Ярослав Салабай, Ігор Капран*
[1] НЛТУ України, Львів, Україна, salabajslavcik@gmail.com
[2] НЛТУ України, Львів, Україна. ORCID: 0000-0001-9666-9531,
kapran@nltu.edu.ua

Рекомендаційна система для персоналізованого підбору одягу засобами нейронних мереж

Анотація. Проведено огляд та аналіз рекомендаційних систем для персоналізованого підбору одягу, їхні особливості, переваги та недоліки. Визначені та проаналізовані основні інформаційні та програмні засоби необхідні для реалізації завдання. Досліджені параметри нейронних мереж, які впливають на процес формування рекомендацій. Програмне забезпечення реалізовано за допомогою таких технологій: технологія R-FCN, нейронна мережа ResNet.

Ключові слова – рекомендаційна система, нейрона мережа, обробка зображень, машинне навчання, класифікація, детектування.

Рекомендаційні сервіси - це підродина систем для фільтрації контенту, що надають користувачеві ті елементи, які могли б його зацікавити. Рекомендації підбираються на основі преференцій та поведінки користувача. Система має спрогнозувати реакцію на той чи інший елемент – і запропонувати інші, які можуть також сподобатися. При програмуванні рекомендаційних систем застосовується безліч методів, але варто звернути увагу на три найпростіші, які використовуються найчастіше. Це колаборативну фільтрація (collaborative filtering), контентна фільтрація (content-based filtering) та експертні системи (knowledge-based systems) [1].

Подібна технологія та взагалі нейронні мережі здатні вирішити до 80% складних та високоінтелектуальних завдань, з якими щодня зустрічається людина. Питання лише в тому, хто першим реалізує подібну технологію та розвантажить користувача від рутинної роботи, звільнивши його простір для творчості та розвитку[1]. Дана рекомендаційна система призначена для використання в інтернет-магазинах як рекомендаційний сервіс.

Математична модель. Для того, щоб отримати рекомендації, системи фільтрації товарів пов'язують дві основні сутності: користувачів та товари.

* науковий керівник

Найпростіший спосіб зв'язку – це вказаний користувачем рейтинг (оцінка) товару. Якщо ми маємо дані про оцінку товарів різними користувачами, то ми отримуємо матрицю користувач – товар.

Математичною мовою це означає, що є метрика близькості між товарами, заснована на оцінках користувачів. Найпростішою мірою близькості є косинус кута між відповідними векторами оцінок для користувача користувачами:

$$sim(\vec{a}, \vec{b}) = \frac{\vec{a} * \vec{b}}{|\vec{a}| * |\vec{b}|} \quad (1)$$

Однак у реальних додатках починають виникати складності, пов'язані, наприклад, з тим, що різні товари оцінює різне число користувачів, а оцінки кожного користувача найчастіше зводяться до однієї зі сторін [2]. Для того, щоб уникнути цієї проблеми, можна застосувати модифіковану косинусну метрику:

$$sim(\vec{a}, \vec{b}) = \frac{\sum_{u \in U}(r_{u,a} - \overline{r_u})(r_{u,b} - \overline{r_u})}{\sqrt{\sum_{u \in U}(r_{u,a} - \overline{r_u})^2} \sqrt{\sum_{u \in U}(r_{u,b} - \overline{r_u})^2}} \quad (2)$$

де U – множина користувачів, які оцінили як a, так і b.

Подальші проблеми виникають, коли у великих додатках виявляється, що отримати оцінку від кожного користувача може лише украй невелика кількість товарів. Отже, до явних оцінок треба додавати неявні, засновані у тому, які товари як довго відвідувач переглядав, які купував тощо. Природно, інтерпретація неявних даних – складніше завдання, і його рішення завжди наближене, отже, призводить до появи шуму. Отже, у реальних завданнях матриця користувач – товар виходить великою, розрідженою та зашумленою.

