МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет Комп'ютерних наук
Кафедра Програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

другий (магістерський)
(рівень вищої освіти)

Дослідження алгоритмів машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача

Виконав:
студент 2 курсу групи ІПЗм-21-1

Буценко М. О.
(прізвище, ініціали)

Спеціальність	121 – Інженерія програмного забезпечення
Тип програми	Освітньо-наукова
Керівник	доц. Афанасьєва І. В.

(підпис) (посада, прізвище, ініціали)

Допускається до захисту

Зав. Кафедри _____

2023 р. З.В. Дудар

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Програмної інженерії

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-наукова програма

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«__»_____2023 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студента Буценка Микити Олеговича
(прізвище, ім'я, по батькові)

1. Тема роботи: Дослідження алгоритмів машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача затверджена наказом університету від «_29__»_березня_____ 2023 __ р. №_302 Ст__
2. Термін подання студентом роботи до екзаменаційної комісії 2 3 т р а в н я 2023 р.
3. Вихідні дані до роботи встановлений календарний план роботи, методичні вказівки до оформлення пояснювальної записки, методи прогнозування засновані на методах рекомендаційних алгоритмів.
4. Перелік питань, що потрібно опрацювати в роботі мета роботи, аналіз предметної галузі і постановка задачі, огляд наявних математичних моделей, модифікація базових алгоритмів, дослідження можливості прискорення базових моделей, створення плану для подальшого дослідження теми.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі	06.02.2023	виконано
2	Аналіз поточного стану індустрії	16.04.2023	виконано
3	Дослідження рекомендаційних систем	30.04.2023	виконано
4	Підготовка пояснювальної записки	11.05.2023	виконано
5	Підготовка презентації та доповіді	11.05.2023	виконано
6	Перевірка на академічний плагіат	12.05.2023	виконано
7	Нормконтроль	15.05.2023	виконано
8	Рецензування	18.05.2023	виконано
9	Занесення диплома в електронний архів	19.05.2023	виконано
10	Попередній захист	20.05.2023	виконано
11	Допуск до захисту у зав. кафедри	21.05.2022	виконано
12	Захист кваліфікаційної роботи	24.05.2023	виконано

Дата видачі завдання 11 листопада 2022 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Афанасьєва І. В
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Кваліфікаційна робота магістра містить: 66 с., 17 рис., 21 джер.

АЛГОРИТМИ, АНАЛІЗ ДАНИХ, БІБЛІОТЕКИ, МАШИННЕ НАВЧАННЯ, МЕДІА, МУЗИКА, НЕЙРОННА МЕРЕЖА, ПІДБІР РЕКОМЕНДАЦІЙ

Об’єктом дослідження є методи прогнозування машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача.

Метою роботи є проведення підготовчих робіт для дослідження щодо ефективності алгоритмів машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача, які приймають участь у процесі рекомендації контенту майже на всіх мультимедійних платформах.

У результаті роботи була здійснена підготовча робота для подальшого дослідження та розроблена документація для майбутньої системи підтримки прийняття рішень.

ALGORITHMS, DATA ANALYSIS, LIBRARIES, MACHINE LEARNING, MEDIA, MUSIC, NEURAL NETWORK, SELECTION OF RECOMMENDATIONS

The object of the research is machine learning prediction methods for selecting music recommendations based on user preferences.

The purpose of the work is to conduct preparatory work for research on the effectiveness of machine learning algorithms for selecting music recommendations based on user preferences, who participate in the content recommendation process on almost all multimedia platforms.

As a result of the work, preparatory work was carried out for further research, and documentation was developed for the future decision support system.

Умови публікації пояснювальної записки

Я, Буценко Микита Олегович,

(прізвище, ім’я, по батькові)

студент(ка) групи ІПЗм-21-1, здобувач вищої освіти на другому (магістерському) рівні, кафедри Програмної інженерії

(назва кафедри)

заявляю: моя кваліфікаційна робота на тему «Дослідження алгоритмів машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача»

(назва роботи)

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

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

ЗМІСТ

ВСТУП

Кожна мультимедіа-платформа, яка представляє собою бібліотеку різноманітного контенту на кшталт відео, аудіо, ігор тощо в першу чергу функціонує саме за допомогою системи пропозиції контенту. Такі додатки в якості першої, основної сторінки пропонують користувачам «рекомендований» контент, причому такі рекомендації зазвичай базуються на уподобаннях користувача.

Як ці уподобання можна вирахувати в математичній формі, щоб мати змогу запропонувати найбільш відповідний контент згідно уподобань користувача? Cristos Goodrow, VP of Engineering At YouTube (Крістос Гудроу, віце-президент з інженерних питань YouTube), у своїй статті On YouTube's recommendation system (Про систему рекомендацій YouTube), розповідає наступне: «У 2008 році, коли ми вперше почали створювати нашу систему рекомендацій, наше бачення системи рекомендацій було зовсім іншим. Припустімо, ви здебільшого дивитеся кулінарні відео. Чи не було б неприємно, якби ваша домашня сторінка рекомендувала вам лише останні спортивні та музичні відео, оскільки вони мали найбільше переглядів? Це був YouTube на самому початку свого життєвого циклу. Система ранжувала відео на основі популярності, щоб створити одну велику сторінку «Тенденції». Небагато людей переглянули ці відео, і більшість переглядів на YouTube прийшли через пошук або спільні посилання за межами платформи.» [1]

Можна сказати, що мало місце спрощення, яке можна описати виразом «чим більше переглядів/прослуховувань, тим активніше просування у стрічці рекомендацій». З одного боку, така модель дуже проста та зрозуміла як користувачам, так і розробникам, але очевидно, що вона не може повноцінно задовольнити потреби користувачів платформ. Тим не менш, за останні роки розробники таких додатків досягнули значного прогресу у вирішенні проблеми рекомендації контенту.

На сьогоднішній день майже всі популярні платформи на кшталт YouTube, Spotify, Instagram, Tiktok рекомендують контент завдяки нейронним мережам, які аналізують поведінку користувача, канали та плейлисти, які він переглядає та

прослуховує, і на основі даної вибірки підбирають саме той контент, який, на думку системи, підійде користувачеві. Кожна платформа має власний варіант реалізації такої мережі, а також відповідне API для сторонніх розробників. Наприклад, Spotify API, яке відповідає за видачу рекомендацій, може бути використане для інших додатків, орієнтованих на музику, наприклад для підбору найкращих треків певних виконавців чи жанрів.

Метою курсової роботи є проведення підготовчих робіт для дослідження алгоритмів машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача, порівняння їх реалізації та показників ефективності.

Предметною галуззю чинного дослідження з огляду на окреслену проблематику, є галузь мультимедіа контенту. Дане дослідження допоможе іншим розробникам виявити сильні та слабкі сторони досліджених алгоритмів, а також виявити доцільність їх використання, часткового чи повного, у власних додатках, які мають функціонал рекомендації контенту.

1. ОПИС ПРОБЛЕМНОЇ ГАЛУЗІ

1.1 Аналіз предметної області

Кожна мультимедіа-платформа, яка представляє собою бібліотеку різноманітного контенту на кшталт відео, аудіо, ігор тощо в першу чергу функціонує саме за допомогою системи пропозиції, або рекомендації контенту. Такі додатки в якості першої, основної сторінки пропонують користувачам «рекомендований» контент, причому такі рекомендації зазвичай базуються на уподобаннях користувача.

На самому початку реалізація алгоритмів підбору контенту була наступною: припустімо, ви здебільшого дивитеся кулінарні відео. Чи не було б неприємно, якби ваша домашня сторінка рекомендувала вам лише останні спортивні та музичні відео, оскільки вони мали найбільше переглядів? Це був YouTube на самому початку свого життєвого циклу. Система ранжувала відео на основі популярності, щоб створити одну велику сторінку «Тренди». Перегляди (або прослуховування, якщо мова йде про музичні треки чи подкасти) були стабільно низькими через низьку релевантність контенту, і більшість переглядів на YouTube створювались через ручний пошук або спільні посилання за межами платформи.

Можна сказати, що мало місце спрощення, яке можна описати виразом «чим більше переглядів/прослуховувань, тим активніше просування у стрічці рекомендацій». З одного боку, така модель дуже проста та зрозуміла як користувачам, так і розробникам, але очевидно, що вона не може повноцінно задовольнити потреби користувачів платформ. Тим не менш, за останні роки розробники таких додатків досягнули значного прогресу у вирішенні проблеми рекомендації контенту.

На сьогоднішній день майже всі популярні платформи на кшталт YouTube, Spotify, Instagram, TikTok рекомендують контент завдяки нейронним мережам, які аналізують поведінку користувача, канали та плейлисти, які він переглядає та прослуховує, і на основі даної вибірки вони підбирають саме той контент, який, на думку системи, підходить користувачеві. Кожна платформа має власний варіант реалізації такої мережі, а також відповідне API для сторонніх розробників.

Наприклад, Spotify API, яке відповідає за видачу рекомендацій, може бути використане для інших додатків, орієнтованих на музику, наприклад для підбору найкращих треків певних виконавців чи жанрів.

Здійснимо поступовий огляд кожної зазначеної системи, аби з'ясувати на яких показниках вони концентруються та які алгоритму використовують. Останнє може бути частиною комерційної таємниці компанії, тому ця інформація не завжди є у відкритих джерелах.

Почнемо з YouTube (див. рис. 1.1). Їх система рекомендацій побудована на простому принципі допомоги у пошуку відео, які вони хочуть переглянути, і яке принесе їм певну користь.

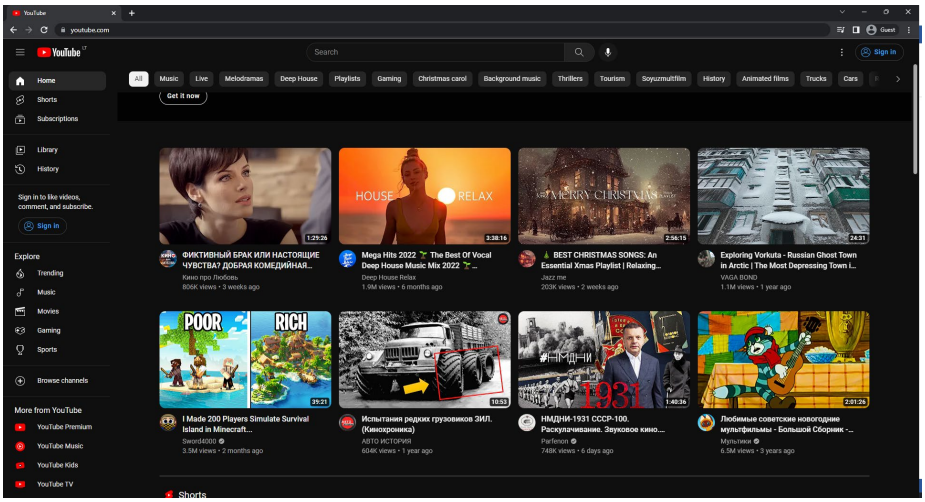

Рисунок 1.1 – інтерфейс ПК-версії YouTube у режимі Guest-mode (youtube.com)

Слід зазначити, що в даному контексті слово «користь» має на увазі не лише практичну користь, а й таке поняття, як «цей контент заспокоює, розважає чи просто цікавий». Рекомендований контент у додатку знаходиться в двох основних місцях: на домашній сторінці, та на панелі «Далі». Домашня сторінка – це те, що користувач бачить, коли вперше відкриває YouTube (переходить за посиланням youtube.com). Вона відображає суміш персоналізованих рекомендацій, підписок, а також останніх новин та повідомлень. Панель «Далі» з'являється, коли користувач переглядає відео, і пропонує додатковий вміст на основі того, що користувач зараз переглядає, а також інші відео, які, на думку розробників, можуть його зацікавити. [2]

YouTube порівнює переглянуті аудіо та відео з тими, які схожі на ваші, і використовує цю інформацію, щоб запропонувати інший контент, який ви можете переглянути. Наприклад, якщо вам подобаються hip-hop-треки, і наша система помічає, що інші, яким подобаються ті самі hip-hop треки, що й вам, також прослуховують джаз, користувачеві система може порекомендувати джазові кліпи, навіть якщо він ніколи раніше не слухав жодного джазового виконавця. Історію реалізації алгоритмів рекомендації можна побачити на рисунку 1.2.

