

Введение в рекомендательные системы

Нейросетевые рекомендательные
системы 2025/26, лекция 1

Кирилл Хрыльченко

Старший преподаватель,
ФКН ВШЭ

Яндекс



План занятия

На лекции:

1. Организационные вопросы
2. Информационная перегрузка
3. Что оптимизирует рекомендательная система
4. Как устроены рекомендательные алгоритмы
5. Зачем для рекомендаций нужны нейросети
6. Трансформеры для рекомендаций

На семинаре:

1. Оффлайн-метрики качества рекомендаций
2. Обработка данных

Организационные вопросы

Часть 1



Команда курса



Кирилл Хрыльченко
Преподаватель



Артём Матвеев
Ассистент



Владимир Байкалов
Ассистент



Елизавета Кононова
Менеджер

Кирилл Хрыльченко

Преподаватель и исследователь из
ФКН ВШЭ

- Окончил ВМК МГУ
- 5 лет занимался нейросетевыми рекомендательными системами в Яндексе
 - Руководил R&D-командой, занимавшейся нейросетевыми рекомендациями
- Первый автор статей на RecSys'25 и KDD'26
- Преподаю рекомендательные системы в ШАД и ВШЭ
- tg-канал @inforetriever



Артём Матвеев

Лид команды предобучения генеративных рекомендательных моделей в Яндексе

- Учится в магистратуре СКН ВШЭ
- Соавтор статей на RecSys'25 и KDD'26
- Выступал на Data Fest 2025 с докладом "Нужны ли графовые нейросети для кодирования товаров в рекомендациях?"



Владимир Байкалов

Ведущий исследователь в отделе
исследований VK

- Учится в аспирантуре ИТМО
- Первый автор статьи на WWW'24, соавтор
статьей на RecSys'25 и KDD'26
- Работал в Google и дважды работал в
Яндексе над рекомендациями
- Соавтор датасета Yambda, на котором вы
будете делать домашние задания
- tg-канал @ducks_rec
- Выступал на Дата ёлке 2026 с докладом
“Итоги года RecSys”



Цель курса

1. Передать вам наши знания про рекомендательные системы
2. Продемонстрировать применение математики, алгоритмов, ML/DL на практике
3. Научить заниматься исследованиями и R&D

Отличия от других курсов по рекомендательными системам

Рассказываем в первую очередь про свой опыт:

1. Большая часть занятий и домашних заданий – про нейросетевые рекомендации
2. Освещаем различные аспекты работы исследователя
3. На семинарах разбираем статьи, кейсы внедрений, решения соревнований
4. Другие курсы комплементарны – про классические модели, разработку, продуктовые аспекты

Программа курса

Более классическая часть курса:

1. Введение в рексистемы
2. Дизайн рексистем глазами исследователя
3. Двухбашенные модели для генерации кандидатов
 1. Архитектура
 2. Обучение
4. Трансформеры для рекомендаций
5. Ранжирование
 1. Кодирование признаков и их взаимодействий
 2. Архитектура

Программа курса

Продвинутый материал:

6. Векторная квантизация, semantic IDs, generative retrieval
7. Генеративные модели, LLM для рекомендаций, масштабирование рексистем
8. Графовые нейросети для рекомендаций
9. Обучение с подкреплением
 1. Бандиты и exploration; long-term оптимизация
 2. Off-policy learning & evaluation
10. Две гостевые лекции: industry case & research case

Домашние задания

1. Данные, метрики, бейзлайны (после второй лекции)
2. Двухбашенные нейросетевые модели для генерации кандидатов (после четвертой лекции)
3. Нейросетевое ранжирование (после шестой)
4. Трансформеры для рекомендаций (после восьмой)

Финальный проект – воспроизведение статьи из заранее выданного пула:

- Нужно также предложить модификацию и её реализовать
- В группах до трех человек
- Выдаём в середине курса (после седьмой лекции)

Финальная оценка

Финальная оценка = $0.4 * \text{ДЗ} + 0.3 * \text{ФП} + 0.3 * \text{Э}$, где

- ДЗ – средняя оценка за все домашние задания
- ФП – оценка за финальный проект
- Э – результат экзамена

