товар, що дозволяє моделі краще передбачати оцінки навіть для нових або маловідомих продуктів.

MPCN [20] працює за принципом багаторівневої ієрархії: модель припускає, що не всі відгуки однаково важливі, а ключові коментарі мають

найбільший вплив на прогнозування рейтингу. Використовується метод «рецензія за рецензією», який виділяє найбільш значущі відгуки та порівнює їх слово за словом. Це дозволяє системі глибше аналізувати текст і точніше визначати, які аспекти продукту найбільше впливають на вибір користувача.

NRPA [21] застосовує персоналізовану увагу. Модель враховує, що одні й ті ж слова можуть мати різне значення для різних користувачів або для різних товарів. Наприклад, слово «комфортний» в описі взуття важить по-різному для студента і для спортсмена. NRPA автоматично підлаштовує увагу під кожного користувача і кожен товар, що підвищує точність рекомендацій.

DAML [22] використовує локальну і взаємну увагу у згортковій нейронній мережі, щоб одночасно аналізувати характеристики відгуків і взаємодії між користувачами та товарами. Потім рейтинг та текстові дані об'єднуються в єдину нейронну мережу. Модель здатна обробляти складні нелінійні взаємодії ознак високого порядку за допомогою нейронної факторизації, що дозволяє отримати точні прогнози фінального рейтингу. Крім того, цей підхід робить систему більш «інтерпретованою» – можна зрозуміти, чому саме той чи інший відгук впливає на рекомендацію.

Сьогодні ці моделі використовуються для вирішення широкого спектру завдань у системах рекомендацій: від прогнозування рейтингів товарів у маркетплейсах до персоналізованих рекомендацій новин і мультимедіа. Вони дозволяють враховувати не лише оцінки користувачів, а й зміст відгуків, послідовність дій, контекст і навіть індивідуальні інтереси. Завдяки цьому рекомендаційні системи стають більш гнучкими, точними і здатними адаптуватися під конкретного користувача, а не просто відображати популярність товарів чи послуг.

Використання цих моделей особливо ефективне у великих e-commerce платформах, де мільйони користувачів залишають сотні тисяч відгуків щодня. Можливість автоматично обробляти текстові дані, виділяти найбільш релевантні частини відгуків та об'єднувати їх із структурованими даними

робить систему рекомендацій реально потужним інструментом для підвищення продажів, залучення користувачів і покращення їхнього досвіду.

2.2. Методи навчання глибоких нейронних мереж

Метод стохастичного градієнтного спуску (SGD)

Градієнтні методи належать до широкого класу алгоритмів оптимізації, які застосовуються не лише в машинному навчанні, але й у багатьох інших сферах. У контексті нейронних мереж їх використовують для налаштування синаптичних ваг www у шарах мережі, щоб мінімізувати функцію втрат і поліпшити точність прогнозів.

Нехай у нас є навчальна вибірка:

$$X_l = \{(x_i, y_i)\}_{i=1}^{l}, \quad y = y^*(x)$$

Знайдемо алгоритм a(x,w) що наближає залежність y*. Для лінійного класифікатора цей алгоритм має вигляд:

$$a(x, w) - \varphi\Big(\sum_j w_j x_j - w_0\Big)$$

Алгоритм SGD працює наступним чином:

- Вхідні дані:
- x=(*x*1,*x*2,…,*x*n) -вектор вхідних ознак для нейрона.
- *w*=(*w*1,*w*2,…,*w*n) -вектор ваг нейрона, який показує важливість кожної ознаки *xjx_jxj*
- Вихід: оптимальні ваги *www*.
- Ініціалізація: випадкові значення ваг *wjw_jwj* та обчислення початкового функціоналу якості:

$$Q = \sum_{i=1}^{l} L(a(x_i, w), y_i)$$

- x *i* -Вектор ознак *i* -го навчального прикладу.
- y *i* -Правильна мітка (label) для *i* -го прикладу.
- a(x *i*,*w*) -Вихід моделі (нейрона або мережі) на прикладі *xix_ixi* з параметрами *w*.

Наприклад, обчислити результат алгоритму a(xi, w) та помилку:

$$\xi_i = L(a(x_i, w), y_i)$$

Прогноз a(xi,w)a(x_i, w)a(xi,w) та помилку:

Оновити ваги:

$$w := w - \eta\, L'\big(a(x, w), y\big)\, \varphi'\big(\langle w, x\rangle\big)\, x$$

Оновити функціонал якості:

$$Q := (1 - \lambda)Q + \lambda\, \xi_i$$

Поки значення Q не стабілізується та/або ваги w не припинять змінюватись.

SGD простий у реалізації і добре працює на великих вибірках, проте іноді вимагає підбору темпу навчання і може коливатися під час сходження.

Adam (Adaptive Moments)

Adam – це більш сучасний метод оптимізації, який поєднує ідеї RMSProp та моменту (momentum). Його головна перевага – адаптивний крок навчання для кожного параметра, що робить алгоритм стійким до вибору гіперпараметрів.

Алгоритм Adam:

1. Дано:
   - крок *ϵ\epsilonϵ* (звичайно 0.001),
   - коефіцієнти експоненційного затухання *ρ1,ρ2∈[0,1)\rho_1, \rho_2 \in [0,1)ρ1,ρ2∈[0,1)* (зазвичай 0.9 і 0.999),
   - мала константа δ\deltaδ для чисельної стабільності,

- початкові параметри θ\thetaθ.

2. Ініціалізація: $s=0, r=0, t=0$ $s = 0,\ r = 0,\ t = 0$ $s=0,r=0,t=0$ (змінні для моментів та лічильник кроків).
3. Поки критерій зупинки не виконано:

   Вибрати міні-пакет прикладів

$$\{(x^{(i_1)}, y^{(i_1)}), (x^{(i_2)}, y^{(i_2)}), \ldots, (x^{(i_m)}, y^{(i_m)})\}$$

Ці m прикладів утворюють **mini-batch**, на якому обчислюється середній градієнт.

Обчислити градієнт:

$$g \leftarrow \frac{1}{m} \nabla_\theta \sum_i L(f(x^{(i)}; \theta), y^{(i)}), \quad t \leftarrow t + 1$$

Оновити моменти:

$$s \leftarrow \rho_1 s + (1 - \rho_1) g, \quad r \leftarrow \rho_2 r + (1 - \rho_2) g \odot g$$

Скоригувати зміщення:

$$\hat{s} \leftarrow \frac{s}{1 - p_t}$$

Оновити параметри:

$$\theta \leftarrow \theta - \eta \frac{\hat{s}}{\sqrt{\hat{r}} + \delta}$$

У Adam оновлення ваг враховує градієнт, момент та регуляризацію, що дозволяє швидко і стабільно досягати мінімуму функції втрат навіть на великих і розріджених даних. Цей метод особливо популярний при навчанні глибоких нейронних мереж і сучасних рекомендаційних систем.

2.3. Критерії якості задачі надання рекомендацій

Метрики оцінки якості рекомендацій

У сучасних рекомендаційних системах важливо не просто давати користувачу випадкові пропозиції, а реально пропонувати те, що йому цікаво. Для цього застосовують різні метрики, які допомагають оцінити ефективність системи, її точність та практичну користь.

Основні категорії метрик:

Точність (Precision) – показує, яка частка запропонованих системою товарів або послуг була релевантною для користувача. Наприклад, якщо система запропонувала 10 книг, а користувач зацікавився 7 із них, точність дорівнює 0,7.

Охоплення (Coverage) – показує, скільки різних товарів система здатна рекомендувати відносно всього асортименту. Чим ширше охоплення, тим більше шансів, що користувач знайде щось нове й цікаве.

Метрики точності

Метрики точності поділяються на статистичні та специфічні для рекомендацій.

MAE (Mean Absolute Error, середня абсолютна похибка) Показує середнє відхилення прогнозу системи від реальної оцінки користувача. Чим менше значення MAE, тим точніше модель передбачає вподобання.

RMSE (Root Mean Squared Error, середньоквадратична помилка) Більш сувора метрика, оскільки великі помилки “касають сильніше”. Наприклад, якщо система зовсім не вгадала улюблену книгу користувача, RMSE за це покарає сильніше, ніж MAE.

Спеціалізовані метрики для рекомендаці

Precision@k

Відображає, яка частка з топ-k рекомендованих елементів була релевантною. Наприклад, якщо користувач переглянув 10 товарів із топ-20 рекомендацій, Precision@20 = 0,5.

AP@k (Average Precision) Показує не лише факт попадання релевантних елементів у список, але й їхню позицію. Елементи, що опинилися на верхніх місцях, мають більший вплив.

MAP@k (Mean Average Precision) Це середнє значення AP@k по всім користувачам тестової вибірки. Допомагає оцінити, наскільки система стабільно працює для різних користувачів.

nDCG (Normalized Discounted Cumulative Gain) Оцінює, наскільки порядок рекомендацій близький до ідеального. Якщо релевантні товари опинилися внизу списку, nDCG покаже це, навіть якщо вони присутні.

F1@k Гармонійне середнє Precision@k та Recall@k. Особливо корисно, коли важливо не лише показати правильні рекомендації, а й не пропустити важливі для користувача елементи.

MCC (Matthews Correlation Coefficient) Оцінює кореляцію між передбаченими та реальними бінарними результатами (купив/не купив, лайк/дизлайк). Вважається дуже надійною метрикою для збалансованих та розріджених даних.

Додаткові якісні показники

Окрім класичних метрик точності, важливо дивитися на “людський аспект” системи:

– Різноманітність – система пропонує користувачу різні категорії товарів, а не тільки популярні.

– Охоплення всього асортименту – включно з новими або рідко обираними товарами.

– Покриття користувачів – частка користувачів, для яких система може сформувати рекомендації.

– Новизна – наскільки пропоновані товари або контент нові для користувача.

– Неочікуваність – здатність здивувати користувача, запропонувавши щось несподіване.

– Надійність – наскільки користувач довіряє рекомендаціям системи.

– Відтік (Drift) – як швидко змінюються рекомендації після нових оцінок або дій користувача.

Для комерційних проєктів важливі показники реальної користі системи:

– кількість кліків на рекомендовані товари,

– конверсії у покупки,

– вплив на розподіл продажів і популярність товарів,

– утримання користувачів та їх лояльність.

Таким чином, оцінка рекомендаційної системи – це не лише цифри, а й розуміння того, наскільки вона реально корисна, цікава та зручна для користувача. Комбінація статистичних, специфічних і бізнес-метрик дозволяє знайти баланс між точністю прогнозів та практичною цінністю рекомендацій.

2.4. Пропонована модифікація нейромережевої моделі формування рекомендацій та її обґрунтування.

Нехай у нас є: множина користувачів u∈Uu \in Uu∈U та множина об’єктів i∈Ii \in Ii∈I, множина подій (rui,u,i)∈D(r_{ui}, u, i) \in D(rui,u,i)∈D, які описують взаємодії користувачів із товарами чи сервісами, тексти користувацьких відгуків для відповідних об’єктів.

Кожна подія містить рейтингову оцінку ruir_{ui}rui та інші характеристики, що можуть впливати на прогноз. Основна мета – передбачити рейтинг, який користувач поставить товару, і на основі цього формувати персоналізовані рекомендації. Це може бути як user-based підхід, коли рекомендації формуються для конкретного користувача, так і item-based, коли фокус робиться на конкретному товарі.