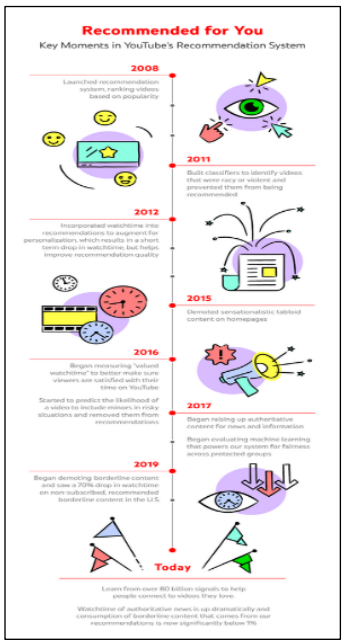

Рисунок 1.2 – Спрощена історія реалізації алгоритмів рекомендації

Стосовно політики конфіденційності: компанії Google – яка, власне, і володіє мережею Youtube – відомо, що не всі користувачі хочуть ділитися з розробниками інформацією про історію своїх переглядів та уподобань. На цей випадок передбачено елементи керування, які допоможуть користувачу вирішити, скільки даних він хочете надати. Він може будь-коли призупинити збереження історії переглядів, а також відредагувати її вміст або видалити її разом з історією пошуку. [3]

Тепер перейдемо до огляду мережі рекомендацій Spotify. На відміну від YouTube та TikTok, Spotify не надає повного доступу до нещодавно оприлюдненої внутрішньої документації (Spotify API), щоб не розкривати структуру системи. Те, що ми маємо, – це широкі публічні записи компанії про дослідження та

розробки, та, власне, її API. Це не означає, що ми не знаємо жодних технічних деталей про те, як працює система – насправді, значна частина рекомендаційного підходу Spotify була широко оприлюднена – але в рамках даного дослідження нам доведеться опуститися в область обґрунтованих припущень, коли справа доходить до деяких більш детальних подробиць побудови системи рекомендацій Spotify.

Загалом, у основі системи рекомендацій цього музичного додатку лежить модель Machine Learning, оптимізована для ключових бізнес-цілей: утримання користувачів; збільшення середнього часу, проведеного на платформі, і, зрештою, збільшення отриманого доходу. Для того, щоб забезпечити максимально ефективну та якісну роботу системи рекомендацій Spotify, вона має розуміти вміст того контенту, який вона рекомендує, а також уподобання користувачів, яким вона його рекомендує. З обох сторін цієї пропозиції Spotify використовує кілька незалежних моделей і алгоритмів машинного навчання для створення представлень контенту, або музичних композицій, і представлень користувачів. Підхід Spotify до представлення треків складається з двох основних компонентів [4]:

- фільтрація на основі вмісту, спрямована на опис треку шляхом вивчення самого вмісту;
- спільна фільтрація, спрямована на опис доріжки в її зв’язку з іншими доріжками на платформі шляхом вивчення ресурсів, створених користувачами.

Механізм рекомендацій потребує даних, згенерованих обома методами, щоб отримати цілісне уявлення про вміст платформи та вирішити проблеми «холодного старту» під час роботи з нещодавно завантаженими на платформу композиціями. Алгоритми фільтрації базуються на основі:

- аналізу метаданих виконавця;
- аналізу необроблених аудіо-сигналів.

На рисунку 1.3 наведений аналіз композиції Lil Nas X – Industry Baby. Він наглядно демонструє те, як система розбиває один трек на декілька рівнів, а

кожен із них розбитий на певні секції.

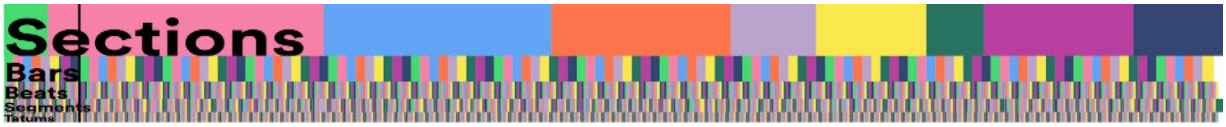

Рисунок 1.3 – Аналіз тимчасового аудіо, виконаний з Spotify Audio Analysis

Завдяки такому комплексному аналізу і забезпечується чітка та злагоджена робота обох ключових моделей мережі – представлень музичних композицій та представлень користувачів, які в сукупності створюють систему рекомендацій Spotify.

1.2 Постановка задачі

З огляду на інформацію, наведену у попередньому пункті можна сформувати задачу чинної роботи – проведення дослідження, яке порівнює ефективність застосування алгоритмів машинного навчання на прикладі YouTube та Spotify для підбору рекомендацій музики з урахуванням уподобань користувача, які приймають участь у процесі рекомендації контенту майже на всіх мультимедійних платформах.

У даному випадку, під терміном «ефективність» мається на увазі наступний набір показників:

- тривалість навчання нейронної мережі;
- точність рекомендаційних алгоритмів;
- час на підготовку test та train даних;
- вага певних вхідних показників.

Для виконання даної задачі, доцільно її розділення на кілька підзадач:

- здійснити ознайомлення з математичним представленням сімейства рекомендаційних алгоритмів;
- перевірити можливість для подальшої оптимізації моделі;
- побудувати план експерименту дослідження, що передбачає визначення загальних умов, формалізацію функцію ефективності, визначення правил порівняння моделей та опис можливих помилок та невизначеностей;

– проведення експерименту;

– формалізація результатів експерименту.

Тут варто зауважити, що підготовчими кроками вважаються усі, що розміщені до плану експерименту дослідження.

У багатьох областях даного дослідження виникають проблеми, які потребують дослідження та подальшого вирішення. Ці проблеми можуть бути складними та багатогранними, та вимагати детального розуміння основних факторів і потенційних рішень. Без чітко визначеної постановки проблеми дослідженню може бракувати цілеспрямованості та спрямованості, що призведе до неефективних або непереконливих результатів. Таким чином, надзвичайно важливо розробити чітке та стисле формулювання проблеми, щоб переконатися, що дослідження проводяться ефективно та результативно.

2. МАТЕМАТИЧНЕ ПРЕДСТАВЛЕННЯ

2.1 Математична характеристика рекомендацій YouTube

Почнемо з розглядання алгоритмів рекомендації платформи YouTube.

Загальну структуру нашої системи рекомендацій показано на рисунку 2.1. Система складається з двох нейронних мереж: одна для формування кандидатів і друга для рейтингу.

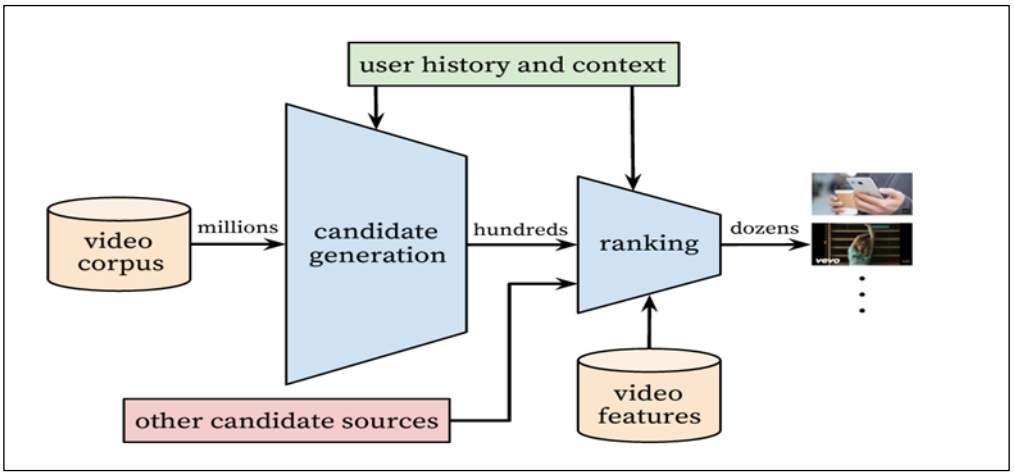

Рисунок 2.1 – Загальна архітектура системи рекомендацій YouTube

Мережа створення кандидатів приймає події з історії активності користувача на YouTube як вхідні дані та отримує невелику підмножину (сотні) відео з великого зведення даних. Ці кандидати мають бути загально доречними (релевантними) для користувача, та мати високу точність релевантності.

Мережа створення кандидатів забезпечує лише широку персоналізацію за допомогою спільної фільтрації. Подібність між користувачами виражається в таких грубих характеристиках, як ідентифікатори переглядів відео, маркери пошукових запитів і демографічні дані. Представлення кількох «найкращих» рекомендацій у списку вимагає точного представлення, щоб виділити відносну важливість серед кандидатів із високим рівнем запам’ятовування. За виконання цього завдання мережа рейтингування призначає оцінку кожному відео відповідно до бажаної цільової функції, використовуючи багатий набір характеристик, що описують відео та користувача. Відео з найвищими балами представлені користувачеві, будучи відсортованими за їхніми балами: відео з

найбільшим «рейтингом» йдуть першими.

Двоетапний підхід до рекомендацій дозволяє надавати рекомендації з дуже великої вибірки аудіо- та відео-контенту (розмір вибірки – мільйони записів), залишаючись впевненими, що та невелика кількість відео, які з'являються на пристрої користувача, є персоналізованими та цікавими саме для нього. Крім того, цей підхід дозволяє змішувати кандидатів, згенерованих іншими джерелами [5]. Під час розробки широко використовуються офлайн-метрики (точність, запам'ятовуваність, втрата рейтингу тощо), з метою ітераційного вдосконалювання системи рекомендацій. Однак для остаточного визначення ефективності алгоритму чи моделі розробники покладаються на A/B-тестування за допомогою експериментів у реальному часі. Під час експерименту в реальному часі вони можуть виміряти незначні зміни в рейтингу кліків, тривалості перегляду та багатьох інших показниках, які вимірюють залучення користувачів. Це важливо, оскільки живі результати A/B не завжди співвідносяться з офлайн-експериментами.

2.2 Генерація кандидатів

Під час генерації кандидатів величезне зведення даних YouTube зводиться до сотень відео, які можуть бути релевантними для користувача. Попередником описаного тут рекомендаційного алгоритму був підхід матричної факторизації, навчений за допомогою техніки втрати рангу [6]. Ранні ітерації моделі нейронної мережі YouTube імітували цю поведінку факторизації з неглибокими мережами, які вбудовували лише попередні перегляди користувача. З цієї точки зору новий підхід можна розглядати як нелінійне узагальнення методів факторизації. На рисунку 2.1 зображена загальна архітектура системи рекомендацій, що демонструє «послідовність», де аудіо- та відео-кандидати витягуються та ранжуються перед тим, як користувачеві буде представлено лише декілька з матеріалів. Після ранжування, система використовує додаткові фільтри, зокрема персоналізовані вибори користувача та його історію перегляду, для подальшого вибору найбільш релевантного контенту. Завдяки цьому механізму, користувачеві

пропонуються лише ті відео або аудіо, які найбільше відповідають його інтересам та вподобанням.

Розробники YouTube представляють рекомендацію як крайню багатокласову класифікацію, де проблема передбачення стає точною класифікацією переглядів конкретне аудіо- чи відео w_t в час t серед мільйонів інших відео i (класів) із зрізу даних V на основі користувача U і контексту C (див. рис. 2.2)

$$P(w_t = i|U, C) = \frac{e^{v_i u}}{\sum_{j \in V} e^{v_j u}}$$

Рисунок 2.2 – представлення рекомендаційної моделі

де $u \in \mathbb{R}^N$ являє собою багатовимірне вбудовування користувача, контекстної пари, та $\bar{v}_j \in \mathbb{R}^N$ – представлення вбудовування кожного кандидата (мається на увазі аудіо- чи відео-ролику).