На основі прикладної математики дану матрицюд дуже легко перетворити у простішу. А саме, за допомогою сингулярного розкладання отримуємо оптимально наближену до заданої матриці M іншу матрицю M_k меншого рангу k:

$$M_k = U_k \sum k \, V_k^* \quad (3)$$

де матриці U_k, $\sum_k$, та V_k^* виходять з відповідних матриць в сингулярному розкладанні матриці M обрізанням до рівно k перших стовпців (елементи матриці $\sum$ впорядковані по незростанню).

* науковий керівник

Отже, ми виконуємо свого роду стиснення інформації, що міститься в *M*. Це стиснення відбувається з втратами - в наближенні зберігається лише найбільша частина матриці *M*, а шум відфільтровується. Ще один погляд на таку апроксимацію полягає в тому, що всі товари проектуються в простір меншої розмірності *k*, в якій визначається відстань між ними. За приблизно такою схемою працює досить багато рекомендаційних систем [3].

Програмна реалізація

Завдання та основний use-case системи звучить досить просто та зрозуміло:

- користувач подає на вхід (наприклад, за допомогою мобільного додатка) фотографію, на якій присутні предмети одягу та/або сумки та/або взуття;
- система визначає (детектує) усі ці предмети;
- знаходить до кожного їх максимально схожі (релевантні) товари в реальних інтернет-магазинах;
- видає користувачеві товари з можливістю перейти на конкретну сторінку товару для купівлі.

Для початку необхідно досить чітко визначитися з тими категоріями, які будуть усередині нашої системи, тобто ті категорії, які детектуватиме нейронна мережа. Тут важливо розуміти, що чим ширший і детальніший список категорій, тим він є більш універсальним, оскільки велика кількість вузьких невеликих категорій типу mini-dress, midi-dress, maxi-dress завжди можна в один дотик об'єднати в одну категорію типу dress, але не навпаки. Іншими словами, рубрикатор потрібно добре продумати і скласти на початку проекту. Ми складали рубрикатор, взявши за основу кілька великих магазинів, таких як Lamoda.ua, Amazon.com, і постаралися зробити його з одного боку максимально широким, з іншого боку, максимально універсальним для того, щоб надалі простіше було пов'язати категорії детектора з категоріями різних інтернет магазинів (рис. 1).

* науковий керівник

Рисунок 1. Приклад категорій рубрикатора

Для того, щоб шукати надалі схожі товари, нам необхідно створити велику базу того, серед чого ми шукатимемо. Якість пошуку схожих зображень безпосередньо залежить від розміру пошукової бази, яка повинна перевищувати як мінімум 100К зображень, а краще 1М зображень.

Отже, щоб створити велику базу зображень, потрібно обробляти каталоги різних інтернет-магазинів. Ось що містить в себе цей процес:

- для початку потрібно знайти XML-фіди інтернет-магазинів, зазвичай їх можна знайти у вільному доступі в інтернеті, або запросивши у самого магазину, або в різних агрегаторах типу Admitad;
- фід обробляється (парситься) спеціальною програмою - краулером, який завантажує всі зображення з фіда, складає їх на жорсткий диск (точніше на мережеве сховище, до якого підключено ваш сервер), записує всю мета-інформацію про товари в БД;
- далі запускається інший процес - індексатор, який розраховує для кожного зображення бінарні 128-розмірні ознаки вектора. Можна об'єднати краулер та індексатор в один модуль або програму, але у нас історично склалося, що це були різні процеси. Це в основному було пов'язано з тим, що спочатку ми розраховували дескриптори (хеш) для кожного зображення розподілено на великому парку машин, так як це був дуже ре-

* науковий керівник

сурсномісткий процес. Якщо ви працюєте тільки з нейронними мережами, то вам вистачить першої машини з GPU;

- бінарні вектори записуються в БД, всі процеси завершуються і ваша база товарів готова до подальшого пошуку;

