ISSN 1029-4171 (print) ISSN 2524-2407 (online)

**Обрії математики**

**УДК 512.64+514:004.89]:004.738.5** **DOI: https://doi.org/10.17721/1029-4171.2025/2.10**

**Валентин СОБЧУК, Д-р. техн. наук, Проф.**
**ORCID ID: 0000-0002-4002-8206**
**e-mail: sobchuk@knu.ua**
**Київський національний університет імені Тараса Шевченка, Київ, Україна**

**Ірина ЛЕБЕДЄВА, Канд. фіз.-мат. наук, Доц.**
**ORCID: 0000-0001-7150-1310**
**e-mail: lebedyevaiv@knu.ua**
**Київський національний університет імені Тараса Шевченка, Київ, Україна**

**Катерина КЕКАЛО, студентка**
**ORCID: 0009-0002-0677-3955**
**e-mail: erozkova777@gmail.com**
**Київський національний університет імені Тараса Шевченка, Київ, Україна**

# АНАТОМІЯ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

***Анотація.*** *У статті розглянуто принципи побудови сучасних рекомендаційних систем соціальних платформ на прикладі TikTok та Instagram. Показано, що основою їхньої роботи є двоетапна архітектура, яка поєднує швидкий пошук потенційно релевантного контенту (Retrieval) і точне ранжування відібраних кандидатів (Ranking). Детально описано модель двох нейронних «веж» (Two-Tower Neural Network), у межах якої користувач і контент подаються у вигляді векторів у багатовимірному просторі, а їхня відповідність визначається за допомогою косинусної близькості. Проаналізовано, як текстові, візуальні, аудіальні та метадані контенту об'єднуються в єдине векторне представлення, а також, як поведінкові та контекстуальні ознаки формують динамічний профіль користувача. Окрему увагу приділено порівнянню підходів TikTok і Instagram, зокрема їхнім різним цілям оптимізації, швидкості адаптації та ступеню складності моделей. Стаття має на меті продемонструвати, що рекомендаційні системи не є «чорною скринькою», а спираються на добре відомі ідеї лінійної алгебри, геометрії та машинного навчання, доступні для розуміння учнями старших класів і студентами початкових курсів.*

***Ключові слова:*** *рекомендаційні системи; Two-Tower Neural Network; Retrieval і Ranking; косинусна близькість; векторні представлення; TikTok; Instagram; машинне навчання.*

## 1. Вступ

Соціальні мережі стали головними генераторами інформації для мільярдів людей. Ми прокидаємося – і вже бачимо персональну добірку відео, мемів, новин чи оглядів. TikTok демонструє ролики, від переглядів яких важко відірватися; Instagram пропонує нові Reels, що ідеально збігаються з нашими настроями; YouTube, Facebook та інші сервіси щосекунди намагаються передбачити, що саме ми хочемо побачити далі.

Але як ці системи «вгадують» наші інтереси? Чи справді додатки «слухають» телефон, чи є раціональне математичне пояснення? Насправді за майже магічною точністю рекомендацій стоять цілком зрозумілі – хоча й дуже потужні – алгоритми машинного навчання. У цій статті ми спробуємо розібратися в анатомії таких

рекомендаційних систем, зосередившись на двох найвпливовіших платформах сучасності – TikTok та Instagram.

Попри те, що ці сервіси здаються дуже різними, їхні рекомендаційні системи мають спільну структуру. Обидві базуються на ідеї *двох нейронних «веж»* (*Two-Tower Models*) (Covington, Adams & Sargin, 2016), (Amatriain, 2012), які окремо описують користувача і контент, а потім порівнюють ці описи у вигляді багатовимірних векторів. Далі відбувається процес пошуку схожості, відбору найперспективніших відео серед мільйонів доступних, і фінального ранжування – визначення, що саме побачить користувач у першу чергу.