Если $\text{накоп} = \text{Округление}((0.4 * \text{ДЗ} + 0.3 * \text{ФП}) / 0.7) \geq 6$ и
финальный проект защищен на 6 баллов и выше, то студент
может получить накоп в качестве итоговой оценки

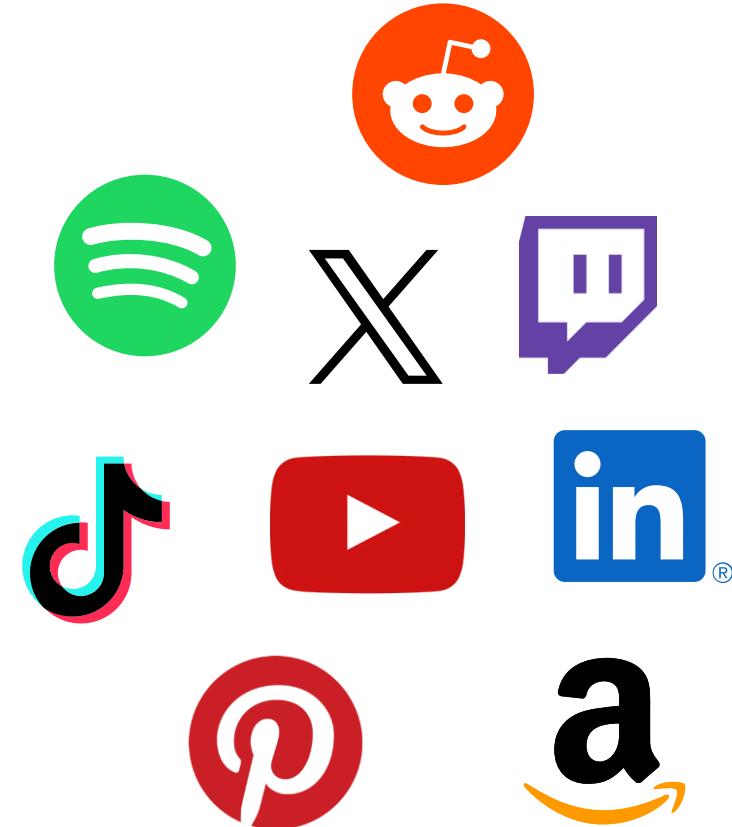
Информационная перегрузка

Часть 2



Цифровой мир = бесконечный каталог

- **Музыка:** >100 млн треков
- **Видео:** миллиарды роликов, десятки миллионов новых в день
- **Социальные сети:** сотни миллионов постов ежедневно
- **Маркетплейсы:** десятки-сотни миллионов товаров