Для розв’язання цієї задачі обрана модель DeepCoNN. Це глибока нейромережева архітектура, яка окремо моделює поведінку користувачів і характеристики продуктів, використовуючи паралельні згорткові нейронні мережі (CNN) для обробки текстів відгуків. Після обробки текстів

застосовується Factorization Machine, яка прогнозує рейтинги на основі латентних представлень користувачів та об’єктів.

У запропонованій модифікації моделі враховані такі вдосконалення:

– Базова модель – DeepCoNN як основа для роздільного моделювання користувачів та продуктів.

– Textual Preprocessing – попередня обробка текстів, яка включає нормалізацію, токенізацію та виділення ключових слів, щоб мережа працювала з чистим і зрозумілим контентом

– Attention-шар – на етапі поєднання латентних представлень користувачів і товарів додається механізм уваги, який дозволяє мережі зважувати більш значущі відгуки і приділяти більше уваги релевантним фразам.

– Глобальні та локальні зміщення (bias) – для користувачів і продуктів застосовуються власні зміщення, що замінюють класичний підхід

– Factorization Machine. Це допомагає моделі краще враховувати індивідуальні вподобання та популярність товарів.

Такий підхід дозволяє не лише прогнозувати рейтинги точніше, а й робить систему більш інтерпретованою: легко відстежувати, які відгуки або характеристики продукту найбільше вплинули на рекомендацію.

```
Model(
  (word2vec): Embedding(42455, 64)
  (user_conv): TextCNN(
    (convs): ModuleList(
      (0): Conv2d(1, 100, kernel_size=(3, 64), stride=(1, 1), padding=(2, 0))
    )
    (fc): Linear(in_features=100, out_features=10,
    bias=True) (dropout): Dropout(p=0.6, inplace=False)
  )
  (item_conv): TextCNN(
    (convs): ModuleList(
      (0): Conv2d(1, 100, kernel_size=(3, 64),
stride=(1, 1), padding=(2, 0))
    )
```

```
  (fc): Linear(in_features=100, out_features=10,
  bias=True) (dropout): Dropout(p=0.6, inplace=False)
)
(attention): Sequential(
  (0): Linear(in_features=128, out_features=10, bias=True)
  (1): ReLU()
  (2): Dropout(p=0.6, inplace=False)
  (3): Linear(in_features=10, out_features=1, bias=True)
)
(final): Sequential(
  (0): Dropout(p=0.6, inplace=False)
  (1): Linear(in_features=20, out_features=10, bias=True)
  (2): ReLU()
  (3): Linear(in_features=10, out_features=1, bias=True)
)
(dropout): Dropout(p=0.6, inplace=False)
(relu): ReLU()
```

Рис.2.5 – Пропонована модифікована нейромережева модель формування рекомендацій

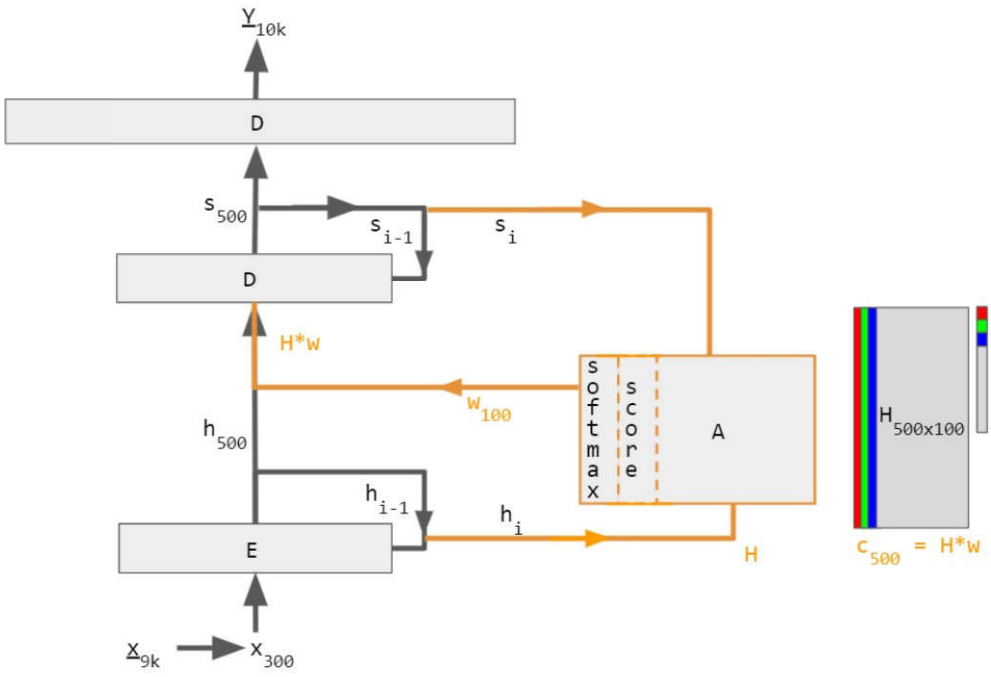


Рис.2.6 – шар Attention на етапі зв’язування текстів оглядів користувачів і продуктів [25]

Розглянемо технологію Textual Preprocessing для аналізу вхідних текстів (рис.2.7).

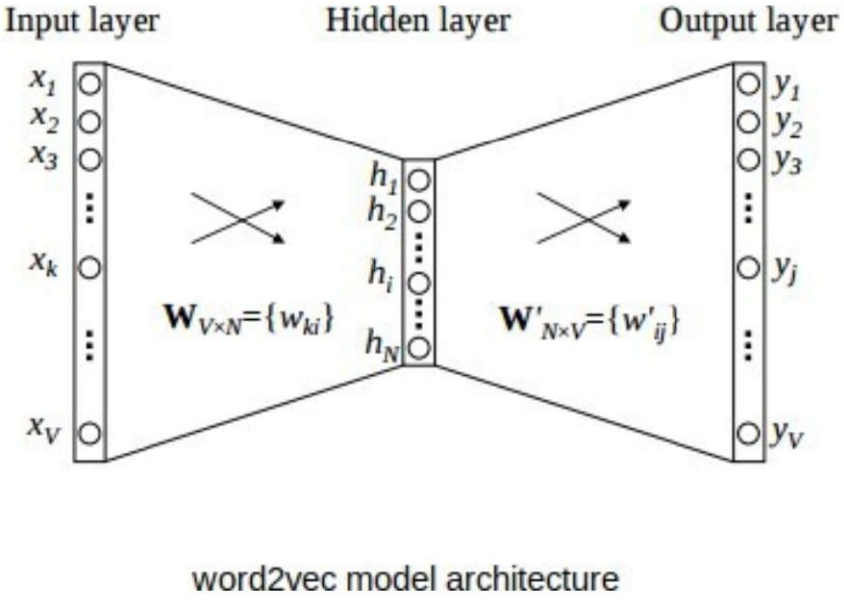


Рис.2.7 – Архітектура моделі Word2vec [26]

Архітектура інтелектуальної системи рекомендацій для інтернет-магазинів.

У рамках цієї роботи запропоновано модель, яка дозволяє будувати персоналізовані рекомендації для користувачів інтернет-магазинів на основі їхньої поведінки та текстових відгуків. Структура моделі показана на рис.2.7 складається з кількох ключових елементів.

1. Word2vec Embedding Першим етапом всі слова з відгуків та описів товарів перетворюються у вектори розміром 64. Словник налічує 42 455 слів, що дозволяє зберігати семантичну і контекстну інформацію. Word2vec допомагає системі розуміти синоніми, знаходити слова, які часто зустрічаються разом, і покращує точність прогнозів навіть для рідкісних товарів.
2. TextCNN для користувачів і товарів (user_conv та item_conv) Для аналізу текстів відгуків користувачів і описів товарів застосовуються дві паралельні згорткові нейронні мережі. Кожна мережа включає:

- згортковий шар Conv2d (1 канал, 100 фільтрів, ядро 3×64, паддінг 2×0);
- лінійний шар для зменшення розмірності (100→10);
- Dropout з ймовірністю 0.6 для запобігання перенавчанню.

Таке рішення дозволяє виділяти важливі ознаки у тексті, що реально відображають інтереси користувачів.

3. Attention-шар

Після виділення ознак користувачів і товарів вони об'єднуються за допомогою механізму уваги. Його структура:

- лінійний шар (128→10);
- ReLU;
- Dropout(p=0.6);
- лінійний шар (10→1).

Завдяки Attention модель оцінює, які відгуки та характеристики товарів найбільше впливають на оцінку продукту, що дозволяє системі давати більш точні та релевантні рекомендації.

4. Фінальний шар прогнозування рейтингу На останньому етапі всі ознаки об'єднуються, застосовується Dropout і нелінійні перетворення, щоб передбачити рейтинг товару для конкретного користувача. Це дає можливість інтернет-магазину пропонувати продукти, які максимально відповідають вподобанням клієнта.

Textual Preprocessing та Word2vec у рекомендаційній системі

Для роботи з текстами застосовується Textual Preprocessing, який включає:

- токенізацію;
- приведення слів до базової форми;
- видалення стоп-слів;
- перетворення слів у вектори.

Word2vec навчає мережу представляти кожне слово як числовий вектор, який зберігає його значення та контекст. Це дає системі змогу

- знаходити синоніми і контекстно схожі слова;
- підвищувати точність прогнозів для нових або малопопулярних товарів;
- ефективно працювати з великою кількістю текстових відгуків.

Таким чином, запропонована модель дозволяє інтелектуально передбачати вподобання користувачів, формувати точні рекомендації і враховувати поведінку клієнтів онлайн-магазину, що робить систему максимально ефективною для реального застосування.

У цьому розділі ми детально розглянули математичні основи сучасних методів рекомендацій товарів та алгоритми, які використовуються для прогнозування оцінок користувачів. Особливу увагу приділено аналізу існуючих моделей та способів оцінки їх ефективності. Для цього були обрані як статистичні метрики (MAE, RMSE), так і специфічні показники для рекомендаційних систем (Precision@k, MAP@k, nDCG, F1@k та інші), що дозволяє всебічно оцінювати якість рекомендацій і адаптивність системи до різних користувачів та товарів.

На основі проведеного аналізу була запропонована модифікація глибокої нейромережевої моделі DeepCoNN, яка адаптована саме для інтелектуальної системи рекомендацій в інтернет-магазинах на основі поведінки користувачів. Ключові особливості цієї модифікації:

Textual Preprocessing – перед обробкою відгуків користувачів і характеристик товарів проводиться токенізація, нормалізація та перетворення слів у вектори, що дозволяє нейромережі точніше «розуміти» смислове навантаження текстів і виділяти ключові характеристики.

Attention-шар, який зв’язує ознаки користувачів і продуктів. Завдяки ньому модель визначає, які саме відгуки користувачів і властивості товарів є найважливішими для прогнозу оцінки. Це дозволяє системі робити більш персоналізовані рекомендації, враховуючи індивідуальні переваги клієнтів.

Глобальні, користувацькі та продуктові зміщення (bias) замість класичної Factorization Machine. Такий підхід допомагає точніше враховувати як загальні тренди (наприклад, популярність товару), так і особливості конкретного користувача та конкретного продукту.

Додатково, завдяки Attention-механізму система здатна виділяти найбільш інформативні відгуки та оцінювати їхню вагу для кожного користувача окремо. Це дозволяє не просто передбачати рейтинг продукту, а формувати дійсно релевантні та персоналізовані рекомендації, підвищуючи задоволення клієнтів і ефективність продажів у інтернет-магазині.