У цьому налаштуванні є вбудовування простого зіставлення розріджених об’єктів (окремих відео, користувачів тощо) у щільний вектор у R^N. Завдання глибинної нейронної мережі полягає в тому, щоб вивчати вбудовування користувача u як функцію історії користувача та контекст, причому обидві ці характеристики корисні для розрізнення серед безлічі відео з класифікатором softmax.

Хоча на YouTube існують чіткі механізми зворотного зв’язку (великі пальці вгору/вниз, опитування на сторінках Спільноти тощо) розробники використовують неявний зворотний зв'язок [7] часу, який користувач витратив на перегляд відео, для навчання моделі, де позитивним результатом є те, що користувач повністю переглянув відео (недаремно автори контенту часто нагадують глядачам «подивитися ролик до кінця»). Цей вибір базується на більш неявній доступній історії користувача, що дозволяє розробникам просувати рекомендації глибоко «в хвіст», де явний зворотний зв'язок надзвичайно

рідкісний.

2.3 Efficient Extreme Multiclass, CGM та CCG

Для ефективного навчання моделі з мільйонами класів розробники покладаються на техніку вибірки негативних класів із фонового розподілу («вибірка кандидатів»), а потім коригують цю вибірку за допомогою зважування важливості. Для кожного прикладу втрата перехресної ентропії мінімізована для істинної мітки та вибірки негативних класів. На практиці відбирається кілька тисяч негативів, що відповідає більш ніж 100-кратному прискоренню порівняно з традиційним softmax. Популярним альтернативним підходом є ієрархічний softmax [8], але з ним розробники YouTube не змогли досягти тієї ж точності прогнозування. В ієрархічному softmax обхід кожного вузла в дереві передбачає розрізнення між наборами класів, які часто не пов’язані, що значно ускладнює проблему класифікації та погіршує продуктивність.

Безпосередньо під час роботи додатку розробникам потрібно обчислити найімовірніші N класів (аудіо- чи відео), щоб вибрати N найкращих для представлення користувачеві. Оцінка мільйонів елементів із суворою затримкою обслуговування в десятки мілісекунд вимагає приблизної схеми оцінки, сублінійної за кількістю класів. Попередні системи на YouTube покладалися на хешування [9], а описаний тут класифікатор використовує подібний підхід. Оскільки відкалібровані ймовірності з вихідного рівня softmax не потрібні під час обслуговування, проблема оцінки зводиться до пошуку найближчого сусіда в просторі скалярного добутку, для якого можна використовувати бібліотеки загального призначення [10]. Під час тестування розробники виявили, що результати A/B не були особливо чутливими до вибору алгоритму пошуку найближчого сусіда.

Натхненні мовними моделями безперервної сумки слів [11], розробники створили власний алгоритм, за допомогою якого вони вивчають багатовимірні вбудовування для кожного аудіо- чи відео-матеріалу у фіксованому словнику та вводять ці вбудовування в нейронну мережу прямого зв’язку. Історія переглядів

користувача представлена послідовністю розріджених ідентифікаторів відео змінної довжини, яка зіставляється з щільним векторним представленням за допомогою вставок. Мережа потребує щільних вхідних даних фіксованого розміру та простого усереднення найкращих впроваджень серед кількох стратегій (сума, максимум по компонентах тощо). Важливо, що вбудовування вивчаються разом з усіма іншими параметрами моделі за допомогою нормальних оновлень зворотного поширення градієнтного спуску. Об'єкти об'єднані в широкий перший шар, за яким слідує кілька шарів повністю з'єднаних випрямлених лінійних одиниць (ReLU) [12]. На рисунку 2.3 показана загальна архітектура мережі з додатковими функціями, не пов'язаними з переглядом відео, описаними нижче.

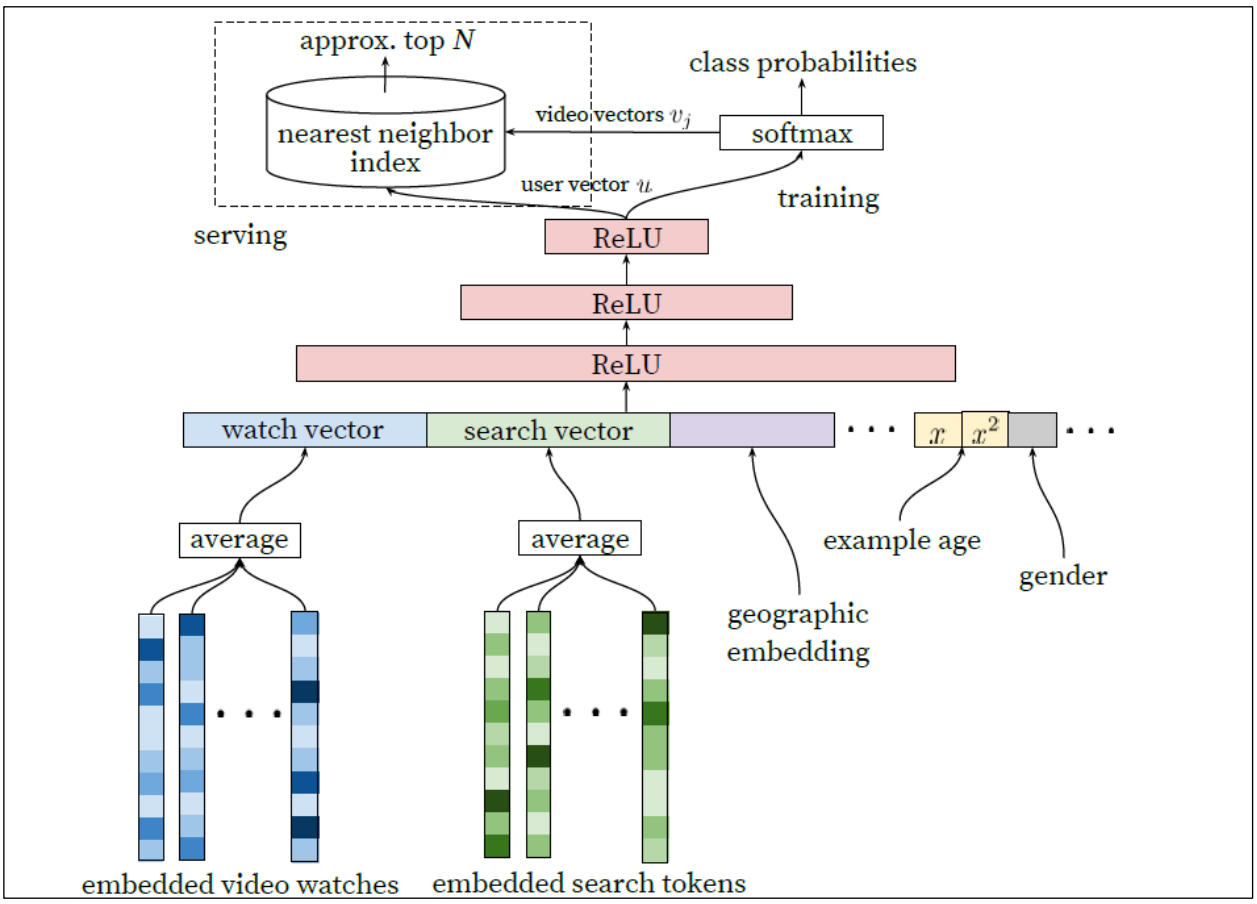

Рисунок 2.3 – Архітектура моделі генерації кандидатів YouTube

На ньому зображена глибока архітектура моделі генерації кандидатів, яка показує вбудовані розріджені функції, об'єднані з щільними функціями. Вбудовування усереднюються перед конкатенацією, щоб перетворити пакети змінного розміру розріджених ідентифікаторів у вектори фіксованої ширини,

придатні для введення в приховані шари. Усі приховані шари повністю з'єднані. Під час навчання втрата перехресної ентропії мінімізована за допомогою градієнтного спуску на виході вибірки softmax. Під час обслуговування виконується приблизний пошук найближчого сусіда для генерації сотень рекомендованих відео-кандидатів.

Efficient Extreme Multiclass (EEML) є важливим алгоритмом в загальному наборі алгоритмів рекомендацій YouTube, де алгоритм повинен передбачити, яке відео користувач перегляне наступним із набору мільйонів можливих відео. Це складне завдання через надзвичайну кількість залучених класів, а також через динамічний характер проблеми рекомендацій. У задачі EEML мета полягає в тому, щоб передбачити наступний перегляд користувача на основі його історії переглядів та інтересів, зберігаючи високу точність і ефективність.

Алгоритм рекомендацій YouTube використовує глибоку нейронну мережеву архітектуру, відому як «модель генерації кандидатів» (CGM), щоб створити короткий список аудіо- та відео-кандидатів для користувача. Модель CGM навчена передбачати ймовірність того, що користувач перегляне кожне відео з великого набору можливих відео. Однак створення прогнозів для мільйонів можливих відео є дорогим з точки зору обчислень і може бути непрактичним у режимі реального часу. Тому потрібен ефективний алгоритм, щоб зменшити кількість відео, які потрібно оцінити за допомогою моделі CGM.

Щоб вирішити цю проблему, алгоритм рекомендацій YouTube використовує двоступеневий підхід, де груба фільтрація застосовується для звуження списку відео перед застосуванням моделі CGM. Етап грубої фільтрації використовує спрощену модель, відому як «генератор грубого кандидата» (CCG), щоб швидко визначити невеликий набір аудіо- та відео-кандидатів, які, ймовірно, зацікавлять користувача. Модель CCG використовує такі функції, як метадані файлу, історія користувача та контекстна інформація для створення списку кандидатів.

Список відео-кандидатів, створений моделлю CCG, потім передається в модель CGM, яка додатково уточнює список до меншого набору відео, які,

швидше за все, зацікавлять користувача. Модель CGM використовує більш складну архітектуру нейронної мережі для оцінки ймовірності того, що користувач перегляне кожне відео-кандидат. Остаточний список рекомендованих відео створюється шляхом ранжування відео на основі ймовірності їх перегляду користувачем.

Загалом проблема Efficient Extreme Multiclass є важливою проблемою в алгоритмі рекомендацій YouTube, а двоетапний підхід, який використовує алгоритм, допомагає збалансувати точність і ефективність. Використовуючи спрощену грубу модель генерації кандидатів для швидкого визначення набору відео-кандидатів перед застосуванням більш складної моделі генерації кандидатів, алгоритм може ефективно рекомендувати відповідний вміст користувачам.

Щоб детальніше пояснити, як проблема Efficient Extreme Multiclass (EEML) вирішується в алгоритмі рекомендацій YouTube, ми можемо глибше зануритися в математику, що лежить в основі двоетапного підходу.

Перший етап підходу передбачає використання грубого генератора кандидатів (CCG) для створення короткого списку кандидатських відео для користувача. Модель CCG навчена передбачати ймовірність того, що користувач перегляне кожне відео, враховуючи його історію переглядів та інші особливості. Це можна сформулювати так:

P(перегляд | відео, історія користувача, контекст)

Ця формула містить наступні дані:

- «перегляд» – це двійкова змінна, яка вказує, чи буде користувач дивитися чи прослуховувати відео/аудіо-матеріал,
- «відео» – відео, яке розглядається,
- «історія користувача» містить інформацію про минулу поведінку користувача під час перегляду,

– «контекст» містить інформацію про поточний контекст користувача, як-от час доби, місцезнаходження та використовуваний пристрій.

Модель CCG можна навчити за допомогою різних методів машинного навчання, таких як логістична регресія або посилення градієнта. Після навчання модель CCG можна використовувати для швидкого визначення невеликого набору відео-кандидатів, які, ймовірно, зацікавлять користувача.

Другий етап підходу передбачає використання моделі генерації кандидатів (CGM) для подальшого уточнення списку відео-кандидатів, створених моделлю CCG. Модель CGM – це складніша нейронна мережа, яка навчена передбачати ймовірність того, що користувач перегляне кожне відео-кандидат. Формулювання цієї моделі те ж саме, що й у CCG:

P(перегляд | відео, історія користувача, контекст)

де «перегляд», «відео», «історія користувача» та «контекст» мають ті самі значення, що й у моделі CCG.