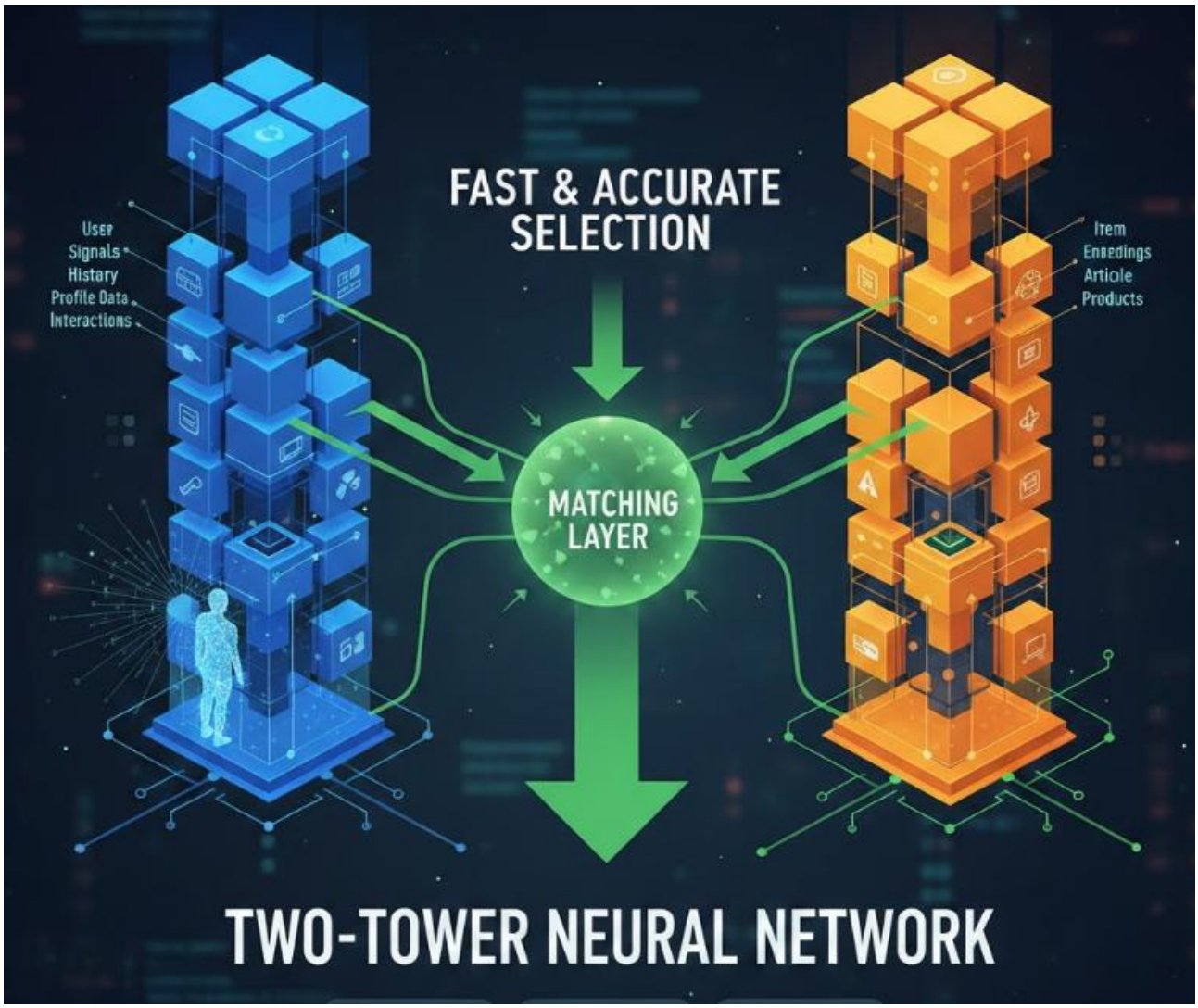


Рис. 1. Візуалізація архітектури Two-Tower Neural Network

У TikTok та Instagram працюють десятки різних алгоритмів, які щосекунди аналізують перегляди, лайки, час утримання уваги, попередні взаємодії, характеристики відео, текстові та візуальні ознаки, а часто й поведінку схожих користувачів. Хоча все це виглядає надскладною системою, її можна розкласти на зрозумілі математичні кроки, і саме це ми й зробимо.

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

Метою цієї статті є проілюструвати читачам, що рекомендаційні системи – це не загадкова «чорна скринька», а результат роботи геометрії, лінійної алгебри, теорії графів та оптимізаційних алгоритмів. Ми поглянемо на процес зсередини: від формування векторів контенту до ранжування, від пошуку найближчих сусідів до мультимодальних нейромереж.

## 2. Як працюють рекомендаційні системи: ідея двох «веж»

Сучасні платформи на кшталт TikTok, Instagram Reels, YouTube Shorts чи Netflix використовують масштабовані системи рекомендацій, здатні аналізувати мільярди взаємодій у реальному часі. Однією з найефективніших архітектур, що забезпечує швидкість і точність відбору контенту, є *Two-Tower Neural Network* – модель, у якій сигнал від користувача та від об'єкта рекомендації (відео/посту/товару) обробляється незалежними нейромережевими «вежами» (рис. 1) (Google Gemini, 2025).

Узагальнено, така система виконує дві ключові операції:

i. *кодування* користувача та контенту у векторні представлення,
ii. *оцінювання їхньої взаємної відповідності*.

Опишемо докладніше кожен етап.

### 2.1. Вежа користувача

*Вежа користувача* – це нейронна мережа, призначена для перетворення інформації про користувача на щільний вектор фіксованої довжини (embedding), який позначають $u$. Цей вектор є формалізованим описом інтересів, звичок і поведінкових патернів людини.

Вежа користувача перетворює весь набір наявної інформації на вектор $u$, який компактно відображає інтереси користувача у багатовимірному просторі (зазвичай 32–512-вимірному). Джерелами ознак тут є:

- *Історія переглядів*. Сучасні системи використовують *sequence models* (RNN, GRU, Transformers), щоб зрозуміти, як змінюються вподобання з часом. Модель аналізує: які відео переглядав користувач; як довго дивився кожне; чи додав у вибране; чи переглядав повторно; які категорії чи теми домінують у його історії.
- *Час утримання уваги* (*watch time*) є одним з ключових індикаторів. Якщо ролик переглянутий повністю, модель вважає його релевантним, а якщо ж користувач швидко прогорнув – це сигнал негативної взаємодії. Алгоритм не лише враховує абсолютний час, а й відсоток перегляду відео та місця, де користувач зупинявся або перемотував контент.
- *Типи відео, з якими він взаємодіє*. При цьому враховуються лайки, коментарі, шерінги, збереження, натискання «*не цікаво*», перегляди профілів авторів. Кожен тип взаємодії має власну вагу. Наприклад, «поділитися» зазвичай важливіше, ніж «лайк».