Таким чином, запропонована модифікація DeepCoNN є основою для розробки інтелектуальної системи рекомендацій, яка не лише прогнозує рейтинги товарів, але й адаптується під конкретну поведінку користувачів, що робить її практичною і ефективною для комерційного застосування.

РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МОДЕЛЕЙ ГЛИБОКИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ ФОРМУВАННЯ РЕКОМЕНДАЦІЙ

3.1. Дослідження ефективності запропонованого методу

Щоб підготувати дані до навчання, усі підмножини були розділені за схемою 80-10-10: 80% — навчальний набір, 10% — валідаційний, 10% — тестовий. Такий розподіл дозволяє оцінювати роботу моделей на незалежних даних і запобігає перенавчанню.

Особливу увагу було приділено підготовці текстів. Для моделей на кшталт NARRE стоп-слова не видаляються, а словник обмежується 50 тисячами найуживаніших слів. Тексти перетворюються на вектори за допомогою Word2vec — популярного методу NLP, який навчає модель представляти кожне слово числовим вектором із врахуванням його контексту. Для цього використовувалася бібліотека Gensim, що дозволяє швидко й ефективно навчити вбудовування слів на великому обсязі текстових даних.

У моделі DeepCoNN довжина кожного документа (для користувача або продукту) стандартизована до 1000 токенів: короткі огляди доповнюються спеціальними маркерами, а довгі — обрізаються, щоб уникнути перевантаження мережі. Інші моделі отримують аналогічно відрегульовані довжини відгуків, що гарантує уніфікований формат даних для всіх нейронних мереж. Такий підхід дозволяє моделі коректно порівнювати поведінку користувачів та характеристики продуктів, не звертаючи увагу на природні відмінності у довжині текстів.

Крім того, попередня обробка текстів включає токенізацію, нормалізацію та видалення непотрібних символів, що дозволяє мережі фокусуватися на суттєвій інформації. Після цього слова переводяться у вектори, які стають вхідними даними для згорткових шарів нейромережі.

Такий підхід дозволяє точно аналізувати як окремі огляди, так і загальні патерни поведінки користувачів.

Завдяки такій підготовці даних можна ефективно навчати моделі, прогнозувати оцінки користувачів і формувати релевантні рекомендації для кожного товару та користувача, що безпосередньо відповідає темі кваліфікаційної роботи: **«Розробка інтелектуальної системи рекомендацій для інтернет-магазинів на основі поведінки користувачів»**.

Таблиця 3.1

Інформація про використані датасети

	Метрики		
Датасети	MSE	NDCG	HR@1
Android Apps	1.3966	1.2973	37.81
Video Games	1.1765	1.2267	43.64
Cell Phones	1.3489	1.2649	45.45
Office products	0.7692	1.2495	33.33

Як видно з Таблиці 3.1, навіть на початкових етапах навчання DeepAtnCoNN демонструє невелике, але помітне покращення точності прогнозів у порівнянні з іншими моделями, за винятком NeuMF, яка все ще показує високі результати. Це вже свідчить про те, що додавання Attention-шару дозволяє моделі краще виділяти релевантні частини текстів оглядів і точніше оцінювати вподобання користувачів.

Для більш детального аналізу був проведений експеримент щодо впливу кількості епох навчання на якість рекомендацій. Кількість епох поступово збільшували до 10, щоб відстежити динаміку зростання точності. Результати цього експерименту представлені в Таблиці 3.2. Як показують дані, точність системи стабільно зростає з кожною новою епохою, що свідчить про ефективність запропонованої модифікації та її здатність до навчання на великих масивах даних.

Таблиця 3.2

Дослідження впливу кількості епох на точність рекомендацій

Epoch	Android Apps		Video Games		Cell Phones		Office products	
	MSE	NDCG	MSE	NDCG	MSE	NDCG	MSE	NDCG
1	1.4778	1.331	1.2688	1.2491	1.4218	1.3092	0.8224	1.182
2	1.3997	1.3045	1.1816	1.2265	1.3745	1.2899	0.7845	1.2049
3	1.3717	1.2853	1.137	1.2089	1.3507	1.2728	0.7611	1.2085
4	1.3635	1.2702	1.1146	1.1954	1.3407	1.2593	0.7465	1.2002
5	1.3638	1.2585	1.1041	1.185	1.3395	1.2461	0.7379	1.198
6	1.3678	1.2492	1.1008	1.1691	1.3437	1.2352	0.7332	1.1949
7	1.3732	1.2419	1.1012	1.1609	1.3514	1.2216	0.731	1.1881
8	1.379	1.2363	1.1039	1.1541	1.3609	1.2179	0.7307	1.1855
9	1.3847	1.232	1.1077	1.1476	1.3715	1.2058	0.7315	1.1823
10	1.39	1.2286	1.1123	1.1415	1.3823	1.2092	0.7331	1.1788

З таблиці видно, що максимальна точність MSE досягається приблизно на 5–6 епосі, після чого спостерігається поступове зниження продуктивності. Детальніше результати будуть розглянуті в порівнянні з іншими методами у розділі 3.4.

Таблиця 3.3

Фінальне тестування найкращих моделей

	Final Test			
	Android Apps	Video Games	Cell Phones	Office products
MSE	1.3708	1.1026	1.3166	0.7042
NDCG	1.2574	1.2053	1.2474	1.1752
HR@1	37.31	40.91	45.45	33.33

Графічно порівняльну залежність від епох для моделі по датасетам можна побачити нижче:

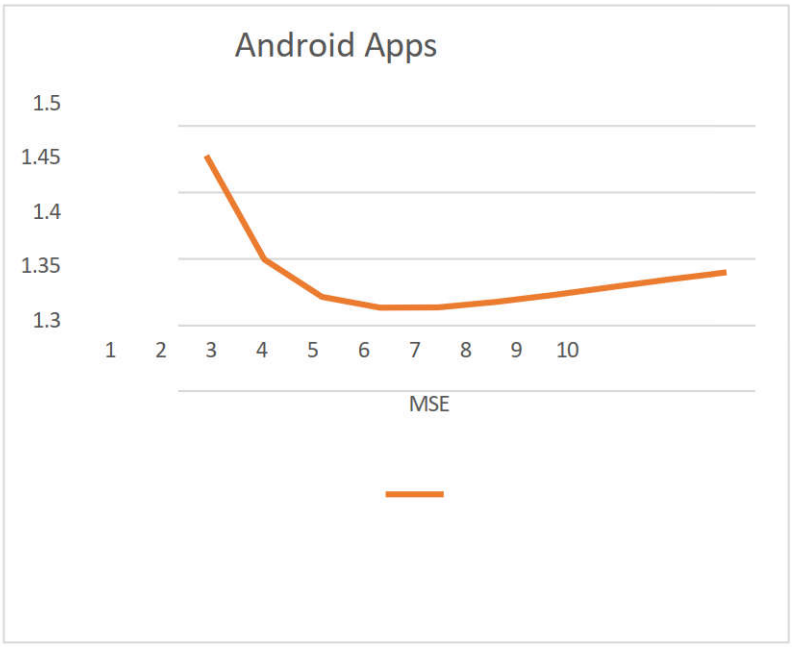


Рис.3.1 – Залежність MSE від кількості епох для датасету Android Apps

Рис. 3.1 – Графік показує зміну середньоквадратичної помилки (MSE) залежно від кількості епох навчання на прикладі датасету Android Apps. Він ілюструє, як точність моделі покращується з ростом числа епох і стабілізується після певного етапу тренування.

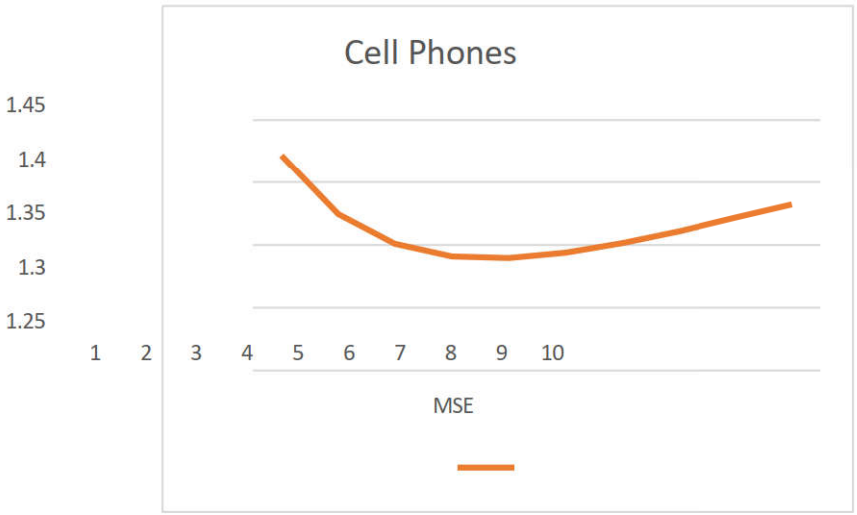


Рис.3.2 – Залежність MSE від кількості епох для датасету Cell Phones

Рис. 3.2 – Графік демонструє зміну середньоквадратичної помилки (MSE) залежно від кількості епох навчання для датасету Cell Phones. Він показує, як точність моделі покращується під час навчання та досягає стабільного значення після певної кількості епох.

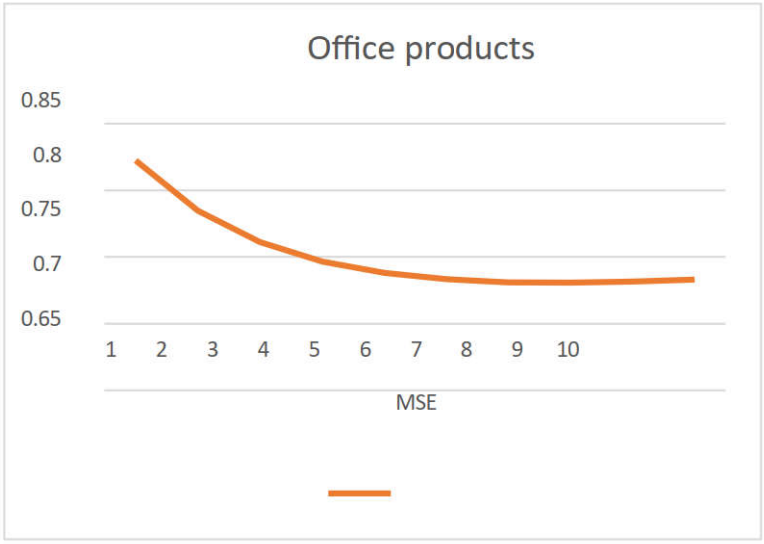


Рис.3.3– Залежність MSE від кількості епох для датасету Office Products

Рис. 3.3 – Графік ілюструє залежність середньоквадратичної помилки (MSE) від кількості епох навчання для датасету Office Products. Він показує, як точність моделі поступово покращується під час навчання і стабілізується після досягнення певного числа епох.