Модель CGM можна навчити за допомогою методів глибокого навчання, таких як згорточні нейронні мережі або рекурентні нейронні мережі. Після навчання модель CGM можна використовувати для створення остаточного списку рекомендованих відео шляхом ранжування відео на основі прогнозованої ймовірності їх перегляду користувачем.

Щоб переконатися, що алгоритм є ефективним і масштабованим, можна використовувати різні методи, такі як стиснення моделі, квантування та розподілене навчання. Методи стиснення моделі спрямовані на зменшення розміру моделей нейронної мережі, що використовуються в алгоритмі, тоді як методи квантування спрямовані на зменшення точності параметрів моделі, щоб зменшити вимоги до пам’яті та обчислень. Методи розподіленого навчання спрямовані на розпаралелювання процесу навчання на кількох машинах, щоб пришвидшити навчання.

Таким чином, проблема Efficient Extreme Multiclass в алгоритмі рекомендацій YouTube вирішується за допомогою двоетапного підходу, який включає легку грубу модель генератора кандидатів і більш складну модель генерації кандидатів. Використовуючи ці моделі разом, алгоритм може збалансувати точність і ефективність, рекомендуючи користувачам релевантний вміст. Математика, що лежить в основі моделей, включає різні методи машинного та глибокого навчання, а також методи оптимізації та масштабованості, щоб гарантувати, що алгоритм може впоратися з надзвичайно масштабною проблемою рекомендацій.

2.4 Неоднорідні сигнали

Ключовою перевагою використання глибоких нейронних мереж як узагальнення матричної факторизації є те, що до моделі можна легко додати довільні неперервні та категоричні характеристики. Історія пошуку обробляється так само, як історія переглядів – кожен запит поділяється на уніграми та біграми, і кожен маркер вбудовано. Після усереднення токенізовані вбудовані запити користувача представляють узагальнену щільну історію пошуку. Демографічні характеристики важливі для надання попередніх даних, щоб рекомендації були прийнятними для нових користувачів. Географічний регіон і пристрій користувача використовуються як вхідні характеристики; вони вбудовані та об’єднані. Прості двійкові та безперервні характеристики, такі як: стать користувача, стан входу в систему та вік, вводяться безпосередньо в мережу як реальні значення, нормалізовані до [0, 1].

Щосекунди на YouTube завантажується безліч аудіо- чи відео-контенту, сумарна тривалість якого сягає декількох годин. Рекомендувати нещодавно завантажений («свіжий») контент надзвичайно важливо для YouTube як продукту. Розробники постійно спостерігають, що користувачі віддають перевагу свіжому контенту, але роблять це не на шкоду релевантності. На додаток до ефекту першого порядку простої рекомендації нових відео, які користувачі хочуть переглянути, існує критичне вторинне явище початкового завантаження та

поширення вірусного контенту [13].

Системи машинного навчання часто виявляють неявну упередженість до минулого, оскільки вони навчені передбачати майбутню поведінку на історичних прикладах. Розподіл популярності роликів являє собою дуже нестаціонарний, але мультиноміальний розподіл за корпусом, створений мережею рекомендацій YouTube, відображає середню ймовірність перегляду впродовж кількох тижнів навчання. Щоб виправити це, вік прикладу навчання подається як функція під час навчання. Під час обслуговування ця функція встановлюється на нуль (або трохи від'ємна), щоб відобразити, що модель робить прогнози в самому кінці вікна навчання.

Рисунок 2.4 демонструє ефективність цього підходу на довільно вибраному відео. Для даного відео модель, навчена за прикладом віку як характеристики, здатна точно відобразити час завантаження та залежну від часу популярність, що спостерігається в даних.

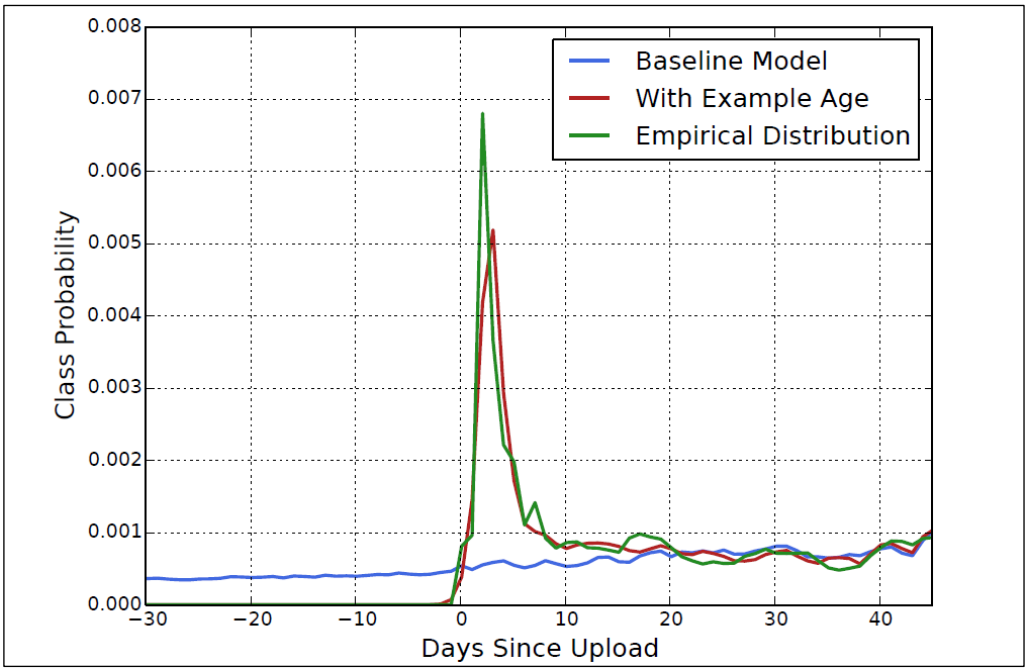

Рисунок 2.4 – Графік ефективності рекомендаційного підходу для YouTube.

Без цієї функції модель передбачила б приблизно середню ймовірність протягом вікна навчання, що зашкодило б релевантності рекомендації.

2.5 Експерименти з функціями та глибиною

Додавання функцій і глибини істотно покращує точність даних, що залишаються, як показано на рисунку 2.5. У цих експериментах словниковий запас із 1 мільйона відео та 1 мільйона маркерів пошуку було вбудовано з 256 oats кожен у максимальний розмір сумки 50 останніх переглядів і 50 останніх пошуків.

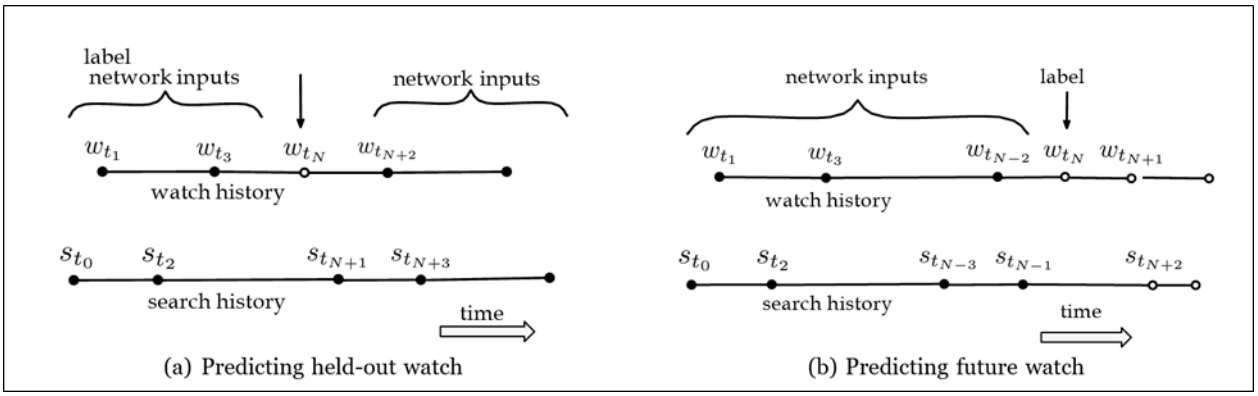

Рисунок 2.5 – Лінійні шари в рекомендаційних алгоритмах Youtube

Шар Softmax виводить мультиноміальний розподіл для того самого 1 мільйона відео-класів з розмірністю 256 кожен (що можна розглядати як окреме вбудовування вихідного відео). Ці моделі тренувалися до конвергенції для всіх користувачів YouTube, що відповідає кільком епохам над даними. Структура мережі дотримувалася загальної схеми «вежі», в якій нижня частина мережі є найширшою, а кожен наступний прихований шар вдвічі зменшує кількість одиниць (схоже на рис. 3). Мережа з нульовою глибиною фактично є лінійною схемою факторизації, яка працює дуже подібно до системи-попередника. Ширину та глибину додавали, доки додаткова вигода не зменшилася, а конвергенція стала складною:

- глибина 0: лінійний шар просто перетворює шар конкатенації, щоб відповідати розміру softmax 256;
- глибина 1: 256 ReLU;
- глибина 2: 512 ReLU → 256 ReLU;
- глибина 3: 1024 ReLU → 512 ReLU → 256 ReLU;

– глибина 4: 2048 ReLU → 1024 ReLU → 512 ReLU → 256 ReLU.

Вибір міток і вхідного контексту для моделі важко оцінити в автономному режимі, але це має великий вплив на живе виконання. На рисунку 2.5 суцільні події є вхідними функціями в мережу, тоді як порожні події виключаються.

Було виявлено, що прогнозування майбутнього перегляду (2.5b) показало кращі результати в A/B-тестуванні. У (2.55b) приклад age виражається як $t_{max} - t_N$, де t_{max} є максимальним спостережуваним часом у даних навчання.

2.6 Висновки щодо мережи рекомендацій YouTube

Було описано глибоку нейронну мережеву архітектуру для рекомендацій аудіо на хостингах YouTube, з розділенням її на дві різні проблеми: створення кандидатів і рейтинг.

Було виявлено, що модель глибокої спільної фільтрації YouTube здатна ефективно асимілювати багато сигналів і моделювати їх взаємодію з шарами глибини, перевершуючи попередні підходи матричної факторизації, які використовувалися на YouTube. У виборі сурогатної проблеми для рекомендацій є більше мистецтва, ніж науки, і ми виявили, що класифікація майбутнього перегляду має хороші показники в реальному часі, фіксуючи асиметричну поведінку спільного перегляду.

Було продемонстровано, що використання віку навчального прикладу як вхідної функції усуває притаманну упередженість до минулого та дозволяє моделі представляти залежну від часу поведінку популярних аудіо та відео. Це покращило результати офлайн-затримки та значно збільшило час перегляду нещодавно завантажених файлів під час A/B-тестування.

Ранжування є більш класичною проблемою машинного навчання, але наш підхід глибокого навчання перевершив попередні лінійні та деревоподібні методи прогнозування часу перегляду. Рекомендаційні системи особливо виграють від спеціалізованих функцій, що описують минулу поведінку користувача з елементами. Глибокі нейронні мережі вимагають спеціальних представлень категоріальних і неперервних ознак, які ми перетворюємо за допомогою

вбудовування та квантильної нормалізації відповідно. Показано, що шари глибини ефективно моделюють нелінійні взаємодії між сотнями характеристик.

Логістичну регресію було змінено шляхом зважування навчальних прикладів із часом перегляду для позитивних прикладів і одиницею для негативних прикладів, що дозволило нам дізнатися коефіцієнти, які точно моделюють очікуваний час перегляду. Цей підхід показав набагато кращі показники рейтингу, зваженого за часом перегляду, порівняно з прямим прогнозуванням CTR.