Оскільки всі магазини мають різні каталоги з різними категоріями всередині, то нам потрібно зіставити категорії всіх фідів, які містяться у нашій БД з категоріями детектора (точніше класифікатора) продуктів, ми називаємо це процесом створення мапінгу. Це ручна рутинна, але корисна робота, під час якої оператор, редагуючи вручну звичайний XML-файл, зіставляє категорії фідів в БД з категоріями детектора. В якості даних, які використовувалися для навчання та тестування (так звані навчальна та тестова вибірки), ми брали всілякі зображення з Інтернету: пошук по Google картинкам, сторонні розмічені датасети, соціальні мережі, сайти модних журналів, інтернет магазини одягу, взуття, сумок. Розмітка проводилася за допомогою самописного інструменту, результатом розмітки стали набори зображень і файлів *.seg до них, в яких зберігаються координати об'єктів і мітки класів для них. У середньому кожну категорію було розмічено від 100 до 200 зображень, загальна кількість зображень за 205 класами становило 65 000.

Після того, як готова навчальна та тестова вибірки, ми робили повторну перевірку розмітки, віддаючи всі зображення іншому оператору. Це дозволило відфільтрувати велику кількість помилок, які сильно впливають на якість навчання нейронної мережі, тобто детектора та класифікатора. Потім ми запускаємо навчання нейронної мережі стандартними засобами і "знімаємо" черговий снепшот нейронної мережі через кілька днів. У середньому час навчання детектора та класифікатора на обсязі даних 65 000 зображень на GPU порядку Titan X становить приблизно 3 дні.

Готову нейронну мережу необхідно якось перевірити на якість, тобто оцінити чи стала поточна версія мережі кращою за попередню і на скільки:

- тестова вибірка складалася з 12 000 зображень і була розмічена так само як навчальна;

* науковий керівник

- ми написали невеликий інструмент, який проганяв всю тестову вибірку через детектор і складав таблицю такого виду (рис. 2);

- ця таблиця додається в Excel на нову вкладку та порівнюється з попередньою вручну або за допомогою вбудованих формул Excel;

- на виході ми отримуємо загальні показники TPR/FPR детектора та класифікатора по всій системі та по кожній категорії окремо. Після того, як ми задетектували предмети гардероба на фотографії, запускаємо механізм пошуку схожих зображень, ось як він функціонує:

* для всіх вирізаних фрагментів зображення (задетектованих товарів) розраховуються нейромережні 128-бітові бінарні вектори ознак за формою та за кольором;

* такі ж вектори, розраховані раніше на етапі індексації, для всіх зображень товарів, що зберігаються в БД вже завантажені в RAM комп'ютера (оскільки для пошуку схожих потрібно буде проводити велику кількість переборів і попарних порівнянь, ми завантажили всю БД відразу в пам’ять, що дозволило підвищити швидкість пошуку в десятки разів, причому база приблизно в 100 тис. товарів поміщається не більше ніж у 2-3 ГБ оперативної пам'яті);

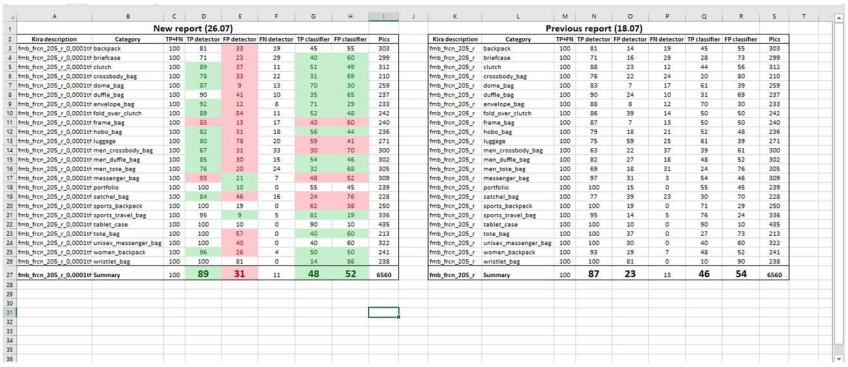
* з інтерфейсу або з властивостей приходять коефіцієнти пошуку для даної категорії, наприклад у категорії "сукня" ми шукаємо більше за кольором, ніж за формою (наприклад 8-к-2 пошук колір-форма), а в категорії "туфлі на підборах" шукаємо 1-к-1 форма-колір так як і форма та колір тут однаково важливі;