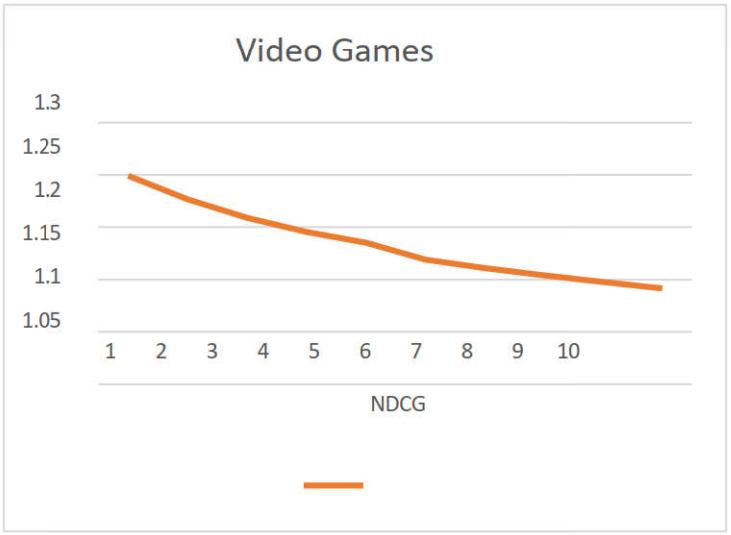


Рис.3.4 – Залежність NDCG від кількості епох для датасету Video Games

Рис. 3.4 – Графік демонструє зміну метрики NDCG (Normalized Discounted Cumulative Gain) залежно від кількості епох навчання для датасету Video Games. Він відображає, як якість ранжування рекомендацій покращується в процесі навчання моделі та досягає стабільного рівня після певної кількості епох.

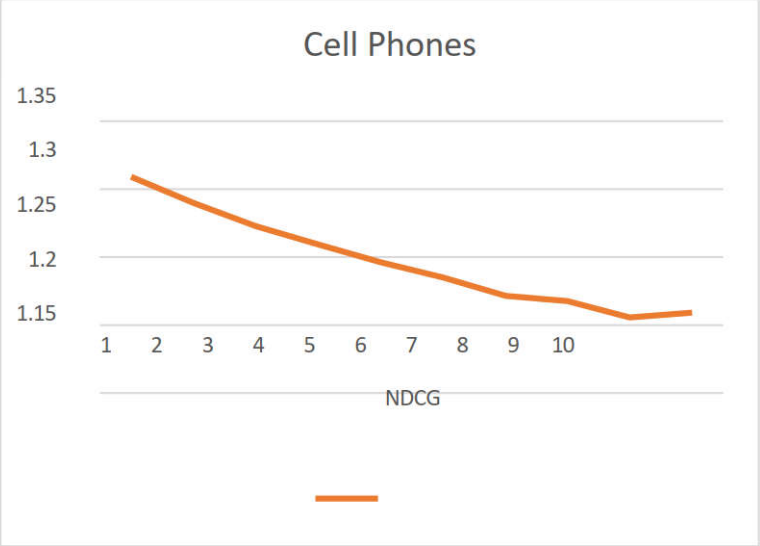


Рис.3.5 – Залежність NDCG від кількості епох для датасету Cell Phones

Рис. 3.5 – Графік показує зміну метрики NDCG (Normalized Discounted Cumulative Gain) залежно від кількості епох навчання для датасету Cell Phones. Він ілюструє покращення якості ранжування рекомендацій у міру навчання моделі та досягнення стабільного рівня після певної кількості епох.

Аналогічно попереднім моделям, було проведено дослідження впливу Dropout на точність рекомендацій при кількості епох, рівній 5. Результати представлені у Таблиці 3.4

Таблиця 3.4

Дослідження впливу Dropout на точність рекомендацій.

Dropout	**Android Apps**			**Video Games**			**Cell Phones**			**Office products**		
Metrics	**MSE**	**NDCG**	**HR@1**	**MSE**	**NDCG**	**HR@1**	**MSE**	**NDCG**	**HR@1**	**MSE**	**NDCG**	**HR@1**
0.2	**1.3638**	**1.2664**	**35.32**	**1.1058**	**1.2852**	**42.73**	**1.3169**	**1.2327**	**45.45**	**0.7041**	**1.1955**	**33.33**
0.4	**1.3671**	**1.2799**	**36.82**	**1.0995**	**1.2509**	**41.82**	**1.3168**	**1.2325**	**45.45**	**0.7153**	**1.2009**	**33.33**
0.6	**1.3677**	**1.2484**	**37.31**	**1.1048**	**1.2109**	**42.73**	**1.3169**	**1.2311**	**45.45**	**0.7147**	**1.1914**	**33.33**
0.8	**1.3708**	**1.257**	**37.31**	**1.1048**	**1.2143**	**41.82**	**1.3169**	**1.2342**	**45.45**	**0.7152**	**1.1964**	**33.33**

Як можна помітити, поведінка моделі дуже нагадує результати, спостережувані у DeepCoNN. Слід звернути увагу, що хоча модель демонструє дещо кращу точність на середніх значеннях Dropout, ця різниця не є критичною — на загальному фоні показники залишаються досить близькими.

Графічно залежність точності від рівня Dropout для різних датасетів представлена на рисунку нижче, що дозволяє наочно оцінити, як зміна цього гіперпараметра впливає на роботу моделі.

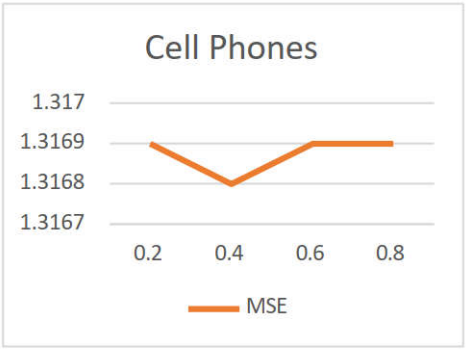


Рис.3.6 – Залежність MSE від Dropout для датасету Cell Phones

Рис. 3.6 – Графік ілюструє залежність середньоквадратичної помилки (MSE) від значення параметра Dropout для датасету Cell Phones. Він показує вплив регуляризації на точність моделі, де оптимальні значення Dropout дозволяють зменшити помилку та покращити загальну продуктивність системи рекомендацій.

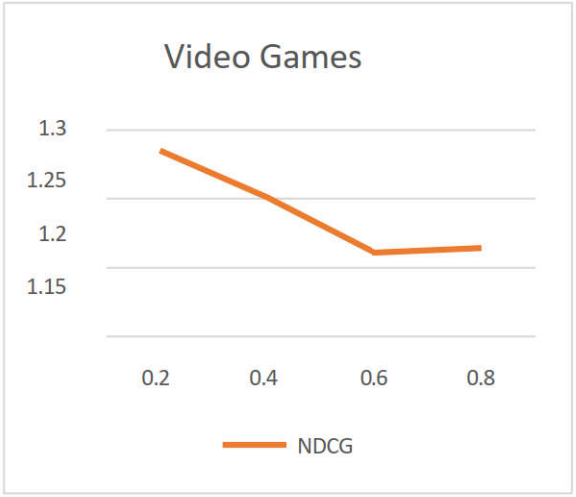


Рис.3.7– Залежність NDCG від Dropout для датасету Video Games

Рис. 3.7 – Графік демонструє залежність метрики NDCG (Normalized Discounted Cumulative Gain) від значення параметра Dropout для датасету Video Games. Він показує, як різні рівні регуляризації впливають на якість ранжування рекомендацій, дозволяючи визначити оптимальні налаштування для моделі.

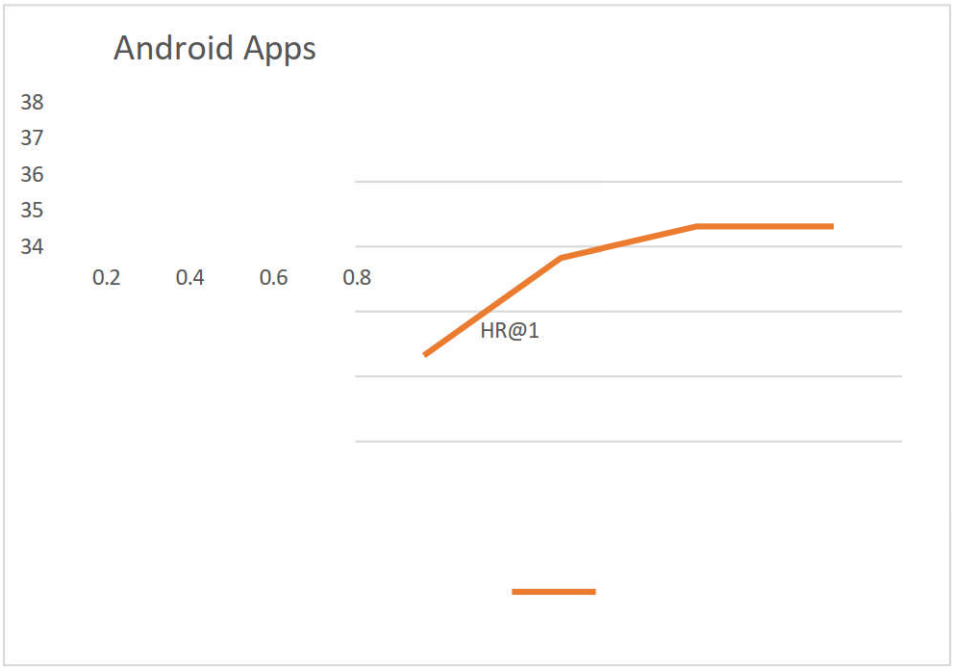


Рис.3.8 – Залежність HR@1 від Dropout для датасету Android Apps

Рис. 3.8 – Графік показує залежність метрики HR@1 (Hit Rate на першій позиції) від значення параметра Dropout для датасету Android Apps. Він ілюструє вплив регуляризації на точність рекомендацій, допомагаючи визначити оптимальні налаштування моделі для досягнення кращих результатів.

3.2. Аналіз отриманих результатів

Запропонована модель демонструє відчутне покращення точності порівняно з трьома попередніми моделями на датасетах Android Apps, Cell Phones та Video Games. Це свідчить про ефективність використаного підходу, зокрема інтеграції Attention-шару та обробки текстових відгуків користувачів.

Щодо датасету Office Products, модель також показує кращі результати, ніж DeepCoNN та NARRE, хоча показники майже зрівнюються з NeuMF. Це пояснюється тим, що для цього набору даних тексти оглядів менш впливають на точність прогнозу, а класичні факторизаційні підходи (як у NeuMF) вже досить добре справляються із завданням.

Аналіз результатів на різних датасетах дозволяє зробити висновок, що запропонована модифікація DeepAtnCoNN добре масштабується та стабільно підвищує точність там, де наявні значущі текстові дані. На Рисунках 49–56 представлено детальне порівняння показників точності навчання всіх чотирьох моделей, що дозволяє наочно оцінити переваги нової моделі над класичними підходами та зрозуміти її поведінку на різних типах даних.