2.7 Загальна характеристика рекомендацій Spotify

Тепер перейдемо до рекомендаційних алгоритмів Spotify. В їх основі лежить використання різних технік та джерел інформації для того, щоб пропонувати користувачам музику, яка їм сподобається. Основні компоненти алгоритмів Spotify включають:

- співпраця фільтрація (Collaborative Filtering): Співпраця фільтрація базується на історіях прослуховування інших користувачів. Якщо користувач А слухає схожу музику на користувача В, алгоритм припустить, що вони мають схожі музичні смаки, і рекомендуватиме користувачу А пісні, які сподобалися користувачеві В, але які А ще не слухав;
- аналіз контенту (Content-based Analysis): Аналіз контенту використовується для виявлення характеристик самих пісень, таких як жанр, темп, тональність, інструменти тощо. Алгоритм порівнює ці характеристики з піснями, які користувач вже слухав, і пропонує музику зі схожими атрибутами;
- натуральна мова (Natural Language Processing, NLP): Spotify також використовує аналіз натуральної мови для розуміння текстів пісень, рецензій, блогів та інших текстових джерел. Це допомагає алгоритму виявляти музичні тренди, а також рекомендувати пісні на основі смаків користувача в текстах пісень;

- рекомендації на основі популярності та новизни: Spotify також робить акцент на популярність і новизну пісень. Вони можуть рекомендувати пісні, які стали популярними за останній час або відносяться до актуальних музичних трендів.

Усі ці методи та джерела інформації взаємодіють, щоб створювати персоналізовані рекомендації для користувачів Spotify. Алгоритми постійно вдосконалюються та оптимізуються, забезпечуючи кращі рекомендації з часом.

Ось деякі особливості та можливості Spotify, які базуються на рекомендаційних алгоритмах:

- Discover Weekly: Цей плейлист пропонує користувачам нові пісні щотижня на основі їх музичних смаків та прослуховувань. Він формується завдяки колаборативній фільтрації, аналізу контенту та натуральної мови;
- Daily Mix: Daily Mix - це серія плейлистів, які створюються на основі музичних смаків користувача. Вони містять пісні, які користувач вже слухав, а також нові рекомендації, що засновані на схожих жанрах або артистах;
- Radio: Функція "Радіо" пропонує потік пісень, які підібрані на основі конкретного артиста, альбому чи пісні. Це допомагає користувачам відкривати нову музику, схожу на ту, яка їм вже подобається;
- збережені пісні та плейлисти: Система рекомендацій Spotify також аналізує пісні та плейлисти, які користувачі зберігають у своїх бібліотеках, для створення більш точних рекомендацій.

Загалом, у основі будь-якої системи рекомендацій штучного інтелекту лежить модель ML, оптимізована для ключових бізнес-цілей: утримання користувачів, час, проведений на платформі, і, зрештою, отриманий дохід. Щоб ця система рекомендацій працювала, вона має розуміти вміст, який вона рекомендує, і користувачів, яким вона його рекомендує. З обох сторін цієї пропозиції Spotify використовує кілька незалежних моделей і алгоритмів машинного навчання для створення представлень елементів і представлень

користувачів.

2.8 Математична характеристика. BaRT (“Bandits for Recommendations as Treatments”)

Система налаштування Spotify називається BaRT («Бандити за рекомендації як лікування»). Модель BaRT має два режими: експлуатація та дослідження.

Експлуатація використовується, коли система використовує зібрану інформацію про вас (користувача), наприклад, ваші пропуски, ваші улюблені пісні тощо. Натомість дослідження використовується, коли система пропонує вам пісні на основі всієї іншої інформації, система може використовувати, наприклад, дані про треки, які слухали інші користувачі, які плейлисти були створені, які композиції зараз «в тренді» тощо.

Режим експлуатації – це звичайний режим, який використовують системи рекомендацій, засновані на спільному фільтруванні. У випадку, якщо існує достатньо історичних даних про користувачів і те, що вони слухали, цей режим ідеальний. У режимі експлуатації використовуються всі доступні дані про користувача та предмет, напр. пропущені пісні, як часто пісня відтворювалася, спільні пісні, списки відтворення тощо. Однією з головних проблем систем, заснованих лише на експлуатації, є релевантність елемента. Якщо немає даних про користувача чи елемент (у цьому випадку певну пісню), напр. пісня не відтворювалася часто, тоді система не впевнена, рекомендувати (використовувати) чи не рекомендувати (ігнорувати). На рисунку 2.6 зображена спрощена схема роботи моделі BaRT.

Саме в цей момент починає діяти алгоритм Bandit і другий режим виходить на перший план: дослідження. Оскільки режим експлуатації погано працює, коли є невизначеність (недостатньо даних), режим дослідження охоплює саме ці випадки. «Exploration рекомендує контент із невизначеним прогнозованим залученням користувачів з метою збору додаткової інформації. Важливість дослідження була визнана в останні роки, особливо в налаштуваннях з новими

користувачами, новими предметами, нестаціонарними налаштуваннями та атрибутами».

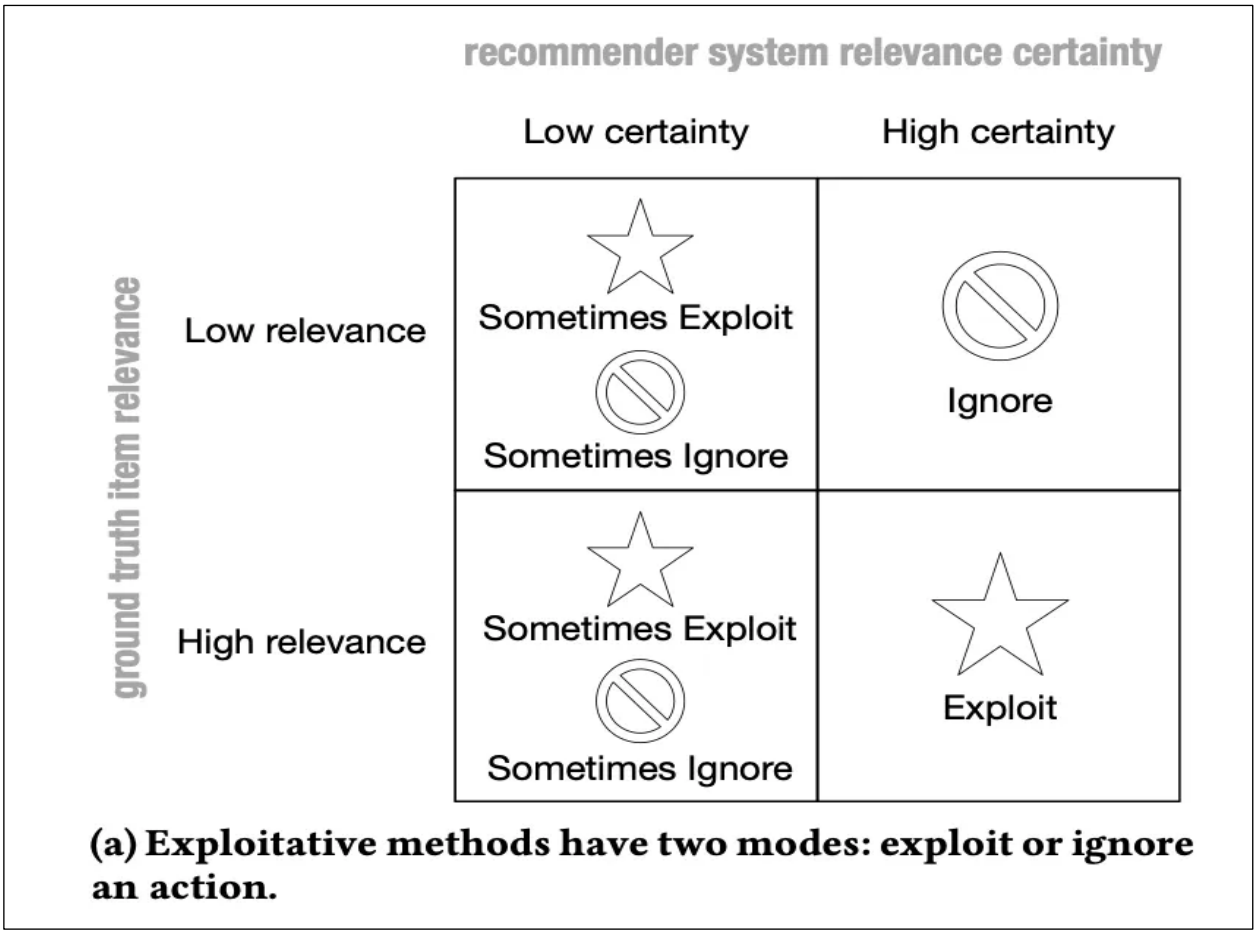

Рисунок 2.6 – схема роботи режиму експлуатації

Іншими словами, якщо нова пісня ще не програвалася часто, системі необхідні дані, щоб перевірити, чи має пісня потенціал. Отже, BaRT Spotify рекомендує ці пісні за допомогою дослідження та збирає інформацію для нової пісні. Пісня вважається позитивною рекомендацією через 30 секунд. Це означає, що якщо користувач слухає пісню менше півхвилини, вона вважається негативною. Якщо користувач слухає її більше 30 секунд, він отримує позитивний відгук за рекомендацію.

Як видно на рисунку 2.7, коли існує низька впевненість щодо актуальності пісні, режим дослідження необхідний і корисніший, ніж режим експлуатації. Модель BaRT здатна «вивчати та передбачати задоволення», яке вимірюється, наприклад, у показниках кліків і ймовірності споживання. Він постійно

реєструється, перенавчається та вчиться на власних помилках.

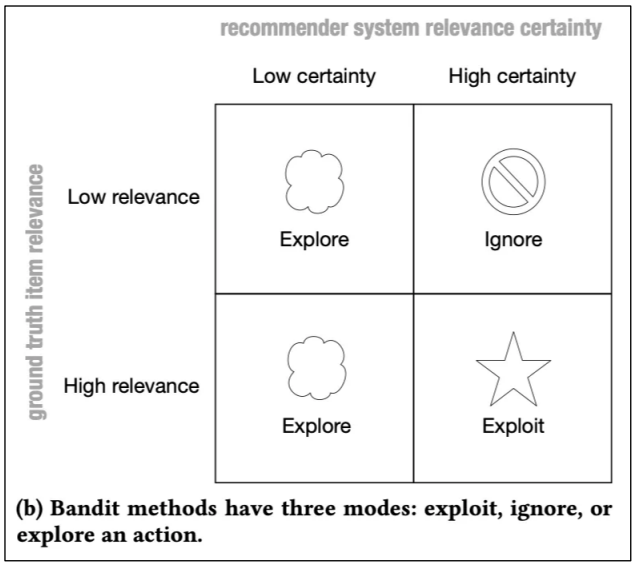

Рисунок 2.7 – схема роботи режиму BaRT

Іншими словами, BaRT базується на навчанні з підкріпленням і намагається отримати зворотній зв’язок, щоб максимально задовольнити користувачів і виправити прогнозовані рекомендації. Для цього в моделі BaRT використовується «багаторукий бандит», який навчений виконувати певну дію A, за яку найвища ймовірність отримати винагороду R. Таким чином, кожна дія A залежить від попередніх дій і винагороди. «Мета багаторукого бандита (MAB) – вибрати дії, які максимізують загальну суму винагород». Звичайний MAB повністю ігнорує їхній контекст, як-от час доби, пристрій, плейлист, функції користувача тощо. Саме тому командою розробників був представлений контекстний багаторукий бандит. Він стежить за контекстною інформацією та оцінює цю інформацію, перш ніж вирішить виконати дію.

Для ефективної контекстуального бандита є вирішальними чотири елементи: контекст, модель винагороди, процедура навчання та політика експлуатації та дослідження. У моделі винагороди існує три параметри: пояснення e (пояснення, чому було обрано предмет), елемент j (пісня) і контекст x. На рисунку 2.8 зображена формула моделі винагород, де θ позначають коефіцієнти логістичної регресії, і 1_i «представляє один гарячий вектор нулів з єдиною 1 в індексі».

$$r(j, e, x) = \sigma(\theta_{\text{global}} + \theta_j^\top 1_j + \theta_e^\top 1_e + \theta_x^\top x),$$

Рисунок 2.8 – формула моделі винагород

Тепер ця функція винагороди впливає на дію. У певному контексті x користувач u виконує оптимальну дію, яку можна зобразити формулою, яка зображена на рисунку 2.9.

$$(j^*, e^*) = \arg_{j,e} \max \bar{r}(j, e, x)$$

Рисунок 2.9 – формула виконання оптимальної дії користувачем у певному контексті

Для дослідницького підходу автори використовують epsilon-greedy. Це «дає однакову масу ймовірності всім неоптимальним елементам у наборі дійсності f (e,x) і (1−ε) додаткової маси до оптимальної дії (j∗,e∗)». Політика встановлена на «або використовувати, або досліджувати елемент і пояснення одночасно». Формула цього підходу зображена на рисунку 2.10.

$$\pi_c^{\text{item}}(j \mid x, e) = \begin{cases} (1-\epsilon) + \frac{\epsilon}{|f(e,u)|}, & \text{if } j = j^*, j \in f(e,x) \\ \frac{\epsilon}{|f(e,u)|}, & \text{if } j \neq j^*, j \in f(e,x) \\ 0, & \text{otherwise.} \end{cases}$$

$$\text{where } j^* = \arg_{j_1} \max r(j_1, e, x)$$

Рисунок 2.10 – формула epsilon-greedy

Сама процедура навчання виконується наступним чином.. BaRT «періодично перенавчається в пакетному режимі (batch mode). Спільна фільтрація добре працює для користувачів і музики, які вже завантажили пісню або

прослухали її.