* далі вектори для кропів (фрагментів) із вхідного зображення попарно порівнюються із зображенням з бази з урахуванням коефіцієнтів (порівняється відстань Хеммінга між векторами);

* в результаті на кожен вирізаний фрагмент-товар формується масив схожих товарів з БД, також для кожного товару призначається вага (за простою формулою з урахуванням нормування, щоб усі ваги були в діапазоні від 0 до 1) для можливості виведення в інтерфейс, а також для подальшого сортування;

* науковий керівник

* масив схожих товарів виводиться в інтерфейс у вигляді web-JSON-API (рис. 3).



New report (26.07)

Kira description	Category	TP+FN	TP detector	FP detector	FN detector	TP classifier	FP classifier	Pics
fmb_frcn_205_r_0.0001it	backpack	100	81	33	19	45	55	303
fmb_frcn_205_r_0.0001it	briefcase	100	71	23	29	40	60	299
fmb_frcn_205_r_0.0001it	clutch	100	89	37	11	51	49	312
fmb_frcn_205_r_0.0001it	crossbody_bag	100	78	33	22	31	69	210
fmb_frcn_205_r_0.0001it	dome_bag	100	87	9	13	70	30	259
fmb_frcn_205_r_0.0001it	duffle_bag	100	90	41	10	35	65	237
fmb_frcn_205_r_0.0001it	envelope_bag	100	92	12	8	71	29	233
fmb_frcn_205_r_0.0001it	fold_over_clutch	100	89	54	11	52	48	242
fmb_frcn_205_r_0.0001it	frame_bag	100	83	13	17	40	60	240
fmb_frcn_205_r_0.0001it	hobo_bag	100	82	31	18	56	44	236
fmb_frcn_205_r_0.0001it	luggage	100	80	78	20	59	41	271
fmb_frcn_205_r_0.0001it	men_crossbody_bag	100	67	31	33	30	70	300
fmb_frcn_205_r_0.0001it	men_duffle_bag	100	85	30	15	54	46	302
fmb_frcn_205_r_0.0001it	men_tote_bag	100	76	20	24	32	68	305
fmb_frcn_205_r_0.0001it	messenger_bag	100	93	21	7	48	52	309
fmb_frcn_205_r_0.0001it	portfolio	100	100	10	0	55	45	239
fmb_frcn_205_r_0.0001it	satchel_bag	100	84	46	16	24	76	228
fmb_frcn_205_r_0.0001it	sports_backpack	100	100	19	0	62	38	250
fmb_frcn_205_r_0.0001it	sports_travel_bag	100	95	9	5	81	19	336
fmb_frcn_205_r_0.0001it	tablet_case	100	100	10	0	90	10	435
fmb_frcn_205_r_0.0001it	tote_bag	100	100	57	0	40	60	213
fmb_frcn_205_r_0.0001it	unisex_messenger_bag	100	100	40	0	40	60	322
fmb_frcn_205_r_0.0001it	women_backpack	100	96	26	4	50	50	241
fmb_frcn_205_r_0.0001it	wristlet_bag	100	100	81	0	14	86	238
fmb_frcn_205_r_0.0001it	Summary	100	89	31	11	48	52	6560

Previous report (18.07)