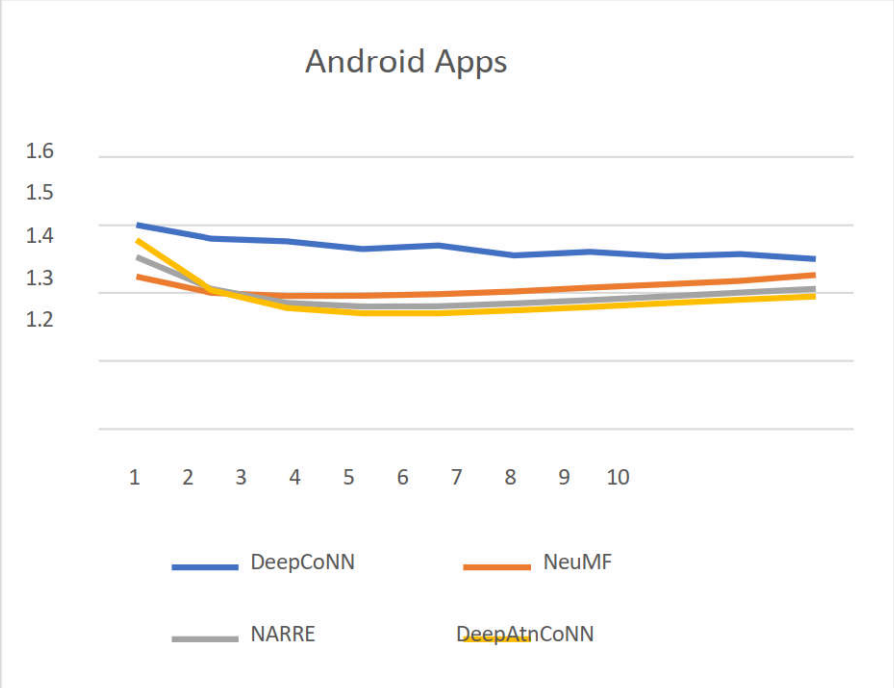


Рис.3.9 – Залежність MSE від кількості епох для датасету Android Apps

Рис. 3.9 – Графік демонструє зміну середньоквадратичної помилки (MSE) залежно від кількості епох навчання для датасету Android Apps. Він показує, як точність моделі покращується під час навчання і стабілізується після досягнення певної кількості епох.

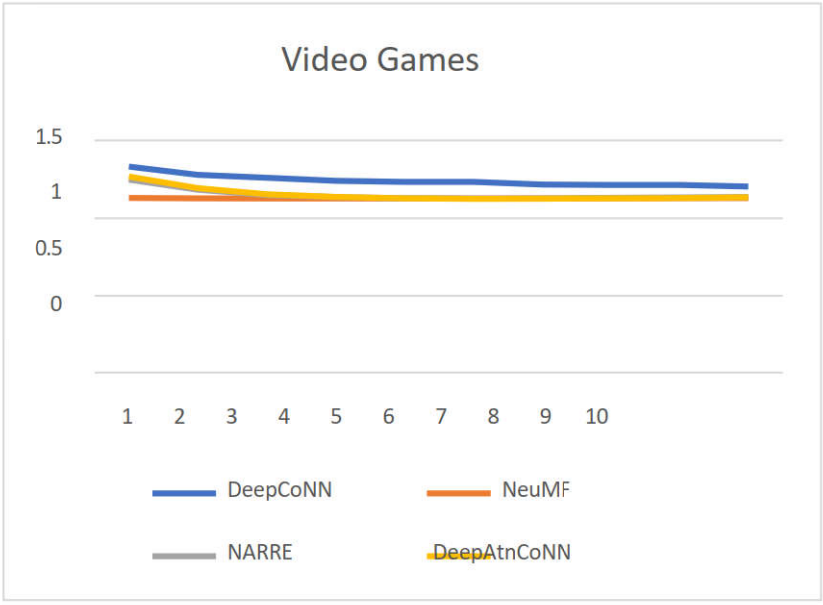


Рис.3.10 – Залежність MSE від кількості епох для датасету Video Games

Рис. 3.10 – Графік ілюструє зміну середньоквадратичної помилки (MSE) залежно від кількості епох навчання для датасету Video Games. Він показує, як точність моделі покращується в процесі навчання та досягає стабільного рівня після певної кількості епох.

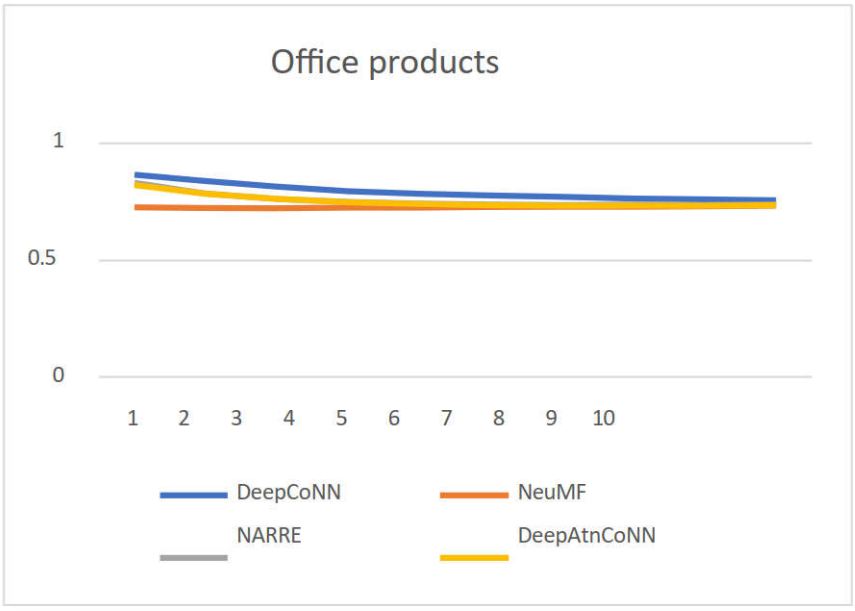


Рис.3.11 – Залежність MSE від кількості епох для датасету Office Products

Рис. 3.11 – Графік демонструє зміну середньоквадратичної помилки (MSE) залежно від кількості епох навчання для датасету Office Products. Він

відображає, як точність моделі покращується під час навчання та досягає стабільного рівня після певної кількості епох.

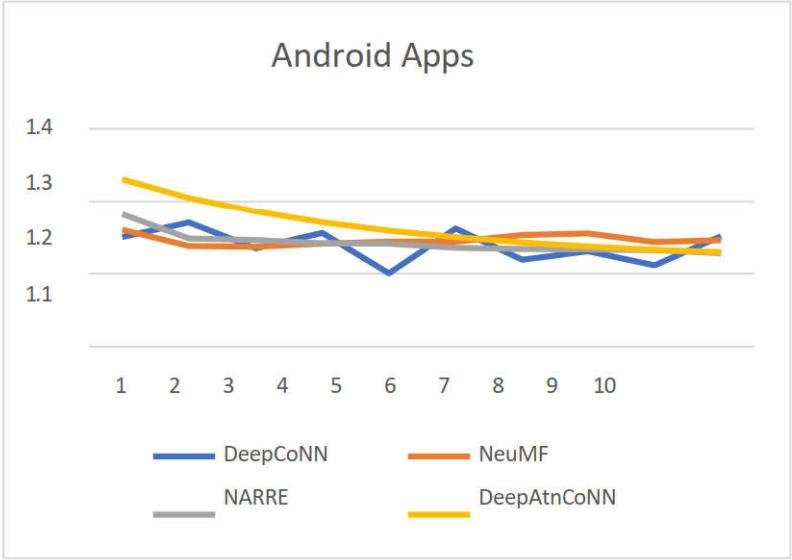


Рис.3.12 – Залежність NDCG від кількості епох для датасету Android Apps

Рис. 3.12 – Графік демонструє зміну метрики NDCG (Normalized Discounted Cumulative Gain) залежно від кількості епох навчання для датасету Android Apps. Він відображає покращення якості ранжування рекомендацій у процесі навчання моделі та стабілізацію показника після досягнення певного числа епох.

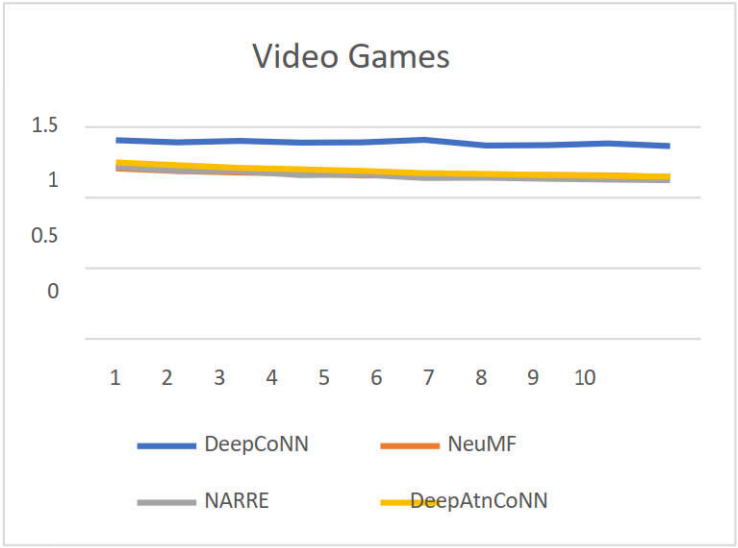


Рис.3.13 – Залежність NDCG від кількості епох для датасету Video Games

Рис. 3.13 – Графік показує залежність метрики NDCG (Normalized Discounted Cumulative Gain) від кількості епох навчання для датасету Video Games. Він ілюструє, як якість ранжування рекомендацій покращується під час навчання моделі і стабілізується після досягнення певного числа епох.

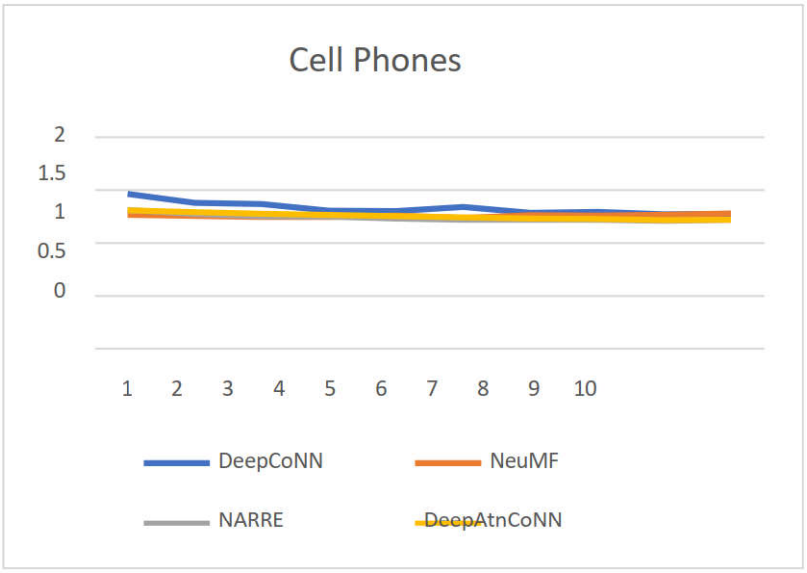


Рис.3.14– Залежність NDCG від кількості епох для датасету Cell Phones

Рис. 3.14 – Графік ілюструє зміну метрики NDCG (Normalized Discounted Cumulative Gain) залежно від кількості епох навчання для датасету Cell Phones. Він демонструє покращення якості ранжування рекомендацій у процесі навчання моделі та стабілізацію показника після досягнення певної кількості епох.