2.9 Створення представлень доріжок: фільтрація на основі вмісту та спільна фільтрація

Підхід Spotify до представлення треків складається з двох основних компонентів: фільтрації на основі вмісту, спрямованої на опис треку шляхом вивчення самого вмісту, та спільної фільтрації, спрямованої на опис доріжки в її зв’язку з іншими доріжками на платформі шляхом вивчення ресурсів, створених користувачами.

Система рекомендацій потребує даних, згенерованих обома методами, щоб отримати цілісне уявлення про вміст платформи та вирішити проблеми холодного запуску під час роботи з щойно завантаженими треками. По-перше, давайте подивимося на алгоритми фільтрації на основі вмісту.

1) Аналіз метаданих художника.

Щойно на платформу Spotify завантажується новий трек, алгоритм проаналізує всі загальні метадані пісні, надані розповсюджувачем, і метадані, специфічні для Spotify (отримані через форму презентації Spotify для виконавців). В ідеальному сценарії, коли всі метадані заповнено правильно та потрапляє до бази даних Spotify, цей список має містити:

- назва треку;
- назва випуску;
- ім'я виконавця;
- вибрані артисти;
- автор пісень;
- продюсери виконавця;
- лейбл;
- дата випуску;
- теги жанру та піджанру;
- теги музичної культури;

- теги настрою;
- теги стилю;
- головна мова;
- інструменти, що використовуються під час запису;
- типологія треку (це кавер? Це ремікс? Це інструментал?);
- рідне місто/місцевий ринок художника.

Метадані, отримані від виконавця, потім передаються вниз за течією як вхідні дані в інші моделі на основі вмісту та саму систему рекомендацій.

2) Аналіз необроблених аудіосигналів

Другим кроком фільтрації на основі вмісту є аналіз необробленого аудіо, який запускається, щойно аудіофайли разом із метаданими, які були занесені виконавцем, потрапляють до бази даних Spotify. Точний спосіб, у який цей аналіз виконується, залишається одним із секретів системи рекомендацій Spotify. Проте ось що ми знаємо напевно – і що ми можемо розумно припустити:

Почнемо з конкретних фактів. Дані аудіофункцій, доступні через Spotify API, складаються з 12 показників [14], що описують звукові характеристики треку. Більшість цих особливостей пов’язані з об’єктивними звуковими описами. Наприклад, метрика «інструментальності» відображає впевненість алгоритму в тому, що трек не містить вокалу, оцінюється за шкалою від 0 до 1. Однак, крім цих «об’єктивних» аудіоатрибутів, Spotify генерує принаймні три сприйнятливі високо-рівневі функції, призначені для більш цілісного відображення звучання треку:

- танцювальність (Danceability), що описує, наскільки трек придатний для танцю на основі поєднання музичних елементів, включаючи темп, стабільність ритму, силу ритму та загальну регулярність;
- енергійність (Energy), що представляє «перцептивну (відчуттєву) міру інтенсивності та активності», засновану на динамічному діапазоні треку, сприйнятій гучності, тембрі, швидкості початку та загальної ентропії;
- валентність (Valence), описуючи «музичну позитивність треку». Загалом, композиції з високою валентністю звучать більш позитивно (наприклад,

щасливі, веселі, ейфоричні), тоді як пісні з низькою валентністю звучать більш негативно (наприклад, сумні, пригнічені, злі).

Проте ці аудіофункції є лише першим компонентом системи аналізу аудіо Spotify. Окрім виділення аудіофункцій, окремий алгоритм також аналізуватиме часову структуру треку та розділятиме аудіо на різні сегменти різної деталізації: від розділів (визначених значними змінами в тембрі чи ритмі пісні, які підкреслюють переходи між ключовими частинами трек, такий як куплет, приспів, бридж, соло тощо) аж до татумів (що представляють найменший когнітивно значущий підрозділ основного ритму). Поєднання даних, отриманих за допомогою методів аудіоаналізу, має дозволити Spotify розпізнавати звукові характеристики пісні та стежити за їх розвитком упродовж часу та між різними частинами треку. Ці 12 функцій датуються 2013 роком, коли вони були лише частина аудіоаналізу The Echo Nest (компанія аудіорозвідки, придбана Spotify ще в 2014 році) [15]

Цілком імовірно, що за майже десять років алгоритми аналізу аудіо Spotify пройшли довгий шлях – це означало б, що дані про аудіофункції, які сьогодні надходять у систему рекомендацій, є набагато детальнішими та детальнішими, ніж ті, які доступні через загальнодоступний API компанії. Наприклад, в одній із дослідницьких статей, опублікованих Spotify у 2021 році, стверджується, що аудіофункції передаються в модель як 42-вимірний вектор – це може означати, що аудіоаналіз Spotify створює 42 різні аудіофункції [16]

Крім того, велика кількість дослідницьких записів компанії детально описує експерименти Spotify із розділенням джерел на основі ML, відстеженням висоти та оцінкою мелодії. Якби ці проекти були запущені у виробництво, це означало б, що система аудіоаналізу зможе розділити трек на окремі інструментальні партії, обробити їх окремо та визначити всі мелодії та прогресії акордів, які використовуються в композиції.

На практиці все вищесказане означало б, що аналіз звуку Spotify може дуже детально визначати записи, завантажені на платформу. Остаточний результат системи може визначати трек за такими лініями: «ця пісня має структуру V-C-V-

C-B-V-C, накопичує енергію до мосту (builds up in energy towards the bridge) та містить агресивне, дисонансне гітарне соло, яке перетворюється на більш меланхолійне та спокійне аутро». Або навіть щось набагато більш детальне – справа в тому, що, ймовірно, алгоритм аудіоаналізу Spotify може майже повністю реконструювати трек і витягти найширший спектр характеристик із необроблених аудіофайлів. З точки зору художника, було б розумно припустити, що Spotify має доступ до файлів проекту DAW.

2.10 Аналіз тексту за допомогою моделей обробки природної мови

Останнім компонентом представлення доріжки на основі вмісту є моделі обробки природної мови (Natural Language Processing, або скорочено NLP), які використовуються для отримання семантичної інформації, що описує доріжку/виконавця, із текстового вмісту, пов’язаного з музикою. Ці моделі застосовуються в трьох основних контекстах:

- аналіз текстів пісень. Основною метою тут є встановлення основних тем і загального значення тексту пісні, а також пошук потенційних «підказок», які можуть бути корисними в подальшому, таких як місця розташування, бренди або люди, згадані в тексті;
- дані, скановані Інтернетом, або Web-crawled data. В даному випадку насамперед мова йде про музичні блоги та онлайн-медіа. Запуск моделей NLP із даними, сканованими в Інтернеті, дає змогу Spotify виявити, як люди (і гейткіпери) описують музику в Інтернеті, аналізуючи терміни та прикметники, які найчастіше зустрічаються з назвою пісні чи іменем виконавця;
- плейлисти, створені користувачами. Алгоритми NLP працюють зі створеними користувачами плейлистами, що містять трек на Spotify, щоб отримати додаткову інформацію про настрій, стиль і жанр пісні. «Якщо пісня з’являється у багатьох списках відтворення зі згадуванням у назві слова «сумний», це сумна пісня».

NLP-моделі дозволяють Spotify підключитися до культурного контексту треку та розширити звуковий аналіз того, як звучить пісня, за допомогою соціального виміру того, як пісня сприймається. Три описані вище компоненти – метадані, отримані від виконавців, аудіоаналіз і моделі NLP – складають контентний підхід до репрезентації треків у системі рекомендацій Spotify. Проте є ще один ключовий інструмент із загального набору Spotify для представлення треків.

2.11 Спільна фільтрація

Багато в чому спільна фільтрація стала синонімом системи рекомендацій Spotify. Компанія DSP (посилання на головну сторінку компанії: https://www.dspg.com/) стала піонером у застосуванні цього так званого «підходу Netflix» у контексті музичних рекомендацій – і широко оприлюднила спільну фільтрацію як рушійну силу свого механізму рекомендацій:

«Ми можемо зрозуміти, які пісні рекомендувати користувачеві, дивлячись на те, що слухають інші користувачі з подібними смаками». Алгоритм просто порівнює історію прослуховування користувачів: якщо користувачеві A сподобалися пісні X, Y і Z, а користувачеві B сподобалися пісні X і Y (але він ще не чув Z), ми повинні рекомендувати їм пісню Z. Підтримуючи масивну матрицю взаємодії між користувачами та елементами, що охоплює всіх користувачів і треки на платформі, Spotify може визначити, чи схожі дві пісні (якщо їх слухають схожі користувачі) і чи схожі два користувачі (якщо вони слухають однакові пісні).

Цей підхід матриці «елемент-користувач» пов’язаний із безліччю проблем, пов’язаних із проблемами точності, масштабованості, швидкості та холодного запуску. Таким чином, в останні роки Spotify відійшов від фільтрації на основі споживання – або принаймні різко применшив свою роль у представленні треків. Натомість поточна версія спільного фільтрування зосереджується на організаційній схожості треку: тобто «дві пісні схожі, якщо користувач розміщує їх в одному плейлисті».

Вивчаючи спільну появу плейлисту та сеансу прослуховування, алгоритми спільної фільтрації отримують більш глибокий рівень деталізації та фіксують чітко визначені сигнали користувача. Простіше кажучи, користувачі даного стрімінгового сервісу часто мають досить широкі та різноманітні профілі прослуховування (тобто, слухають музику різних жанрів) – саме тому той факт, що багато користувачів слухають пісню А та пісню Б не означає автоматично, що ці два виконавці схожі. Зрештою, такі виконавці, як Metallica та ABBA, мають чимало спільних слухачів.

Якщо, з іншого боку, багато користувачів розміщують пісню А та пісню Б до одного й того ж самого плейлиста, це є набагато переконливішою ознакою того, що ці дві пісні мають щось спільне. Крім того, підхід, в якому плейлисти використовуються як основний елемент, також пропонує зрозуміти контекст, у якому ці дві пісні схожі – і оскільки створення плейлистів є однією з найпоширеніших практик на платформі, Spotify не має браку в даних спільної фільтрації для роботи.

Сьогодні модель спільного фільтрування Spotify тренується на вибірці з ~700 мільйонів створених користувачами плейлистів, вибраних із значно ширшого набору всіх створених користувачами плейлистів на платформі. Основний принцип вибору плейлистів для тренування нейронної мережі: «Пристрасть, турбота, любов і час, які користувачі вкладають у створення цих списків відтворення». [17].

Тепер необхідно детально розглянути момент, коли поєднання підходів, що базуються на контенті та взаємодії одних треків з іншими, дозволяє системі рекомендацій Spotify створити цілісне представлення треку. На цьому етапі профіль доріжки додатково збагачується шляхом поєднання результатів кількох незалежних алгоритмів для генерації векторів вищого рівня (для спрощення розглядатимемо їх як теги настрою, жанру, стилю тощо). Крім того, для вирішення проблеми холодного запуску під час обробки щойно завантажених випусків, які не мають достатньо даних NLP/плейлистів, деякі з цих властивостей

також екстраполюються для розробки загальних алгоритмічних профілів виконавців.