- *Географія*. За цією ознакою відстежують країну чи регіон проживання, часовий пояс, мову інтерфейсу, локальні тренди в даному регіоні. Рекомендації в даному випадку, особливо у TikTok, локалізуються.
- *Сигнали безпеки*. Система повинна гарантувати безпечний контент. Має бути забезпеченим фільтрація контенту за віковими обмеженнями, уникнення шкідливих тем, блокування матеріалів, позначених як потенційно небезпечні для неповнолітніх. User Tower враховує такі сигнали, щоб *не підбирати ризикований контент невідповідним споживачам.*
- *Час доби та інтенсивність користування.* У різний час доби користувач може віддавати перевагу різним типам контенту, наприклад, увечері – розважальний, вдень – освітній. Активні користувачі отримують більш «*свіжі*» рекомендації, а малоактивні користувачі – більш загального характеру. Власне це так звані *контекстуальні фактори*.
- *Поведінка схожих користувачів*. Система застосовує *колаборативну фільтрацію* – якщо група користувачів зі схожими поведінковими патернами масово взаємодіє з певним типом відео, то модель може рекомендувати його іншим користувачам у цій «кластерній» групі.

Опишемо математично представлення вектора користувача $u$.

У двовежевих рекомендаційних системах вектор користувача $u$ – це компактне числове представлення його інтересів. Він є елементом багатовимірного простору

$$u \in \mathbb{R}^d,$$

де $d$ – заздалегідь вибрана розмірність (наприклад, 64, 128, 256 або 512).

Щоб побудувати цей вектор, модель обробляє всі дані про користувача: історію переглядів, часові та поведінкові ознаки, географію, взаємодії та інше. Формально цей процес можна зобразити як перетворення

$$u = f_{\text{user}}(X),$$

де $X$ – сукупність усіх доступних ознак.

Нехай для кожного користувача існує послідовність переглядів. Кожний елемент цієї послідовності (кожне відео) кодується у вектор:

$$e_{c_t} \in \mathbb{R}^d.$$

Далі ця послідовність подається в модель, наприклад GRU або Transformer[1], які «витягують» з неї узагальнену інформацію про зміну інтересів користувача:

$$s = \text{SeqModel}\left(e_{c_1}, \dots, e_{c_T}\right).$$

[1] Трансформер (англ. Transformer) – це архітектура штучної нейронної мережі, заснована на механізмі багатоголової уваги, в якій текст перетворюється на числові представлення, які називаються токенами, а кожен токен перетворюється на вектор шляхом пошуку з таблиці вбудовування слів.

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

Отриманий вектор $s$ можна розглядати як «динамічний профіль» користувача. Окрім самого факту перегляду, модель враховує лайки, коментарі, шерінги, натискання «не цікаво», час перегляду.

При цьому кожній взаємодії відповідає своя вага, яка показує, наскільки така взаємодія інформативна. Узагальнений вплив усіх взаємодій можна подати у вигляді вектора:

$$r = \sum_{\text{переглянутих відео } c} \alpha_c \, v_c,$$

де $v_c$ – вектор відео, $\alpha_c$ – підсумкова вага взаємодії з цим відео (позитивної чи негативної).

При цьому модель також враховує додаткові фактори, як от мову інтерфейсу, регіон користувача, час доби, групи схожих користувачів тощо, які теж кодуються у вектори. Наприклад:

$$e_{\text{locale}}, e_{\text{country}}, e_{\text{time}}, e_{\text{cluster}}.$$

Це допомагає моделі розуміти локальні тренди та поведінку подібних (схожих) користувачів.

Усі описані вектори об'єднуються (*конкатенуються*[2]) в єдину велику ознакову репрезентацію:

$$x_{\text{emb}} = \left[ s \parallel r \parallel e_{\text{locale}} \parallel e_{\text{country}} \parallel e_{\text{time}} \parallel e_{\text{cluster}} \right].$$

Останній крок – пропускання об'єднаного вектора через *нейронну мережу* (MLP)[3]:

$$u = \text{MLP}(x_{\text{emb}}). \qquad (1)$$

Результатом є щільний вектор $u$, який у багатовимірному просторі відповідає «профілю» користувача і використовується для порівняння з векторами контенту.

Відтак, оскільки *послідовність переглядів* формує «динамічне ядро» інтересів користувача, *взаємодії* (наприклад, лайки чи пропуски) підсилюють або, навпаки, послаблюють вплив окремих відео на його вподобання, а *контекстуальні ознаки* (час доби чи місцезнаходження) додатково коригують рекомендації під конкретний момент,

[2] Конкатенація (від лат. *concatenatio* – зчеплення, об'єднання) – це операція послідовного з'єднання двох або більше об'єктів лінійної структури (найчастіше текстових рядків, але також списків чи інших послідовностей) в одну, створюючи новий, довший об'єкт. У програмуванні це робиться, щоб об'єднати слова, речення, дані з різних змінних, часто за допомогою оператора $+$.

[3] MLP – це штучна *нейронна мережа* прямого поширення, яка виконує серію математичних операцій над вхідними даними для створення прогнозу або результату.

то вся ця складна інформація проглядається у *векторі* $u$, який є *стислим цифровим підсумком* того, що людина найбільше любить дивитися.

### 2.2. Вежа контенту

Розглянемо тепер підхід до математичного підходу представлення *вектора контенту* $v$.

Метою *вежі контенту* (*Item Tower*) є перетворення всіх доступних ознак відео/поста (текст, аудіо, кадри, хештеги, метадані) у єдиний вектор фіксованої довжини:

$$v \in \mathbb{R}^d.$$

Формально вежа контенту – це функція

$$v = f_{\text{item}}(Y),$$

де $Y$ – множина всіх ознак конкретного відео.