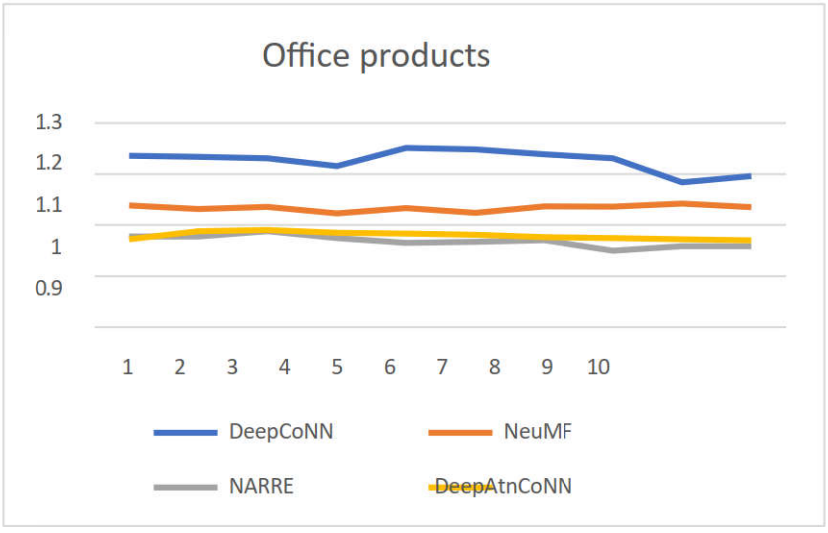


Рис.3.15– Залежність NDCG від кількості епох для датасету Office Products

Рис. 3.15 – Графік демонструє зміну метрики NDCG (Normalized Discounted Cumulative Gain) залежно від кількості епох навчання для датасету Office Products. Він показує, як покращується якість ранжування рекомендацій у процесі навчання моделі та коли показник досягає стабільного рівня.

Таким чином, можна зробити висновок, що запропонована система рекомендацій демонструє найвищу ефективність на великих датасетах, випереджаючи інші порівнянні моделі. Для невеликих наборів даних використання текстових оглядів не є критично необхідним, адже модель NeuMF, яка їх не враховує, показує приблизно такий же рівень точності.

Детальний аналіз поведінки моделі при різних значеннях Dropout показав, що оптимальна продуктивність досягається при Dropout = 0.2. У цьому режимі DeepAtnCoNN демонструє стабільно вищу точність у порівнянні з попередніми моделями, забезпечуючи більш надійні рекомендації для користувачів.

Графічне представлення порівняння точності для чотирьох моделей при різних значеннях Dropout наведено на Рисунках 57–68, що дозволяє наочно оцінити, як налаштування регуляризації впливає на ефективність рекомендацій. Ці результати підтверджують практичну користь запропонованої системи для інтернет-магазинів, де точні рекомендації можуть безпосередньо впливати на конверсію та задоволеність користувачів.

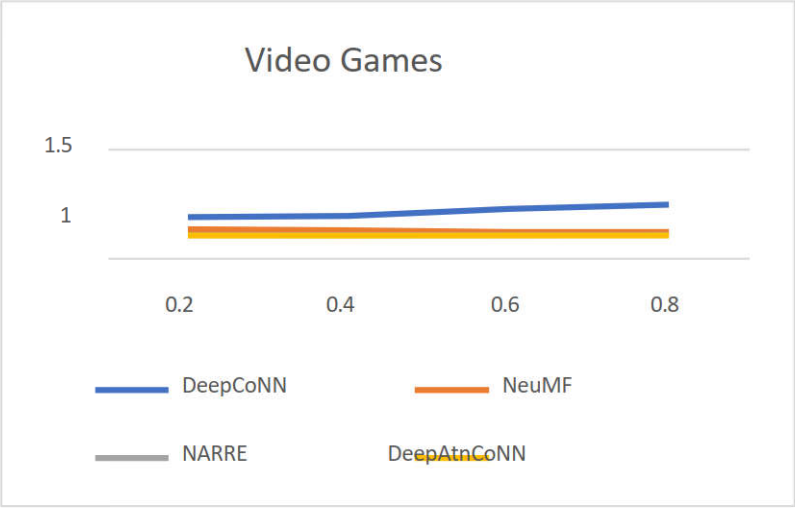


Рис.3.16 – Залежність MSE від Dropout для датасету Video Games

Рис. 3.16 – Графік показує залежність середньоквадратичної помилки (MSE) від значення параметра Dropout для датасету Video Games. Він демонструє вплив регуляризації на точність моделі та допомагає визначити оптимальні налаштування Dropout для покращення прогнозів.

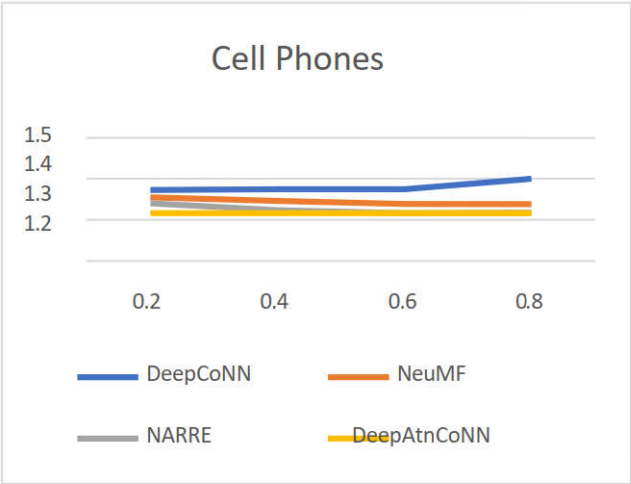


Рис.3.17– Залежність MSE від Dropout для датасету Cell Phones

Рис.3.17 – Графік демонструє залежність середньоквадратичної помилки (MSE) від значення параметра Dropout для датасету Cell Phones. Він ілюструє вплив регуляризації на точність моделі та допомагає визначити оптимальні налаштування Dropout для покращення результатів рекомендацій.

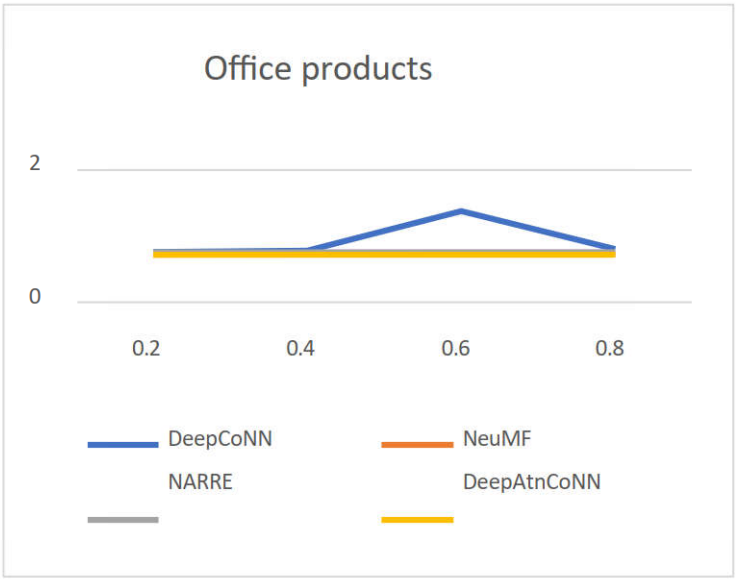


Рис.3.18 – Залежність MSE від Dropout для датасету Office Products

Рис. 3.18 – Графік ілюструє залежність середньоквадратичної помилки (MSE) від значення параметра Dropout для датасету Office Products. Він показує, як різні рівні регуляризації впливають на точність моделі та дозволяють визначити оптимальні налаштування для підвищення якості прогнозів.

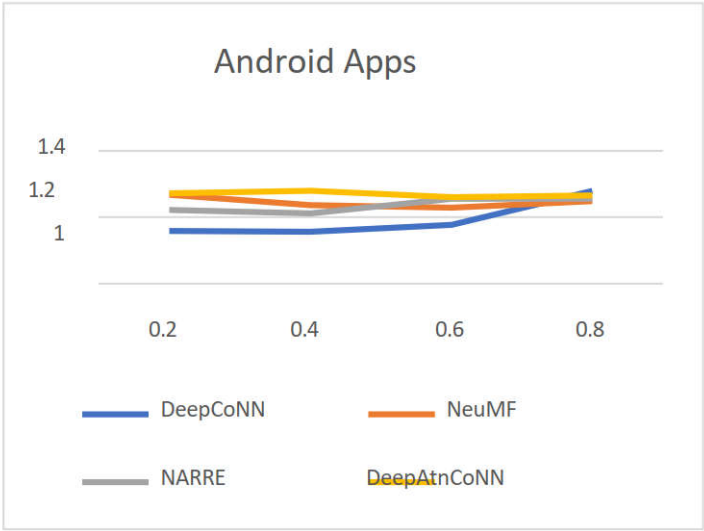


Рис.3.19 – Залежність NDCG від Dropout для датасету Android Apps

Рис. 3.19 – Графік показує залежність метрики NDCG (Normalized Discounted Cumulative Gain) від значення параметра Dropout для датасету Android Apps. Він демонструє, як рівень регуляризації впливає на якість ранжування рекомендацій та допомагає визначити оптимальні налаштування моделі.

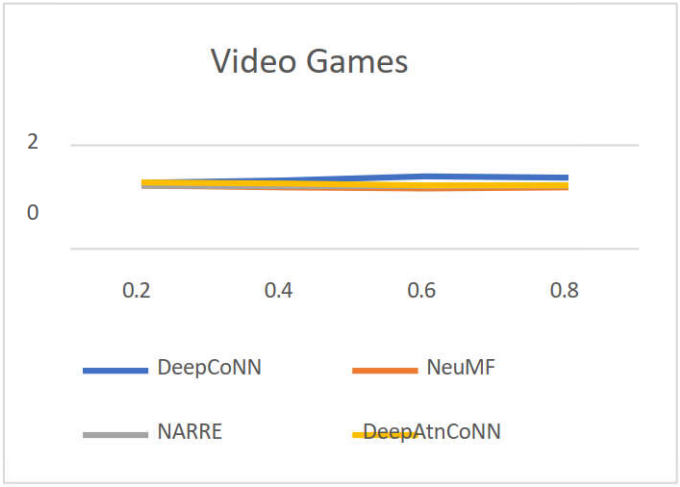


Рис.3.20 – Залежність NDCG від Dropout для датасету Video Games

Рис. 3.20 – Графік ілюструє залежність метрики NDCG (Normalized Discounted Cumulative Gain) від значення параметра Dropout для датасету Video Games. Він демонструє вплив регуляризації на якість ранжування рекомендацій тадопомагає визначити оптимальні параметри моделі.

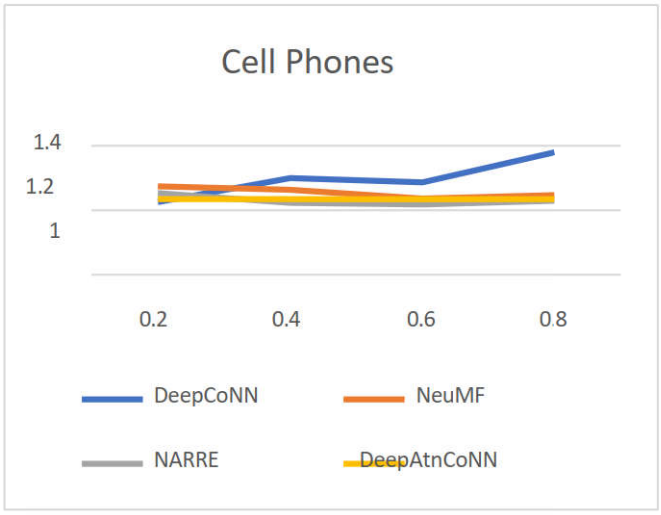


Рис.3.21– Залежність NDCG від Dropout для датасету Cell Phones

Рис. 3.21 – Графік показує залежність метрики NDCG (Normalized Discounted Cumulative Gain) від значення параметра Dropout для датасету Cell Phones. Він демонструє вплив регуляризації на якість ранжування рекомендацій та допомагає визначити оптимальні налаштування моделі для підвищення точності прогнозів.