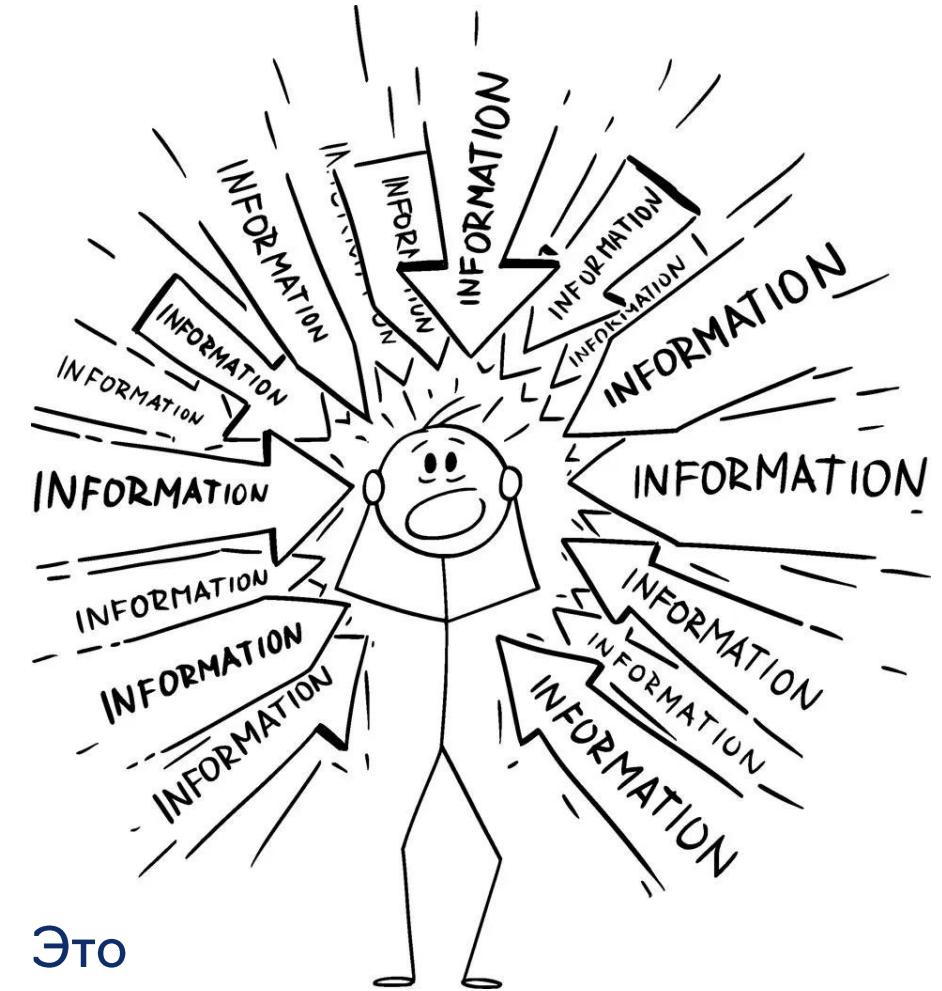
Масштабы YouTube

2022:

- 10 млрд видео
- 500 часов нового контента каждую минуту
- 1 млрд часов просмотров в день

Чтобы посмотреть всё – нужны десятки тысяч лет.

А ещё – 4% видео дают 94% всех просмотров.



Это

информационная
перегрузка

Тяжелый хвост айтемов

(элементы каталога,
которые мы рекомендуем)



Тяжелый хвост айтемов

(элементы каталога,
которые мы рекомендуем)



Возникновение рексистем

Рассмотрим три кейса:

- Netflix
- Социальные сети
- Spotify

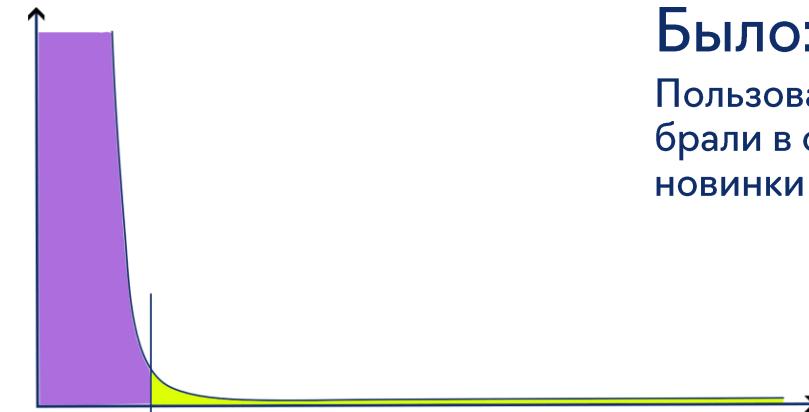


[RecSys 2014 Keynote by Neil Hunt: Quantifying the Value of Better Recommendations](#)

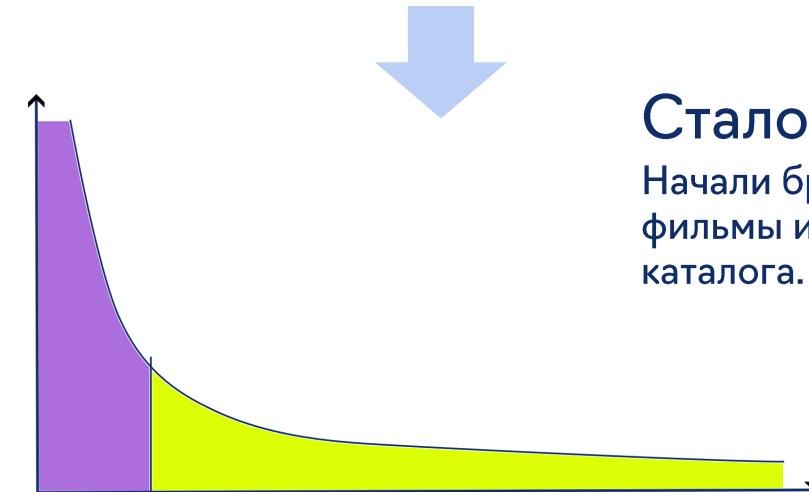
Рассылка DVD по почте:



Добавили персонализированные рекомендации на основе пользовательского фидбека (5-star ratings).