Нехай відео має опис або підпис – тобто, певний текст $T$, який можна подати як послідовність токенів:

$$T = (w_1, w_2, \dots, w_n).$$

Кожен токен кодується у вектор:

$$e_{w_i} \in \mathbb{R}^{d_t}.$$

Для цих векторів застосовується модель обробки тексту (наприклад, Transformer або LSTM[4]):

$$t = \text{TextModel}(e_{w_1}, \dots, e_{w_n}) \in \mathbb{R}^d.$$

Тоді вектор $t$ відображає зміст тексту та його тематику.

Якщо контент має відео або обкладинку, беруть вибрані кадри:

$$I = \{F_1, F_2, \dots, F_m\}.$$

Кожен кадр обробляється візуальною моделлю (наприклад, CNN[5], Vision Transformer):

$$e_{F_j} = \text{VisionModel}(F_j) \in \mathbb{R}^d.$$

[4] LSTM (long short-term memory) – це особливий різновид архітектури рекурентних нейронних мереж, здатний до навчання довгостроковим залежностям.

[5] CNN або ConvNet (convolutional neural network) згорткові нейронні мережі – це клас глибоких штучних нейронних мереж прямого поширення, який успішно застосовувався до аналізу візуальних зображень.

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

Кадри агрегуються, наприклад, середнім або *attention-механізмом* – це механізм, який інтуїтивно імітує когнітивну увагу. Він обчислює «м'які» (soft) ваги для кожного слова, точніше, для його вкладення, у контекстному вікні. Ці ваги можна обчислювати або паралельно (як у трансформерах), або послідовно (як у рекурентних нейронних мережах). Як результат отримуємо вектор $v_{\text{img}}$, який характеризує візуальний зміст ролика:

$$v_{\text{img}} = \text{Agg}(e_{F_1}, \dots, e_{F_m}) \in \mathbb{R}^d.$$

Аудіодоріжка подається у вигляді спектрограми $S$ – візуального представлення аудіосигналу, яке показує, як розподіл енергії (амплітуда) звуку змінюється з часом у різних частотах, або MFCC-фіч – набору ознак, отриманих зі спектрограми, які широко використовуються для розпізнавання мови та ідентифікації звуку. Вони імітують те, як людське вухо сприймає частоти. Модель аудіо (Audio CNN або Audio Transformer) обчислює:

$$a = \text{AudioModel}(S) \in \mathbb{R}^d.$$

Вектор $a$ *кодує ритм, тональність, інструменти, голос, темп* та інші аудіофічі.

Опишемо тепер, як модель працює з метаінформацією – хештегами, авторами, тривалістю, форматом.

Нехай контент має множину хештегів: $H = \{h_1, h_2, \dots, h_k\}$. Кожний хештег має ембеддінг[6] $e_{h_i} \in \mathbb{R}^d$. Відтак агрегований вектор хештегів має вигляд:

$$v_H = \text{Agg}(e_{h_1}, \dots, e_{h_k}) \in \mathbb{R}^d.$$

Автор має вектор представлення

$$e_{\text{author}} \in \mathbb{R}^d,$$

який враховує популярність, стиль, тематику, історію взаємодій.

Тривалість, формат, технічні параметри – це числові й категоріальні ознаки, які після нормування або вкладення утворюють вектор:

$$e_{\text{meta}} \in \mathbb{R}^d.$$

Усі джерела інформації конкатенуються:

[6] Ембеддінг (англ. word embedding) – вкладання слів – це загальна назва низки методик мовного моделювання та навчання ознак в обробці природної мови (ОПМ), в яких слова або фрази зі словника відображують у вектори дійсних чисел. Концептуально воно дає математичне вкладення з простору з багатьма вимірами, по одному на слово, до неперервного векторного простору набагато нижчої розмірності.

$$uy_{\text{emb}} = \left[t \parallel v_{\text{img}} \parallel a \parallel v_H \parallel e_{\text{author}} \parallel e_{\text{meta}}\right]. \quad (2)$$

Останній крок – пропускання об'єднаного вектора через MLP:

$$v = \text{MLP}(y_{\text{emb}}) \in \mathbb{R}^d.$$

Це і є *векторне представлення кожного відео або поста*, яким система оперує для пошуку відповідних користувачів.

Іншими словами, наш розум сприймає цілісне мультимедійне повідомлення, коли *текст* пояснює, *про що* йдеться у відеоматеріалі, а *відео/зображення* демонструють, *що саме* ми бачимо, тоді як *аудіо* передає, *як* це звучить; при цьому *хештеги й автор* одразу вказують, *до якої категорії чи стильової ніші* належить цей контент, і вся ця інформація, разом із *метаінформацією*, що описує його формат, зливається у *вектор* $v$, який, мов *«цифровий відбиток»*, унікально характеризує контент у багатовимірному просторі.

### 2.3. Як вимірюється схожість між користувачем і відео

Оскільки $u$, і $v$ – це вектори, їх можна порівняти за допомогою добре відомого з курсу геометрії способу.