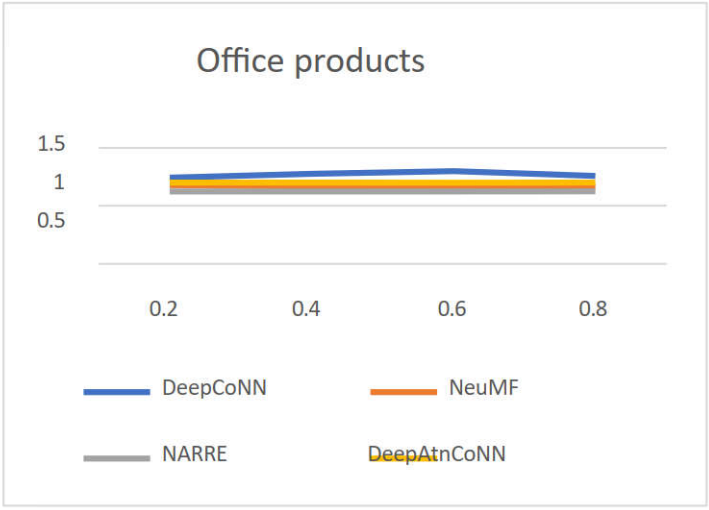


Рис.3.22– Залежність NDCG від Dropout для датасету Office Products

Рис. 3.22 – Графік ілюструє залежність метрики NDCG (Normalized Discounted Cumulative Gain) від значення параметра Dropout для датасету Office Products. Він показує, як рівень регуляризації впливає на якість ранжування рекомендацій та допомагає обрати оптимальні параметри моделі для підвищення точності прогнозів.

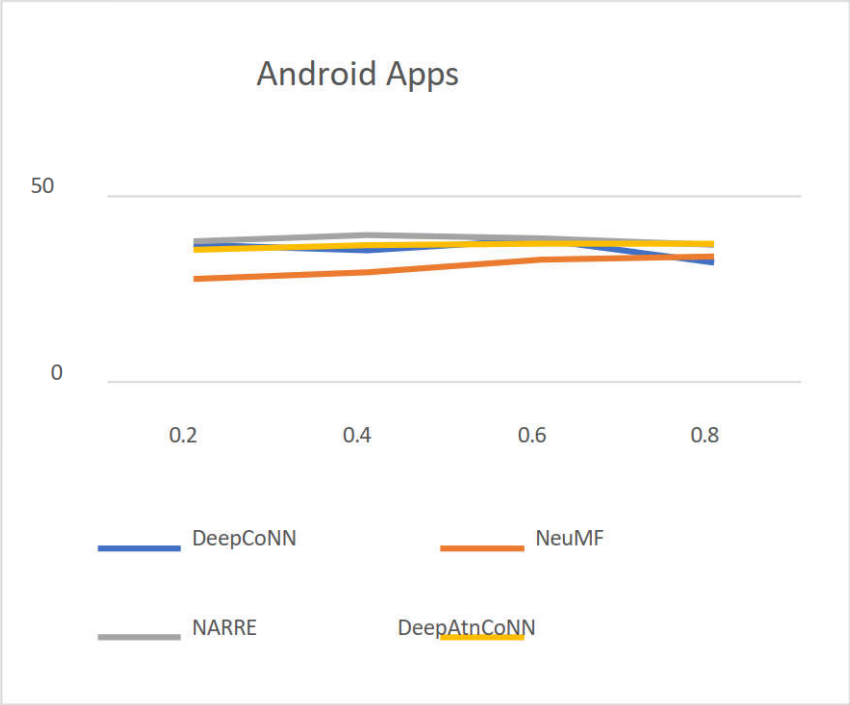


Рис.3.23 – Залежність HR@1 від Dropout для датасету Android Apps

На рис.3.23 по осі абсцис відкладені значення Dropout, а по осі ординат — значення HR@1. Графік ілюструє, як зміна рівня регуляризації (Dropout) впливає на якість моделі: за різних значень Dropout метрика HR@1 змінюється, відображаючи баланс між перенавчанням і недонавчанням. Як правило, за надто малого значення Dropout модель схильна до перенавчання, а за надто великого — втрачає здатність ефективно навчатися, що призводить до зниження HR@1. На графіку можна виділити область оптимальних значень Dropout, за яких досягається найкраще значення HR@1 для даного датасету.

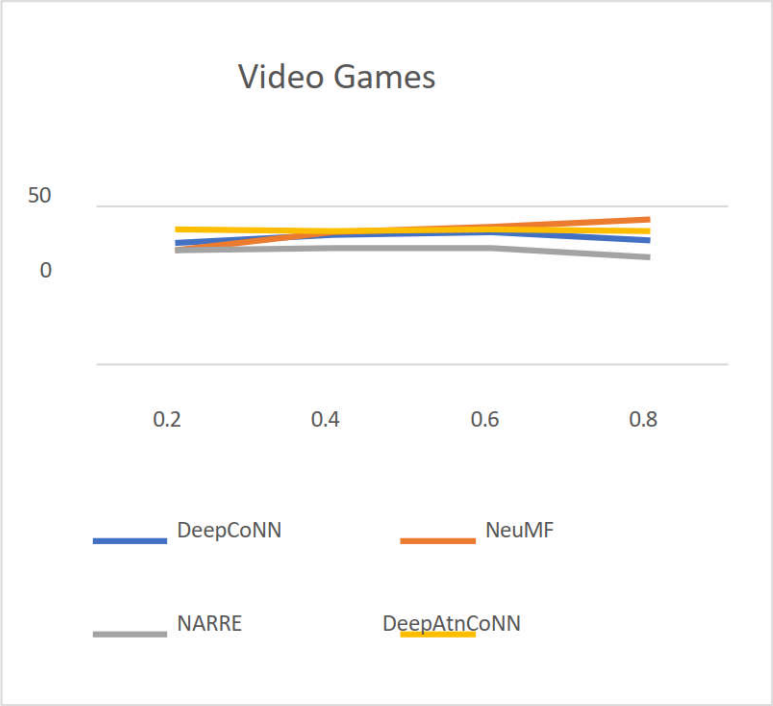


Рис. 3.24 – Залежність HR@1 від Dropout для датасету Video Games

Рис. 3.24 Графік показує залежність метрики NDCG (Normalized Discounted Cumulative Gain) від значення параметра Dropout для датасету Office Products. Графік демонструє, як рівень регуляризації впливає на якість ранжування рекомендацій, та дає змогу визначити оптимальні значення параметрів моделі з метою підвищення точності прогнозів.

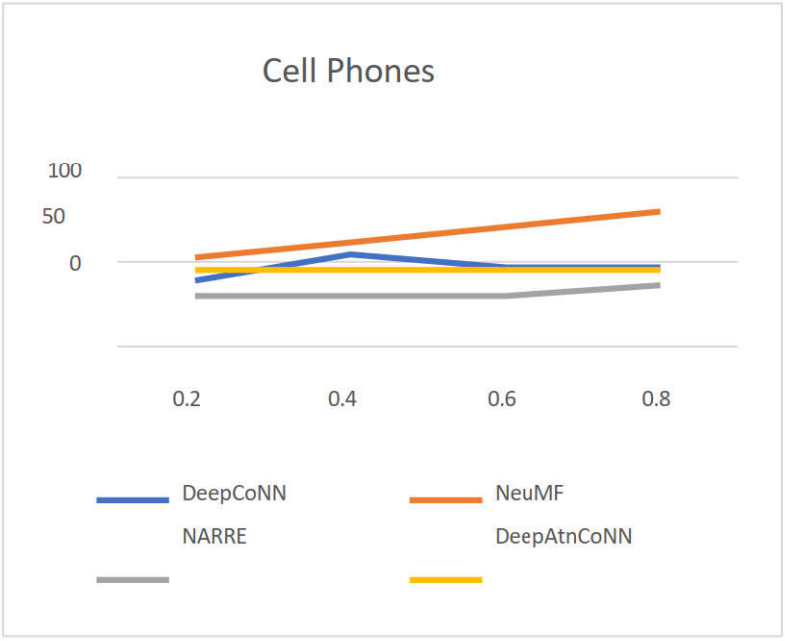


Рис.3.25 – Залежність HR@1 від Dropout для датасету Cell Phones

Рис. 3.25 – Графік показує залежність метрики HR@1 (Hit Rate на першій позиції) від значення параметра Dropout для датасету Cell Phones. Він демонструє, як різні рівні регуляризації впливають на ймовірність того, що найрелевантніший товар буде рекомендований користувачу на першій позиції.

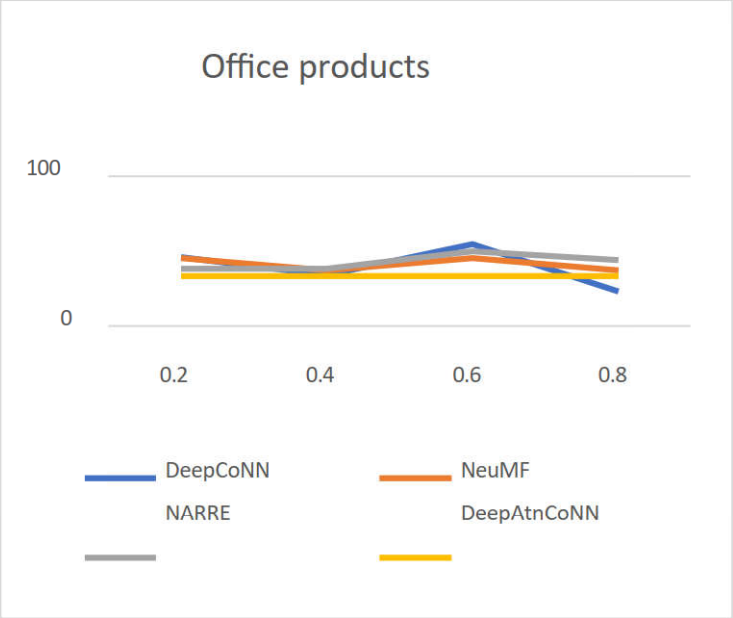


Рис.3.26– Залежність HR@1 від Dropout для датасету Office Products

Рис. 3.26 – Графік ілюструє залежність метрики HR@1 (Hit Rate на першій позиції) від значення параметра Dropout для датасету Office Products. Він демонструє, як рівень регуляризації впливає на ймовірність того, що найрелевантніший товар буде рекомендований користувачу на першій позиції, та допомагає визначити оптимальні налаштування моделі.

Щоб підвищити точність роботи алгоритмів рекомендацій у майбутньому, можна спробувати кілька напрямків, які виглядають особливо перспективно:

– Автокодери замість TextCNN. У моделях на кшталт NARRE можна

– застосувати автокодери для більш глибокого вивчення структури текстів відгуків. Це допомагає моделі краще «розуміти» контекст і виділяти

– найважливіші характеристики, що впливають на рекомендації.