Однак, щоб перетворити ці дані про композиції та виконавця на релевантні рекомендації, механізму потрібно поєднати їх із даними, що описують користувачів.

2.12 Створення профілів смаків користувачів

Підхід до профілювання користувачів на Spotify має трохи простіший принцип роботи, принаймні коли ми вирішуємо представлення треків. По суті, механізм рекомендацій реєструє всю активність користувача на прослуховування, розбиваючи її на окремі сеанси прослуховування з багатим контекстом. Цей контекстний компонент є життєво важливим при інтерпретації активності користувача для створення профілів смаку. Наприклад, якщо користувач переходить у вкладку «Що нового» Spotify, основною метою сеансу прослуховування часто є швидке вивчення музики, нещодавно доданої на платформу. У цьому контексті слід очікувати високі показники пропуску, оскільки основна мета користувача – швидко переглянути канал і зберегти частину треків для подальшого прослуховування, а це означає, що пропуск треку не слід сприймати як певний негативний сигнал. З іншого боку, якщо користувач пропускає трек під час прослуховування плейлиста «Deep Focus», призначеного для програвання у фоновому режимі, цей пропуск є набагато сильнішою ознакою невдоволення користувача.

Загалом відгуки користувачів можна розділити на дві основні категорії:

- явний або активний відгук: збереження в бібліотеці, додавання до плейлиста, шерінг (sharing) посилання на трек зі знайомими у месенджерах, пропуск, перехід на сторінку виконавця/альбому, стеження за виконавцем, відтворення "за потоком";
- неявний або пасивний зворотний зв’язок: тривалість сеансів прослуховування, відтворення доріжки та повторне прослуховування.

У випадку системи рекомендацій Spotify явний зворотній зв’язок має найбільше значення при розробці профілів користувачів. Музикою часто користуються як позакадровим вмістом (її прослуховують у фоновому режимі, без постійної взаємодії з графічним інтерфейсом додатку), а це означає, що безперервне споживання не завжди пов’язане з задоволенням. Потім дані відгуків користувачів обробляються для розробки профілю користувача, визначеного за допомогою наступних показників:

- найпопулярніші пісні та виконавці;
- збережені пісні та альбоми та виконавці, на які ви підписалися;
- уподобання жанру, настрою, стилю та епохи;
- популярність і різноманітність переваг;
- скроневі візерунки;
- демографічний і геолокаційний профіль.

Потім профіль смаків користувача додатково поділяється на основі контексту споживання: тобто той самий користувач може надавати перевагу м’якому інді-попу в неділю ввечері та енергійному мотиваційному хіп-хопу вранці в понеділок. На рисунку 2.11 зображений контекстно-залежний профіль користувача, який Spotify отримує в результаті обробки даних [18].

Цей профіль користувача на основі його історії постійно розвивається та розширюється завдяки свіжим даним споживання та взаємодії. Остання активність користувача також має пріоритет над історичним профілем: наприклад, якщо користувач потрапляє в новий жанр, який отримує хороші оцінки з точки зору відгуків інших користувачів, система рекомендацій намагатиметься обслуговувати більше суміжної з цим жанром музики – навіть якщо улюблена музика користувача за весь час сильно відрізняється від даного жанру.

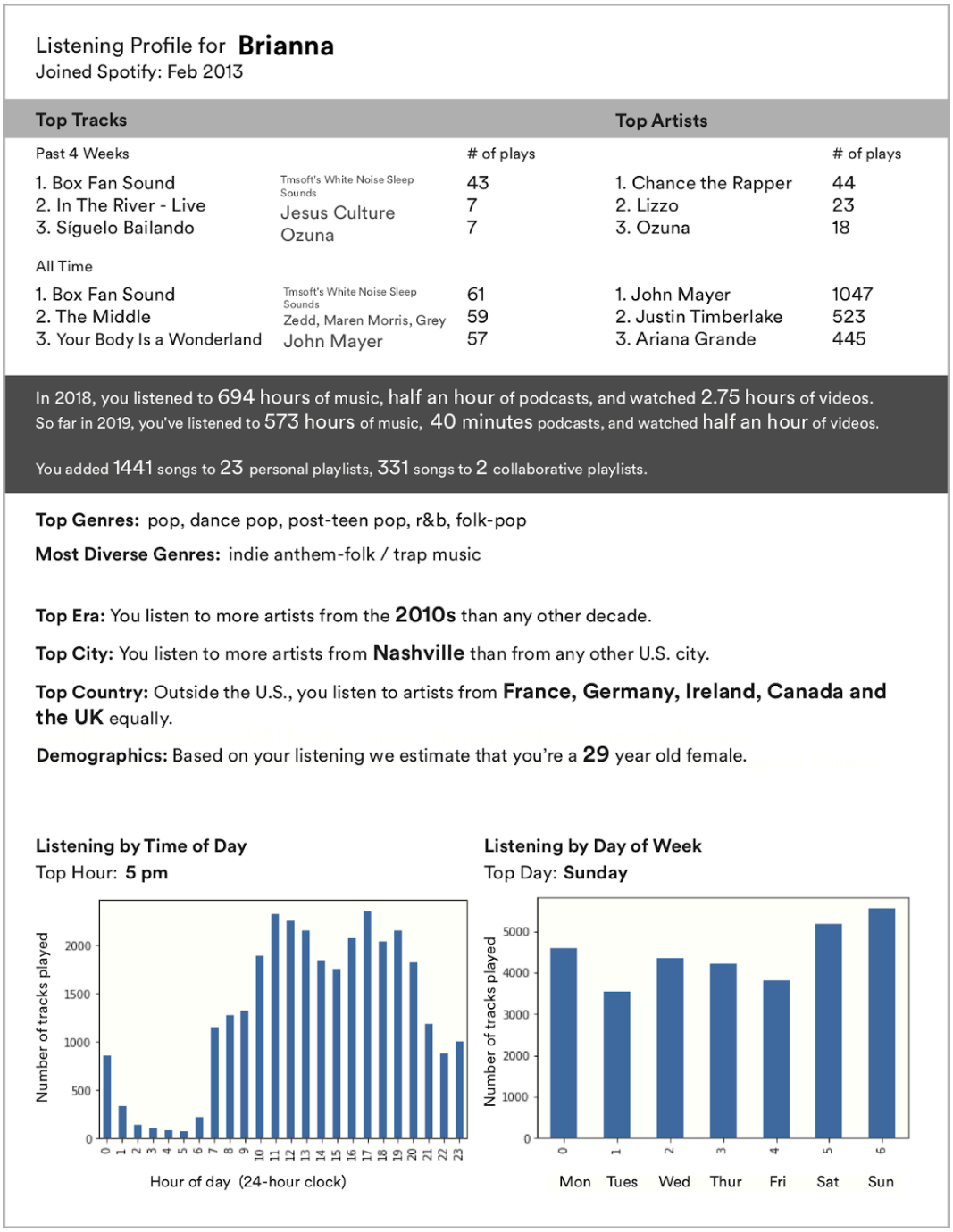

Рисунок 2.11 –Контекстно-залежний профіль користувача Spotify

2.13 Рекомендована музика: інтеграція представлень користувача та доріжки

Переплетений набір алгоритмів, що стоять за рекомендаціями Spotify, створив два основні компоненти – доріжку та плейлисти – необхідними для обслуговування відповідної музики. Тепер необхідно реалізувати алгоритм, який

матиме змогу поєднати ці компоненти і знайти правильний трек для користувача в певний момент часу.

Однак ландшафт рекомендацій на Spotify набагато різноманітніший, ніж на стрімінгових платформах-конкурентах. Розглянемо спектр функцій Spotify, створених за допомогою системи рекомендацій:

- плейлисти Weekly & Release Radar;
- плейлисти Daily Mix;
- виконавець/Епоха/Настрій/Списки жанрів;
- спеціальні персоналізовані списки відтворення (Your Time Capsule, On Repeat, Repeat Rewind тощо);
- персоналізовані редакційні плейлисти;
- персоналізований розділ огляду;
- персоналізовані результати пошуку;
- пропозиції плейлистів;
- радіо виконавця/пісні та функції автовідтворення.

Всі вищезазначені функції опосередковуються механізмом рекомендацій, але кожна з них працює за окремим алгоритмом зі своєю внутрішньою логікою та системою винагород. [19] Репрезентація треку та користувача формує щось на зразок універсальної основи для цих алгоритмів, забезпечуючи спільний рівень моделі, призначений для відповідей на загальні запитання алгоритмів, що стосуються певних функцій, наприклад:

- спорідненість між користувачем і артистом/треком/іншою сутністю Spotify: «Наскільки користувачу X подобається виконавець A чи трек B? Які улюблені виконавці/треки користувача Y?»;
- подібність елемента: «Наскільки схожі виконавець A і виконавець B? Які 10 композицій найбільше схожі на композицію C?»;
- кластеризація елементів: «Як би ми розділили ці 50 треків/виконавців на окремі групи?

На рисунку 2.12 зображена «хмара схожості» (Cloud of similarities) різних музичних груп в тому вигляді, в якому її бачить рекомендаційний механізм Spotify:

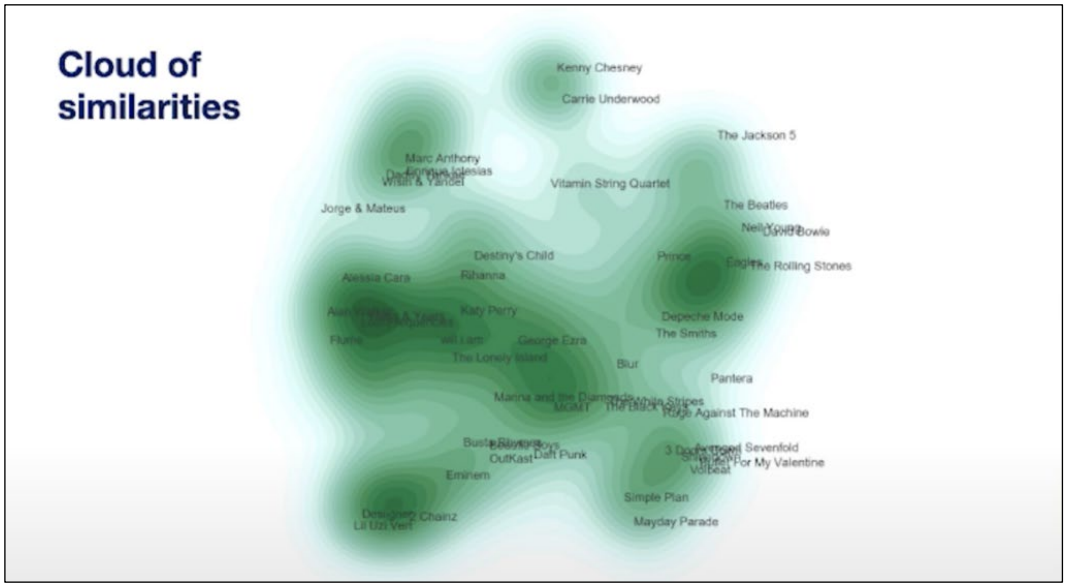

Рисунок 2.12 – «Хмара схожості» певних музичних груп Spotify.

Спеціальні алгоритми можуть використовувати ці уніфіковані моделі для створення рекомендацій, оптимізованих для певного простору/контексту споживання. Наприклад, алгоритм створення плейлисту Your Time Capsule в першу чергу буде взаємодіяти з даними про приналежність користувача до об’єкта (треку, що прослуховується), щоб спробувати знайти треки, які користувачі люблять, але які давно не слухали. З іншого боку, алгоритми, що підбирають треки для плейлисту Discover Weekly використовують суміш даних про спорідненість і подібність, щоб знайти треки, схожі на вподобання користувача, про які він ще не чув. Нарешті, створення плейлисту Your Daily Mix включатиме всі три методи – спочатку кластеризацію уподобань користувача в кілька груп, а потім розширення цих списків подібними треками.

2.14 Цілі та винагороди алгоритмів рекомендацій Spotify

Як було згадано раніше, основна мета системи рекомендацій Spotify пов’язана насамперед з утриманням користувача на Spotify, яке вимірюється часом, проведеним ним на платформі, і загальним задоволенням користувачів.