Для визначення, який контент найбільше підходить користувачу, обчислюється схожість між вектором користувача $u$ та *ембеддінгами контенту* $v_i$. Одним з найпоширеніших методів є *косинусна подібність* (Wu, Guanfeng, et al. 2024):

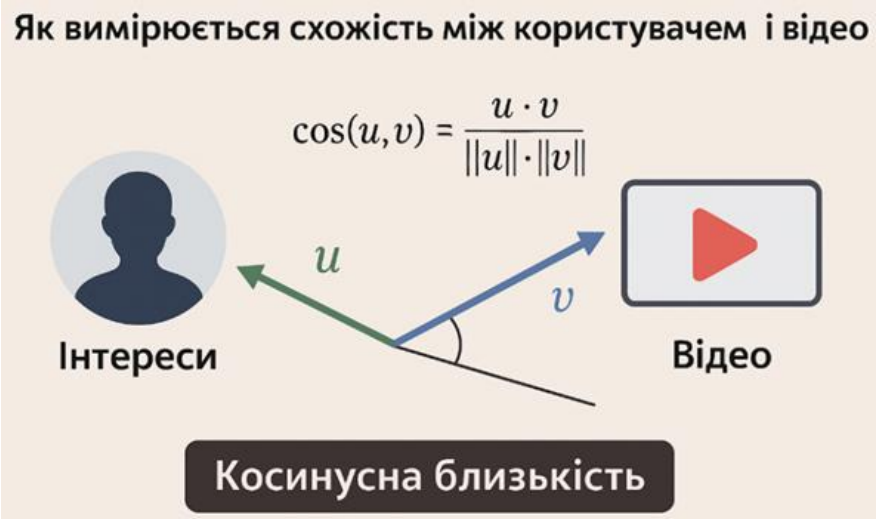


Рис. 2. Візуалізація косинусної подібності

$$\cos(u, v) = \frac{u \cdot v}{\| u \| \cdot \| v \|}. \quad (3)$$

Тут

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

$$u \cdot v = \sum_i u_i \cdot v_i -$$

скалярний добуток, а

$$\| u \| = \sqrt{\sum_i u_i^2} -$$

довжина вектора. Косинусна подібність дає значення від $-1$ до $1$.

Якщо $\cos(u, v)$ близький до $1$ – вектори спрямовані однаково (максимальна схожіть) – ролик дуже схожий на те, що цікавить користувача. Якщо $\cos(u, v)$ дорівнює $0$ – вектори ортогональні (не пов'язані). Якщо $\cos(u, v)$ близький до $-1$ – вектори спрямовані у протилежні боки.

Саме використання $\cos(u, v)$ є зручним, оскільки не важливою є довжина вектора, тобто кількість історії чи інформації; модель зважає лише на напрямок, тобто на тип змісту чи інтересів (рис. 2). Це дуже добре працює у великих нейронних мережах.

**Приклад 1.** *Вектор користувача* $u = (0.5, 0.2, -0.1, 0.8, 0.3)$, *вектори відео* $v_1 = (0.4, 0.1, 0.0, 0.7, 0.2)$; $v_2 = (-0.3, 0.8, 0.5, 0.0, 0.1)$; $v_3 = (0.5, 0.25, -0.05, 0.6, 0.35)$. *Обчислити косинусну подібність.*

Обчислимо косинусну подібність, використовуючи (3). Отримуємо:

$$\cos(u, v_1) \approx 0.989, \qquad \cos(u, v_2) \approx -0.010, \qquad \cos(u, v_3) \approx 0.982.$$

*Висновок.* Відео 1 має найвищий показник схожості, отже, система віддасть перевагу *відео 1*.

Для пересічного користувача в реальному житті це працює таким способом: уявімо, що у користувача є «вектор інтересів» – він любить футбол, хімію і гумористичні відео – кожне відео теж має свій «вектор змісту»: наприклад, про спорт, науку та стиль подачі. Якщо напрямки цих «векторів» майже збігаються, то система каже: «О! Це відео схоже на те, що любить цей користувач». Тоді алгоритм і показує користувачу відповідне відео у стрічці.

### 2.4. Навчання моделей

Системи TikTok та Instagram порівнюють мільйони прикладів «користувач – переглянутий ролик» та «користувач – пропущений ролик». На основі цих даних моделі оновлюються і краще вчаться передбачати поведінку.

По суті, модель – це *дитина, яка вчиться на своїх помилках, але робить це з величезною швидкістю*. Чим більше вона бачить прикладів «успіху» та «провалу», тим краще вона розуміє, які саме ознаки в користувачі (любить котиків, дивиться після 18:00) збігаються з ознаками в контенті (відео про котиків, завантажено о 17:50).

Власне тренування є своєрідною грою в «*Успіх*» чи «*Провал*». Кожна система збирає мільйони прикладів, які можна розділити на два типи:

- *Позитивний приклад (Успіх).* Користувачі довго подивилися якесь відео. Це означає, що *збіг* між стовпчиком користувача і стовпчиком відео був *вдалим*.
- *Негативний приклад (Провал).* Користувачам показали якесь відео, а вони його *майже одразу прогорнули* (пропустили). Це означає, що *збіг* був *невдалим*.

Модель намагається передбачити: «Якщо я покажу цьому користувачеві ось це відео, чи він його подивиться?». Працює так званий *конвеєр* Прогноз – Реальність – Оновлення (рис. 3). Це працює наступним способом.

| **Прогноз:** | **Реальність:** |
|---|---|
| Модель робить свій прогноз. Наприклад, "Я думаю, він подивиться на 80%". | Модель дивиться на реальні дані. Якщо прогноз був неправильний (наприклад, модель думала, що ви подивитеся, а ви прогорнули), вона розуміє, що помилилася. |
| **Оновлення:** | |
| Далі, модель робить маленьку корекцію своїх внутрішніх налаштувань (це називається *оновлення ваг*). Вона ніби каже собі: «Ага, наступного разу, коли я бачу цього користувача (який любить музику), і це відео (яке про спорт), я маю знизити свій прогноз!». | |

Ці корекції відбуваються мільйони разів на хвилину на мільйонах прикладів.