– Більші та щільніші датасети. Чим більше даних для навчання, тим точніше працюватимуть моделі. Особливо це важливо для інтернет-магазинів з великою кількістю користувачів та продуктів.

– Двонаправлені рекурентні мережі з увагою (GRU/LSTM + Attention). Вони дозволяють моделі краще відслідковувати послідовність дій користувача та виділяти важливі моменти у його поведінці, що підвищує точність прогнозу вподобань.

– Використання оглядів як додаткового джерела інформації. Навіть ті алгоритми, що зазвичай не працюють із текстами, можуть стати точнішими, якщо враховувати відгуки користувачів як регуляризуючий фактор.

– Сучасні методи NLP для попередньої обробки текстів. Токенізація, усунення стоп-слів, Word2vec або навіть BERT допомагають моделі «зрозуміти» текст і коректно інтегрувати його у рекомендації.

Застосування хоча б однієї з цих ідей або їх комбінації може значно покращити точність прогнозів та підвищити якість рекомендацій для користувачів інтернет-магазинів, роблячи систему більш «розумною» та адаптивною до індивідуальних вподобань.

Результати експериментів були наочно представлені у вигляді таблиць та графіків. Наприклад, у Таблиці 3.1 наведено початкову точність моделей на кожному з чотирьох датасетів. Як видно, DeepAtnCoNN вже на старті демонструє невелике, але помітне покращення порівняно з DeepCoNN та NARRE. NeuMF зберігає стабільну продуктивність на невеликих датасетах, що пояснюється тим, що вона не використовує тексти оглядів, але все ж показує досить високі результати на *Office Products*.

На Рис. 3.1–3.15 зображено динаміку точності навчання моделей протягом епох. Можна побачити, що більшість моделей досягають стабільної продуктивності приблизно до п’ятої епохи, після чого подальше навчання дає

незначні поліпшення. DeepAtnCoNN при цьому демонструє більш стабільну криву точності та менше коливань, особливо на *Android Apps* та *Video Games*.

Додатково було проведено аналіз впливу Dropout на продуктивність моделей. Рис. 3.16–3.26 показують, що оптимальні значення Dropout для DeepAtnCoNN знаходяться близько 0.2, коли точність моделі перевищує інші підходи. Водночас, на великих датасетах вплив Dropout на продуктивність стає меншим, що свідчить про стабільність навчання запропонованої моделі.

Загалом, на основі проведених експериментів можна зробити такі висновки:

DeepAtnCoNN стабільно демонструє вищу точність на великих датасетах, де тексти відгуків мають суттєвий вплив на рекомендації.

Для невеликих наборів даних використання текстів не є критично необхідним — NeuMF показує порівнянний результат.

Оптимальні налаштування Dropout дозволяють підвищити стабільність та точність моделі, зменшивши ймовірність перенавчання.

Ці спостереження відкривають перспективи для подальшого розвитку системи, включаючи використання автокодерів, двонаправлених GRU/LSTM з Attention, а також інтеграцію додаткової інформації з оглядів для підвищення точності прогнозів та покращення якості рекомендацій для користувачів інтернет-магазинів.

ВИСНОВКИ

Проведене дослідження підтвердило, що персоналізація товарних пропозицій у сучасних інтернет-магазинах є ключовим фактором підвищення конкурентоспроможності. Врахування поведінки користувачів та текстових відгуків дозволяє підвищити ефективність рекомендаційних систем, збільшити конверсію, середній чек та лояльність клієнтів. Розроблена система демонструє, що інтеграція цих даних робить персоналізовані рекомендації більш точними та релевантними, що підтверджує актуальність обраної теми.

Під час роботи були досліджені класичні та сучасні алгоритми (DeepCoNN, NARRE, NeuMF), оцінено їхні переваги та обмеження, а також розроблено модифікацію DeepAtnCoNN, яка враховує текстові відгуки та поведінку користувачів для підвищення точності прогнозів.

Метою роботи було створення програмного додатку, здатного формувати персоналізовані рекомендації товарів з урахуванням дій і інтересів користувачів інтернет-магазину. Експериментальні результати підтвердили, що розроблена система забезпечує стабільно високий рівень точності рекомендацій на різних підмножинах датасету Amazon Products Data, демонструючи практичну реалізацію поставленої мети.

Аналіз сучасних підходів – проведено огляд алгоритмів рекомендацій, оцінено їхні сильні і слабкі сторони, виявлено потребу у вдосконаленні моделей для роботи з текстовими відгуками.

Розробка моделі системи – створено модифікацію DeepCoNN з механізмом Attention, що підвищує точність персоналізованих рекомендацій.

Порівняння існуючих рішень – експерименти показали перевагу DeepAtnCoNN над класичними методами DeepCoNN, NARRE та NeuMF, особливо на великих наборах даних.

Вибір інструментів та технологій – система реалізована на базі Python і PyTorch у середовищах Jupyter Notebook та Google Colab, що забезпечує її

гнучкість, масштабованість та інтеграцію з платформою інтернет-магазину тестування системи – протестовано систему на чотирьох підмножина.

Розробка іх датасету; підтверджено ефективність рекомендацій і стабільність роботи алгоритму.

Підготовка документації – розроблено користувацьку інструкцію для інтеграції та налаштування системи в реальних умовах.

Теоретичне значення полягає у вдосконаленні алгоритмів рекомендаційних систем шляхом інтеграції поведінкових даних користувачів та текстових відгуків із застосуванням механізмів Attention, що дозволяє підвищити точність прогнозів. Практичне значення полягає у створенні модульної системи, яка автоматично генерує персоналізовані пропозиції та може бути інтегрована в інтернет-магазини без змін структури платформи. Система включає:

- модуль збору даних;
- модуль обробки та нормалізації даних;
- модуль генерації рекомендацій на основі Collaborative Filtering, Content Based Filtering та DeepAtnCoNN;
- модуль оцінки точності рекомендацій;
- модуль інтеграції з інтерфейсом інтернет-магазину.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Білоус В. А. *Основи веб-розробки.* Київ: Наукова думка, 2019.
2. Бойко Ю. С. *Розробка веб-додатків на Python.* Київ: Генеза, 2021.
3. Бойчук М. В., Бойчук Л. П. *Python для початківців.* Львів: Видавництво Львівського університету, 2020.
4. Вовк В. І. Методи та засоби аналізу веб-ресурсів: сучасний стан та перспективи розвитку. *Інформаційні технології та засоби навчання,* 2020.
5. Гайдук А. Ю. *Вступ до інформатики та програмування.* Київ: Вища школа, 2021.
6. Гудзик О. С. *Бази даних і SQL: Підручник для студентів.* Київ: Академія, 2020.
7. Данильчук П. В. *Алгоритми та структури даних з Python.* Львів: Центр Європи, 2022.
8. Іваненко І. Г. *Сучасні технології веб-скрейпінгу з використанням Python.* Харків: Ранок, 2021.
9. Іванченко О. *Веб-скрейпінг для початківців.* Дніпро: Ліра, 2020.
10. Кравчук А. О. *Веб-розробка з Flask та Django.* Харків: Основа, 2021.
11. Литвиненко Д. П. *Системи управління базами даних.* Одеса: Південний полюс, 2019.
12. Михайлюк І. М. *Основи програмування на Python.* Київ: Політехніка, 2021.
13. Пилипчук І. І. *Основи комп'ютерних наук.* Тернопіль: Навчальна книга, 2021.
14. Семенова Л. Г., Кравченко В. Ю. Використання хмарних обчислень у веб-аналітиці. *Вісник Харківського національного університету,* 2021.
15. Степаненко Р. А. *SEO для веб-розробників.* Київ: Освіта України, 2022.
16. Шевченко Н. М. *Бази даних для початківців.* Харків: Видавничий дім "Школа", 2019.

17. Ahrefs. *Complete Guide to SEO.* URL: https://ahrefs.com/blog/ (дата звернення: 25.10.2024)

18. Alexandrov, N. *Machine Learning for Web Applications.* Journal of AI Research, 2021.

19. Bautista, J. *Beautiful Soup for Beginners.* Kindle Edition, 2021.

20. BeautifulSoup Documentation. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (дата звернення: 25.10.2024)

21. Beazley, D., Jones, B. K. *Python Cookbook.* O'Reilly Media, 2020.

22. Brown, T. *Data Structures and Algorithms with Python.* Wiley, 2022.

23. DeepCrawl. URL: https://www.deepcrawl.com/ (дата звернення: 25.10.2024)

24. Docker Inc. URL: https://docs.docker.com/ (дата звернення: 25.10.2024)

25. Flask Documentation. URL: https://flask.palletsprojects.com/ (дата звернення: 25.10.2024)

26. GitHub Documentation. URL: https://docs.github.com/ (дата звернення: 25.10.2024)

27. Google AI. TensorFlow Documentation. URL: https://www.tensorflow.org/ (дата звернення: 25.10.2024)

28. Google Developers. Lighthouse Documentation. URL: https://developers.google.com/web/tools/lighthouse (дата звернення: 25.10.2024)

29. Google PageSpeed Insights. URL: https://developers.google.com/speed/pagespeed/insights/ (дата звернення: 25.10.2024)

30. Google Search Console. URL: https://support.google.com/webmasters/answer/9128668 (дата звернення: 25.10.2024)

31. Grinberg, M. *Flask Web Development.* O'Reilly Media, 2021.

32. Gurnani, N. *Mastering Flask.* Packt Publishing, 2020.

33. IBM Cloud Docs. Watson Natural Language Processing Documentation. URL: https://cloud.ibm.com/docs/natural-language-understanding (дата звернення: 25.10.2024)

34. Johnson, R. *Practical Guide to Web Development with JavaScript and Node.js.* O'Reilly Media, 2021.

35. Kimball, R., Ross, M. *The Data Warehouse Toolkit.* Wiley, 2019.

36. Kwon, Y. SEO Optimization Techniques for Modern Websites. *Journal of Digital Marketing,* 2021.

37. Lee, H. Exploring Python-based Web Scraping Techniques. *Web Engineering Journal,* 2019.

38. Lubbers, P., Albers, B. *Pro HTML5 Programming.* Apress, 2021.

39. Mozilla Developer Network. URL: https://developer.mozilla.org/ (дата звернення: 25.10.2024)

40. Nageshwar Rao, B. *Database Management Systems.* New Age International, 2021.

41. Patel, R., Khan, M. Comparative Analysis of Web Scraping Frameworks: Scrapy vs. BeautifulSoup. *Journal of Web Engineering,* 2022.

42. PostgreSQL Global Development Group. URL: https://www.postgresql.org/docs/ (дата звернення: 25.10.2024)

43. Python Software Foundation. URL: https://docs.python.org/3/ (дата звернення: 25.10.2024)

44. Richardson, L., Amundsen, M. *RESTful Web APIs.* O'Reilly Media, 2021.

45. Screaming Frog. SEO Spider Tool Documentation. URL: https://www.screamingfrog.co.uk/seo-spider/ (дата звернення: 25.10.2024)

46. Smith, J. Performance Optimization of Web Applications using Flask. *International Journal of Computer Science,* 2020.

47. Tiwari, S., Bhasin, S. Web Scraping and Data Extraction with BeautifulSoup. *Journal of Data Science,* 2021.

48. VanderPlas, J. *Python Data Science Handbook.* O'Reilly Media, 2019.