Было:
Пользователи
брали в основном
новинки



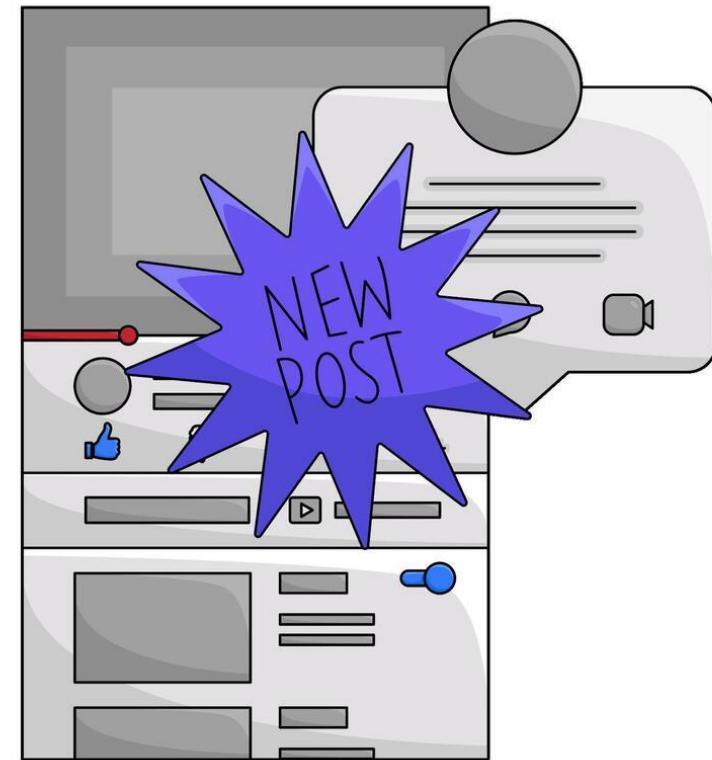
Стало:
Начали брать
фильмы из хвоста
каталога.

Социальные сети

Сначала: **News Feed** = хронологическая лента постов друзей.

Позже – алгоритмическая лента:

- учитывается **релевантность поста пользователю** (поведение и контент)
- Свежесть тоже влияет



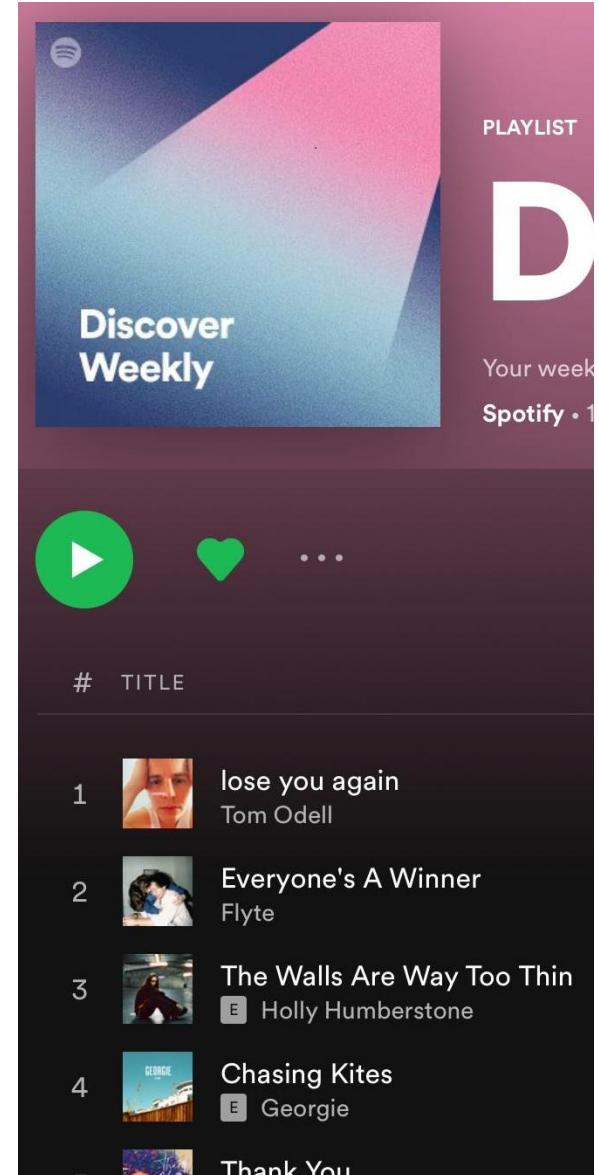
Это полноценная
рекомендательная
система