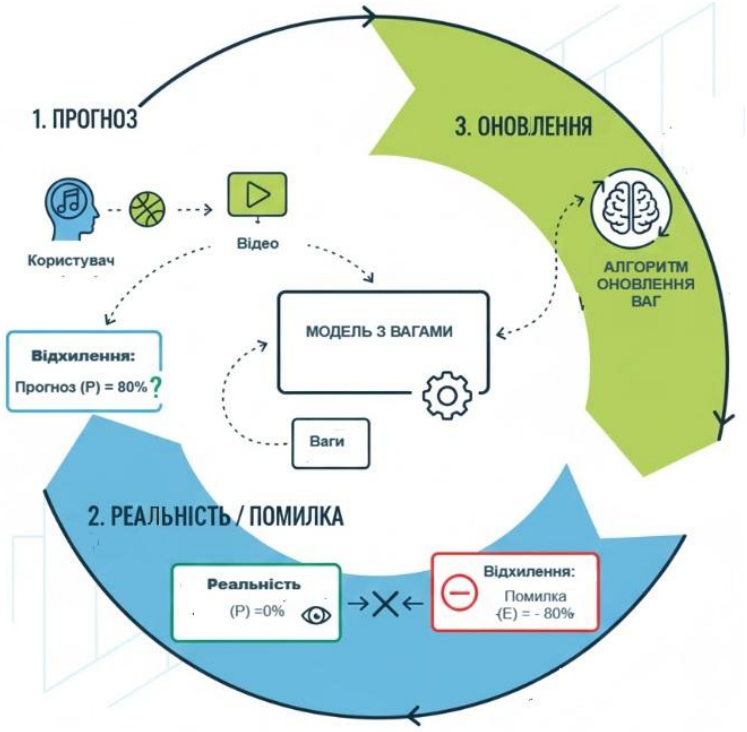


Рис. 3. Конвеєр Прогноз – Реальність – Оновлення

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

### 3. Пошук кандидатів: як із мільйонів знайти тисячі

Вище показано, що кожен користувач описується вектором $u$, кожне відео описується вектором $v$ – чим ближчі ці вектори, тим більша ймовірність, що відео сподобається. Але виникає проблема. Навіть якщо існує можливість порівняти вектори користувача і відео, робити це для кожного існуючого ролика – неможливо. TikTok, YouTube чи Instagram щосекунди мають доступ до сотень мільйонів роликів, і повторне обчислення схожості для кожного з них, це займе не секунди, не хвилини, а години.

Очевидно, так робити не можна. Тому система діє у два етапи:

1. *Retrieval* – це швидкий чорновий відбір, його мета швидко знайти кілька тисяч можливих варіантів;
2. *Ranking* – уже потім уважно відсортувати їх.

Тому використовується перший етап – retrieval, або пошук кандидатів.

#### 3.1. Ідея етапу Retrieval

На цьому етапі необхідно із сотень мільйонів відео залишити лише 1 000 – 10 000, які *в принципі* можуть сподобатися користувачу. Система не шукає ідеальний результат, допускає помилки, головне – *не пропустити цікаві відео*.

Як це виглядає образно? Уявімо, що:

- всі відео – це точки у величезному просторі;
- користувач – це вектор, яка показує напрямок його інтересів.

Тоді Retrieval – це запитання: «Які точки знаходяться *приблизно в тому ж напрямку*, що й вектор користувача?» Користувачам не потрібні всі точки. Їм потрібні лише *найближчі за напрямком*.

При цьому замість того, щоб порівнювати користувача з кожним відео, система робить розумніше – *об'єднує відео у групи* (*кластери*). Наприклад: одна група – спортивні відео; інша – гумор; ще одна – освіта; ще одна – музика. Кожна така група має *свій «середній вектор»*.

На практиці процес пошуку працює таким способом (рис. 4).


Рис. 4. Схема роботи процесу пошуку на етапі Retrieval

У результаті реалізації процесу пошуку система зі 100 000 000 відео відбирає 5 000 відео-кандидатів.

Чому це працює швидко? Тому що порівняти вектор користувача з 1 000 групами – дуже швидко, а порівняти зі 100 000 000 відео – дуже повільно.

Retrieval *жертвує точністю*, але *виграє у швидкості*. Retrieval не вибирає найкраще відео – він лише каже: «*Ось список тих, що можуть підійти*».

Після цього включається другий етап – *Ranking*, який детально аналізує кожне відео, враховує більше факторів, остаточно вирішує, що саме показати.

### 3.1. Ідея етапу Ranking

Після етапу Retrieval система вже має *невеликий список кандидатів* – не мільйони відео, а тисячі, приблизно від 1 000 до 10 000. Тепер постає нове питання: *Які з них показати першими, а які – взагалі не показувати?*

Власне для цього існує етап Ranking (ранжування) – точний і повільніший відбір. Він працює повільніше, аналізує кожне відео набагато детальніше, робить остаточний вибір.