Kira description	Category	TP+FN	TP detector	FP detector	FN detector	TP classifier	FP classifier	Pics
fmb_frcn_205_r	backpack	100	81	14	19	45	55	303
fmb_frcn_205_r	briefcase	100	71	16	29	28	73	299
fmb_frcn_205_r	clutch	100	88	23	12	44	56	312
fmb_frcn_205_r	crossbody_bag	100	76	22	24	20	80	210
fmb_frcn_205_r	dome_bag	100	83	7	17	61	39	259
fmb_frcn_205_r	duffle_bag	100	90	24	10	31	69	237
fmb_frcn_205_r	envelope_bag	100	88	8	12	70	30	233
fmb_frcn_205_r	fold_over_clutch	100	86	39	14	50	50	242
fmb_frcn_205_r	frame_bag	100	87	7	13	50	50	240
fmb_frcn_205_r	hobo_bag	100	79	18	21	52	48	236
fmb_frcn_205_r	luggage	100	75	59	25	61	39	271
fmb_frcn_205_r	men_crossbody_bag	100	63	22	37	39	61	300
fmb_frcn_205_r	men_duffle_bag	100	82	27	18	48	52	302
fmb_frcn_205_r	men_tote_bag	100	69	18	31	24	76	305
fmb_frcn_205_r	messenger_bag	100	97	31	3	54	46	309
fmb_frcn_205_r	portfolio	100	100	15	0	55	45	239
fmb_frcn_205_r	satchel_bag	100	77	39	23	30	70	228
fmb_frcn_205_r	sports_backpack	100	100	19	0	71	29	250
fmb_frcn_205_r	sports_travel_bag	100	95	14	5	76	24	336
fmb_frcn_205_r	tablet_case	100	100	10	0	90	10	435
fmb_frcn_205_r	tote_bag	100	100	37	0	27	73	213
fmb_frcn_205_r	unisex_messenger_bag	100	100	30	0	40	60	322
fmb_frcn_205_r	women_backpack	100	93	19	7	48	52	241
fmb_frcn_205_r	wristlet_bag	100	100	81	0	10	90	238
fmb_frcn_205_r	Summary	100	87	23	13	46	54	6560

Рисунок 2. Приклад таблиці-звіту про якість детектора та класифікатора

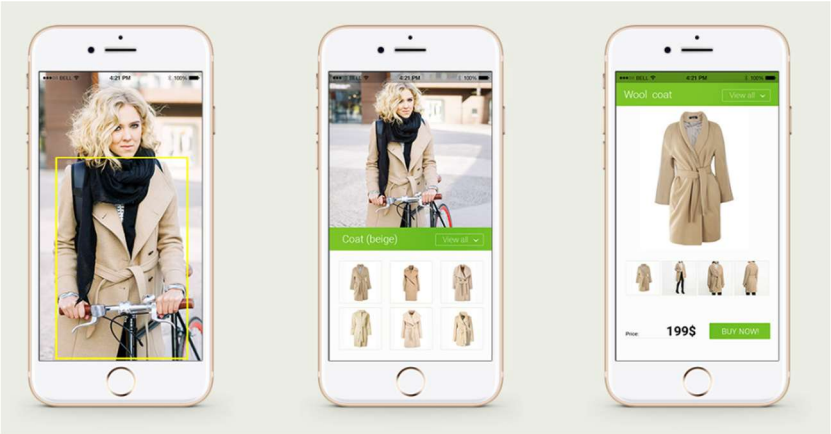


Рисунок 3. Інтерфейс мобільного застосунку

Висновок. Розроблена рекомендаційна система для персоналізованого підбору одягу, яка заснована на нейронних мережах за певних умов дає змогу здійснити пошук і створює рекомендації по певних групах товарів різної форми і кольору. Дана система представлена у форматі вебінтерфейсу та мобільного застосунку на яких можна змінювати параметри товару та характеристики пошуку.

Список використаних літературних джерел

[1] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In BMVC, 2021.

[2] S. Gidaris and N. Komodakis. Object detection via a multi-region & semantic segmentation-aware cnn model. In ICCV, 2021.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hier archies for accurate object detection and semantic segmentation. In CVPR, 2024.

* науковий керівник

Recommender system for personalized clothing selection using neural networks

Yaroslav Salabai, Ihor Kapran

Reviewed and analyzed recommendation systems for personalized clothing selection, their features, advantages and disadvantages. Identified and analyzed the main information and software tools necessary for the implementation of the task. The parameters of neural networks that influence the process of forming recommendations are studied. The software is implemented using the following technologies: R-FCN technology, ResNet neural network.

* науковий керівник