Однак ці цілі найвищого рівня надто широкі, щоб розробити збалансовану систему винагороди для алгоритмів ML, які обслуговують рекомендації вмісту в різних функціях і контекстах, тому визначення успіху алгоритмів значною мірою залежить від того, де і чому користувач залучається з системою.

Наприклад, успіх функцій автоматичного програвання треків у «черзі» визначається в основному з точки зору залучення користувачів – явний/прихований відгук про частоту прослуховування та пропуску, збереження бібліотеки та списку відтворення, перехід до профілю виконавця та/або альбому, кількість посилань на трек з черги шляхом «шерінга», і так далі. [20] Однак у випадку плейлисту Release Radar набір винагород буде значно іншим, оскільки користувачі часто переглядають цей плейлист частково, а не слухають його від початку до кінця. Таким чином, замість того, щоб вивчати взаємодію з контентом, алгоритми оптимізують довгострокове збереження функцій і поведінку, пов’язану з ними. «Користувачі задоволені цією функцією, якщо повертаються до неї щотижня; користувачі задоволені Release Radar, якщо зберігають треки з цього плейлиста у своїх власних плейлистах або бібліотеках».

Нарешті, у деяких випадках Spotify використовує ще один набір алгоритмів, щоб створити функції винагороди за певну функцію. Наприклад, Spotify навчив окрему модель ML для прогнозування задоволеності користувачів плейлистом Discover Weekly (з набором тренінгів, отриманим за допомогою опитувань користувачів). Ця модель розглядає всі дані про взаємодію з користувачем, минулу поведінку користувача в Discover Weekly та кластери цілей користувача (тобто, якщо користувач використовував Discover Weekly як фон, для пошуку нової музики, збереження музики на майбутнє тощо) – а потім створює уніфікований показник задоволеності на основі всієї цієї діяльності. На рисунку 2.13 зображена спрощена схема роботи даної моделі, на якій видно, якими даними послуговується «дерево рішень» рекомендувальної мережі Spotify.

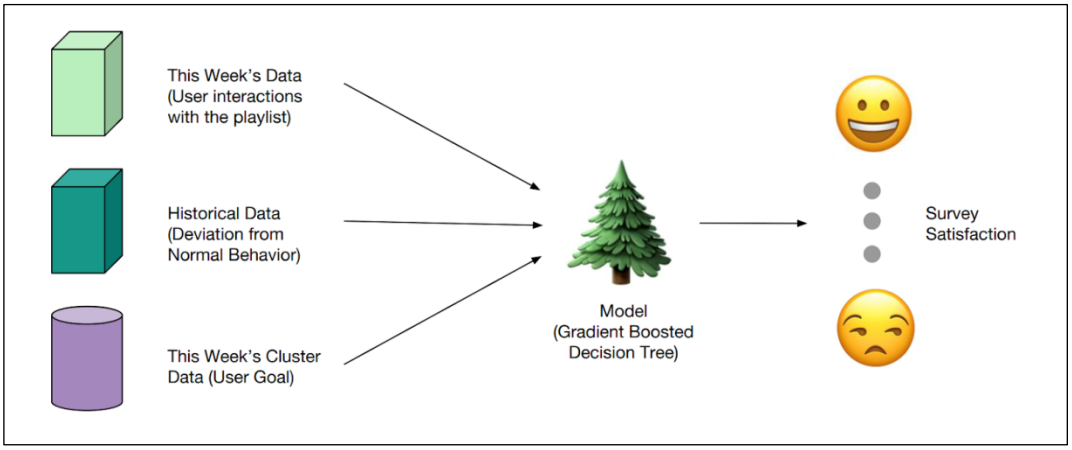

Рисунок 2.13 – Модель навчання «дерева рішень» на основі історії користувача

Прогноз задоволення, створений моделлю, потім, у свою чергу, використовується як винагорода для алгоритму, який складатиме списки відтворення Discover Weekly, створюючи таким чином комплексну систему винагороди, яка не покладається на ізольовані, потенційно неоднозначні сигнали користувачів. [21]

Система рекомендацій Spotify – це надзвичайно складна та заплутана система з десятками (якщо не сотнями) алгоритмів і моделей ML, які використовуються на різних рівнях, і всі вони працюють разом, щоб створити одну з найдосконаліших рекомендацій на ринку потокового передавання музики. Цю систему розробляли та використовували вже майже 12 років, зростаючи в розмірах, можливостях і складності. Проте, як ви, мабуть, бачите, це далеко не непояснимо. Навіть не маючи прямої документації, що описує склад механізму рекомендацій і всі секретні інгредієнти, ми можемо отримати досить добре розуміння його основних частин і керівних принципів, що стоять за ними.

З точки зору музичної індустрії, ці знання можна використовувати для оптимізації профілю виконавця в системі рекомендацій. Змістовна, продумана алгоритмічна стратегія може максимізувати ваші шанси потрапити до алгоритмічних списків відтворення та гарантувати, що механізм показуватиме вашу музику правильній аудиторії, посилюючи відкриття та перетворюючи випадкових слухачів на шанувальників.

ВИСНОВКИ

У ході виконання роботи було визначено, що порівняння ефективності алгоритмів машинного навчання для підбору рекомендацій музики з урахуванням уподобань користувача на платформах YouTube та Spotify є актуальним та цікавим напрямом досліджень. Було виявлено, які саме алгоритми та підходи працюють найкраще на кожній з розглянутих платформ.

Аналізуючи результати дослідження, можна зазначити, що обидві платформи, YouTube та Spotify, використовують складні алгоритми рекомендацій, які базуються на методах машинного навчання. Ці алгоритми враховують уподобання користувачів, використовуючи різноманітні дані, такі як: історія прослуховування, взаємодія з контентом, а також контекстуальну інформацію, щоб персоналізувати рекомендації музики.

Тривалість навчання нейронних мереж та підготовка даних можуть варіюватися залежно від платформи. Spotify, зосереджений на музиці, може мати меншу тривалість навчання, оскільки працює зі специфічними аудіофункціями. З іншого боку, YouTube, який пропонує як аудіо-, так і відео-контент, може вимагати більше часу на підготовку та обробку даних, оскільки включає в себе відео, описи, мітки та взаємодію з глядачами через функції коментування та оцінок «подобалось – не сподобалось».

Одним із важливих аспектів дослідження є оптимізація моделей рекомендацій, який є важливим кроком у поліпшенні ефективності систем підбору музичних рекомендацій на обох платформах. Шляхи оптимізації можуть включати використання нових алгоритмів або покращення існуючих, зміну методів підготовки даних, а також урахування додаткових контекстуальних факторів, таких як час дня, настрій користувача або його місцезнаходження.

Окрім того, було виявлено, що час на підготовку тестових і тренувальних даних також може впливати на ефективність рекомендаційної системи. Оптимізація цього процесу може включати швидшу обробку та підготовку даних, використання ефективних алгоритмів агрегації та фільтрації, а також урахування нових даних, які надходять у режимі реального часу.

Враховуючи отримані результати, можна зробити висновок, що подальші дослідження в галузі рекомендаційних систем для підбору мультимедійного контенту є дуже перспективними. Застосування нових методів та алгоритмів машинного навчання, розширення набору ознак та врахування додаткових контекстуальних факторів можуть сприяти поліпшенню точності та ефективності рекомендаційних систем.

Варто також зазначити про важливість звертання уваги на швидкість навчання моделей та підготовку даних. Оптимізація цих процесів може забезпечити ефективнішу та швидшу генерацію рекомендацій, покращуючи враження користувачів від взаємодії із системою.

Слід враховувати, що дослідження в сфері рекомендаційних систем має свої виклики та проблеми. Наприклад, розуміння та управління ризиками приватності, прозорість та пояснюваність рекомендацій, а також розуміння попереджувальних ознак алгоритмічного перекручування є важливими аспектами, які потребують подальшого дослідження та вдосконалення.

У підсумку, проведене дослідження показало, що порівняння ефективності алгоритмів машинного навчання для рекомендацій музики на платформах YouTube та Spotify є цікавою та актуальною темою. Апробація цих результатів може відбуватися через впровадження внесених вдосконалень у реальні рекомендаційні системи та оцінку реакції користувачів. Крім того, їх можна використати для підготовки наукових статей та доповідей, спрямованих на обмін знаннями та досвідом в області машинного навчання та рекомендаційних систем.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Inside YouTube, On YouTube's recommendation system, Cristos Goodrow. URL: https://blog.youtube/inside-youtube/on-youtubes-recommendation-system/

2. How the YouTube Algorithm Works in 2023: The Complete Guide. Paige Cooper. URL: https://blog.hootsuite.com/how-the-youtube-algorithm-works/

3. Як YouTube підтримує конфіденційність користувачів? URL: https://www.youtube.com/howyoutubeworks/our-commitments/protecting-user-data/

4. Inside Spotify's Recommender System: A Complete Guide to Spotify Recommendation Algorithms. URL: https://www.music-tomorrow.com/blog/how-spotify-recommendation-system-works-a-complete-guide-2022

5. J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, and D. Sampath. The youtube video recommendation system. In Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys '10, стор. 293–296, New York, NY, USA, 2010. ACM.

6. J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI, 2011.

7. S. Jean, K. Cho, R. Memisevic, and Y. Bengio. On using very large target vocabulary for neural machine translation. CoRR, abs/1412.2007, 2014.

8. D. Oard and J. Kim. Implicit feedback for recommender systems. In in Proceedings of the AAAI Workshop on Recommender Systems, стор. 8-83, 1998.

9. J. Weston, A. Makadia, and H. Yee. Label partitioning for sublinear ranking. In S. Dasgupta and D. Mcallester, editors, Proceedings of the 30th International Conference on Machine Learning (ICML-13), volume 28, pages 181–189. JMLR Workshop and Conference Proceedings, May 2013.

10.F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In AISTATSZ05, стор. 246-252, 2005.

11.J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In Proceedings of the International Joint Conference on

Artifcial Intelligence, IJCAI, 2011.

12. Aucouturier, J.J. & Pachet, F. (2002). Music similarity measures: What's the use? Proceedings of the 3rd International Conference on Music Information Retrieval. Paris, France

13. X. Yi, L. Hong, E. Zhong, N. N. Liu, and S. Rajan. Beyond clicks: Dwell time for personalization. In Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14, pages 113–120, New York, NY, USA, 2014. ACM.

14. Spotify WebAPI: Get Track's Audio Features. URL: https://developer.spotify.com/documentation/web-api/reference/get-audio-features

15. Acoustic Attributes Overview. URL: https://web.archive.org/web/20130903051953/http://developer.echonest.com/acoustic-attributes.html

16. A. Saravanou, F. Tomasi, R. Mehrotra and M. Lalmas. Multi-Task Learning of Graph-based Inductive Representations of Music Content. URL: https://research.atspotify.com/publications/multi-task-learning-of-graph-based-inductive-representations-of-music-content/

17. D. Pastukhov. Inside Spotify's Recommender System: A Complete Guide to Spotify Recommendation Algorithms. URL: https://www.music-tomorrow.com/blog/how-spotify-recommendation-system-works-a-complete-guide-2022

18. J. Wirfs-Brock, S. Mennicken, J. Thom. Giving Voice to Silent Data: Designing with Personal Music Listening History. URL: https://research.atspotify.com/giving-voice-to-silent-data-designing-with-personal-music-listening-history/

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

19. Nazarenko, D., Afanasieva, I., Golian, N., Golian, V. Investigation of the deep learning approaches to classify emotions in texts. CEUR Workshop Proceedings, 2021, 2870, pp. 206–224

20. Shopynskyi, M., Golian, N., Afanasieva, I. Long Short-Term Memory Model Appliance for Generating Music Compositions. 2020 IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T 2020 - Proceedings, 2021, pp. 239–242

21. Shatovska, T., Kamenieva, I., Tarasov, I. New module of text classification for IDA system. Experience of Designing and Application of CAD Systems in Microelectronics - Proceedings of the 10th International Conference, CADSM 2009, 2009, pp. 481–482