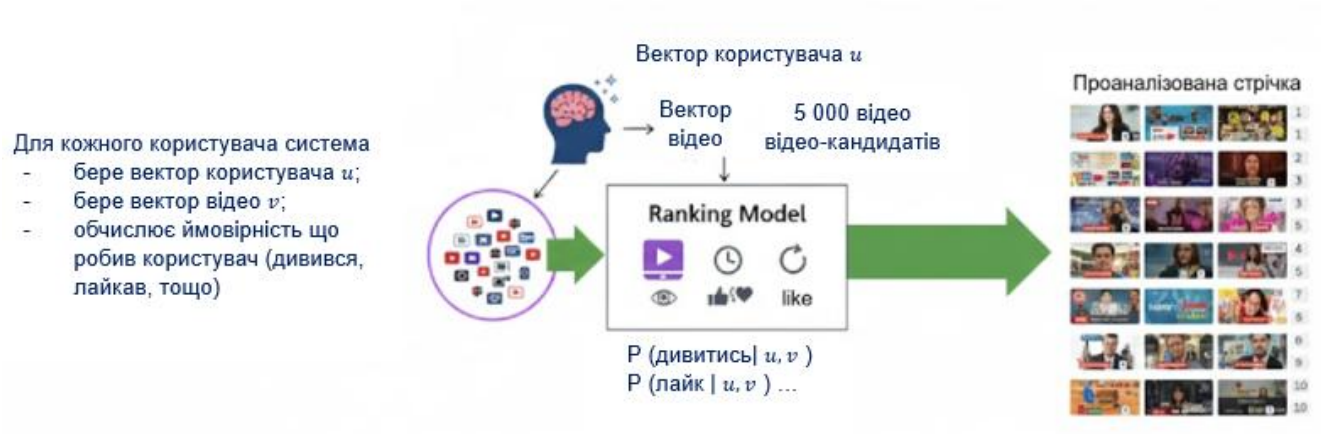


Рис. 5. Схема роботи процесу пошуку на етапі Ranking

Тобто, коли на етапі відбору система каже: «Ось можливі варіанти», то на етапі ранжування – «Ось найкращі з них» (рис. 5). При цьому для кожного відео-кандидата система:

1. бере *вектор користувача*;
2. бере *вектор відео*;
3. обчислює *ймовірність*, що користувач: подивиться відео, додивиться його до кінця, поставить лайк, поділиться з іншими.

Результатом цих дій етапу Ranking є можливість відповісти не на питання «схоже чи не схоже», а на питання *«що саме з цього варто показати зараз?».* Відмінності етапів Ranking та Retrieval дано у табл. 1.

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

Таблиця 1

Характеристика відмінностей етапів Ranking та Retrieval

| **Retrieval (пошук/відбір)** | **Ranking (ранжування)** |
|---|---|
| ***швидкий***, | повільніший, |
| грубий, | ***точний***, |
| порівнює загальні інтереси, | аналізує багато деталей, |
| працює з мільйонами. | працює з тисячами. |

Ranking дивиться на набагато більше речей, ніж Retrieval, зокрема: чи дивився користувач схожі відео раніше, як швидко він зазвичай перегортає контент, чи популярне відео серед схожих користувачів, чи свіже це відео, чи безпечне воно для цього користувача, у який час доби відбувається перегляд? Усе це об'єднується в *одну оцінку*.

Проілюструємо як це може працювати на прикладі.

**Приклад 2.** Уявімо, що Retrieval знайшов 5 відео:

1. про футбол;
2. про меми;
3. про фізику;
4. про рецепт піци;
5. про котиків.

Ranking може вирішити:

- зараз вечір → краще легкий контент;
- користувач сьогодні дивився меми;
- відео з котиками популярне серед схожих користувачів.

У результаті порядок, наприклад, стане таким:

1. Котики;
2. Мем;
3. Футбол;
4. Піца;
5. Фізика.

Підсумовуючи, зазначимо, що разом Retrieval швидко знаходить, що *можна* показати, а Ranking точно вирішує, що показати *першим*. Ця двохетапна архітектура є основою для побудови *швидких, точних і високоперсоналізованих* рекомендаційних систем, які здатні обслуговувати мільйони користувачів.

### 4. TikTok vs Instagram: у чому різниця систем

Instagram (укр. Інстаграм) – соціальна мережа, що базується на обміні світлинами, дозволяє користувачам робити світлини, застосовувати до них фільтри, а також

поширювати їх через свій сервіс і низку інших соціальних мереж. Належить компанії Meta Platforms. Є одним із найпопулярніших сервісів у мистецтві айфонографії.

Instagram розробили Кевін Систром та Майк Крігер, які обидва із Сан-Франциско, переорієнтувавши свій проєкт Burbn на мобільні фотографії. Застосунок з'явився в магазині App Store компанії Apple 6 жовтня 2010.

TikTok (відомий у Китаї як Douyin, кит. 抖音, «коротке відео Доуїнь») – китайський соцмедійний застосунок для створення та поширення коротких відео та проведення онлайн-трансляцій. Дозволяє створювати, редагувати і ділитися відеороликами тривалістю від кількох секунд до кількох хвилин, зазвичай під музику або якийсь аудіозапис.

Сервіс запущено у вересні 2016 року китайською компанією ByteDance. Це найпопулярніша платформа для коротких відео в Азії, яка поширилася на інші частини світу і набирає популярність. Кількість користувачів програми, які мешкають у 170 країнах, сягнула 1 млрд.

Філософія Instagram базується на використанні великої історичної бази користувачів та взаємодії між друзями/підписниками. Головною задачею соціальної мережі є розв'язання *десятків паралельних задач* – завдань рекомендацій для стрічки, історій, Explore, Reels, Shop, Direct тощо.

Філософія TikTok ґрунтується на утриманні уваги (Watch-time) – максимізації часу, який користувач проводить за переглядом відео. Головною задачею застосунку є зосередженість на одній ключовій меті, що забезпечує *максимум релевантності коротких відео.*

TikTok обробляє великі обсяги даних за короткий час**.** Система дуже швидко отримує фідбек про те, що працює, а що ні, завдяки швидкому споживанню контенту. Завдяки великій кількості даних і фокусу на короткому відео швидко навчається. При цьому більший акцент спрямовано на раннє прогнозування вірусності – система може швидко підхопити новий контент, що має потенціал, і показати його широкій аудиторії.

Instagram має більшу історичну базу користувачів. Навчається на тривалій історії взаємодії користувача з платформою. Система використовує складнішу, але повільнішу модель навчання через вирішення багатьох задач одночасно. Для прогнозування суттєвим є вплив перехресних взаємодій – рекомендації значною мірою залежать від взаємодії: коментарі, історії, підписки.

*Модель авторів* у TikTok досить сильна – будь-який новачок може «вистрілити». Алгоритм надає шанс новому контенту та авторам, незалежно від кількості підписників, якщо контент релевантний. Водночас безпека системи менш акцентована (або більш динамічна) порівняно з Instagram. Для якої характерною є строга система безпеки (наприклад, DV365). Це може уповільнювати швидкість поширення контенту, але забезпечує більшу надійність та відповідність стандартам.

Модель авторів Instagram вирізняється більшою залежність від соціального графу. Хоча Reels змінює це, традиційно вірусність більше пов'язана з наявністю аудиторії та взаємодіями.

Для TikTok притаманна висока конзистентність – єдиний фокус на короткому відео. При цьому весь додаток, по суті, є єдиною стрічкою коротких відео.

ISSN 1029-4171 (print) ISSN 2524-2407 (online)

Водночас для Instagram характерна низька конзистентність – різні алгоритми для фото, історій, відео та Reels. Explore та Reels мають різні цілі та різні моделі. Explore фокусується на статичному контенті та акаунтах, які можуть зацікавити, тоді як Reels конкурує з TikTok і має схожу мету.

Головна причина, чому TikTok часто перемагає у швидкості росту та залученні, криється у фокусі та простоті його моделі.

Обидві платформи постійно донавчають свої моделі на свіжих даних, щоб швидко адаптуватися до нових вподобань користувача та підтримувати високу релевантність рекомендацій.

## 5. Висновки

У роботі показано, що ефективність рекомендаційних систем сучасних соціальних мереж базується на поєднанні математично строгих моделей і масштабних інженерних рішень. Використання векторних представлень користувачів і контенту дозволяє звести складну задачу персоналізації до геометричної проблеми пошуку та порівняння у багатовимірному просторі. Двоетапна архітектура Retrieval – Ranking забезпечує одночасно високу швидкість роботи системи та точність рекомендацій, що є критично важливим за умов мільйонів користувачів і контентних об'єктів. Порівняльний аналіз TikTok і Instagram засвідчує, що різні цілі платформ приводять до різних компромісів між складністю моделей, швидкістю навчання та глибиною персоналізації. Отримані результати підтверджують, що рекомендаційні системи є яскравим прикладом практичного застосування лінійної алгебри, теорії ймовірностей і нейронних мереж та можуть слугувати ефективним навчальним матеріалом для популяризації сучасної математики й інформатики серед молоді.

**Список використаних джерел**

Amatriain X. (2012). Building industrial-scale real-world recommender systems. In Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, pages 7– 8, New York, NY, USA. ACM.

Paul Covington, Jay Adams, Emre Sargin *Deep Neural Networks for YouTube Recommendations*. // Google, Mountain View, CA. [Електронний ресурс]. Режим доступу: [https://liamzebedee.com/ml/papers/deepnn-youtube.pdf]

Google. Gemini [Електронний ресурс]. Режим доступу: [ai.google/gemini/]

Wu, Guanfeng, et al. "Multi-modal video search by examples – A video quality impact analysis." *IET Computer Vision* 18.7 (2024): 1017-1033. https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/cvi2.12303

*Отримано редакцією журналу: 01.11.2025*
*Прорецензовано: 25.11.2025*
*Схвалено до друку: 26.12.2025*

**Valentyn SOBCHUK, Dr.Sc.Tech., Prof.**
**ORCID ID: 0000-0002-4002-8206**
**e-mail: sobchuk@knu.ua**
**Taras Shevchenko National University of Kyiv, Kyiv, Ukraine**

**Iryna LEBEDYEVA, Ph.D (Phys&Math), Assoc. prof.**
**ORCID ID: 0000-0001-7150-1310**
**e-mail: lebedyevaiv@knu.ua**
**Taras Shevchenko National University of Kyiv, Kyiv, Ukraine**

**Kateryna KEKALO, Student**
**ORCID: 0009-0002-0677-3955**
**e-mail: erozkova777@gmail.com**
**Taras Shevchenko National University of Kyiv, Kyiv, Ukraine**

## ANATOMY OF RECOMMENDATION SYSTEMS

***Abstract.*** *This article examines the principles underlying modern recommendation systems used by social media platforms, with TikTok and Instagram serving as representative examples. It is shown that their operation is based on a two-stage architecture that combines fast retrieval of potentially relevant content with accurate ranking of the selected candidates. The Two-Tower Neural Network model is described in detail, within which users and content items are represented as vectors in a high-dimensional space, and their relevance is assessed using cosine similarity. The paper analyzes how textual, visual, audio, and metadata features of content are integrated into a unified vector representation, as well as how behavioral and contextual signals are used to construct a dynamic user profile. Particular attention is given to a comparative analysis of TikTok and Instagram, highlighting differences in optimization objectives, adaptation speed, and model complexity. The article aims to demonstrate that recommendation systems are not a "black box" but are grounded in well-established concepts from linear algebra, geometry, and machine learning, which are accessible to senior high school students and early undergraduate learners.*

***Keywords:*** *recommendation systems; Two-Tower Neural Networks; retrieval and ranking; cosine similarity; vector embeddings; TikTok; Instagram; machine learning.*