- Изначально (с 2008 года) – ручные плейлисты пользователей и редакторов
- В 2013 разработчик Erik Bernhardsson сколотил первую рекомендательную систему на основе матричной факторизации
- Через 2 года начали выпускать еженедельный плейлист **Discover Weekly**
- Через 3 месяца – 75 млн слушателей и 1.7 млрд прослушиваний



Discovery – помогаем пользователям находить новый контент, который они сами бы не нашли



Зачем рекомендательные системы авторам



Что оптимизирует рекомендательная система

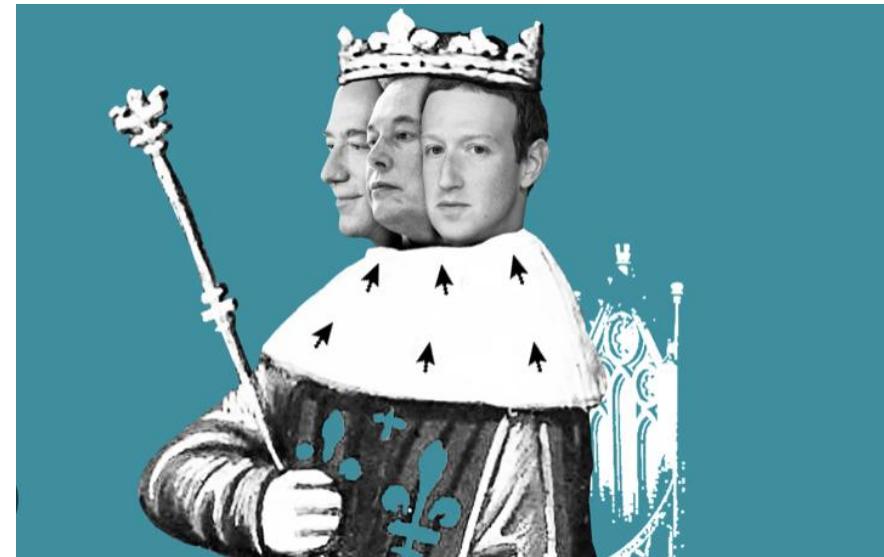
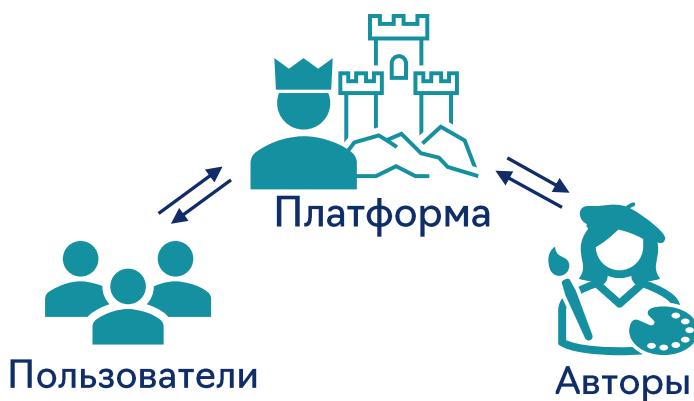
Часть 3



Платформа

Кроме пользователей и авторов контента, есть третья сторона (англ. *stakeholder*) – **платформа**:

- Разрабатывает рекомендательную систему
- Конечная цель – зарабатывать деньги (монетизация)



Технофеодализм:

- **Онлайн-платформы – феодалы:** контролируют цифровое пространство
- **Пользователи и авторы – крепостные:** предоставляют свои данные и труд в обмен на доступ к сервисам

Бизнес-модели плаформ

Как зарабатывают:

- Подписочные сервисы
- Маркетплейсы
- (spoiler)

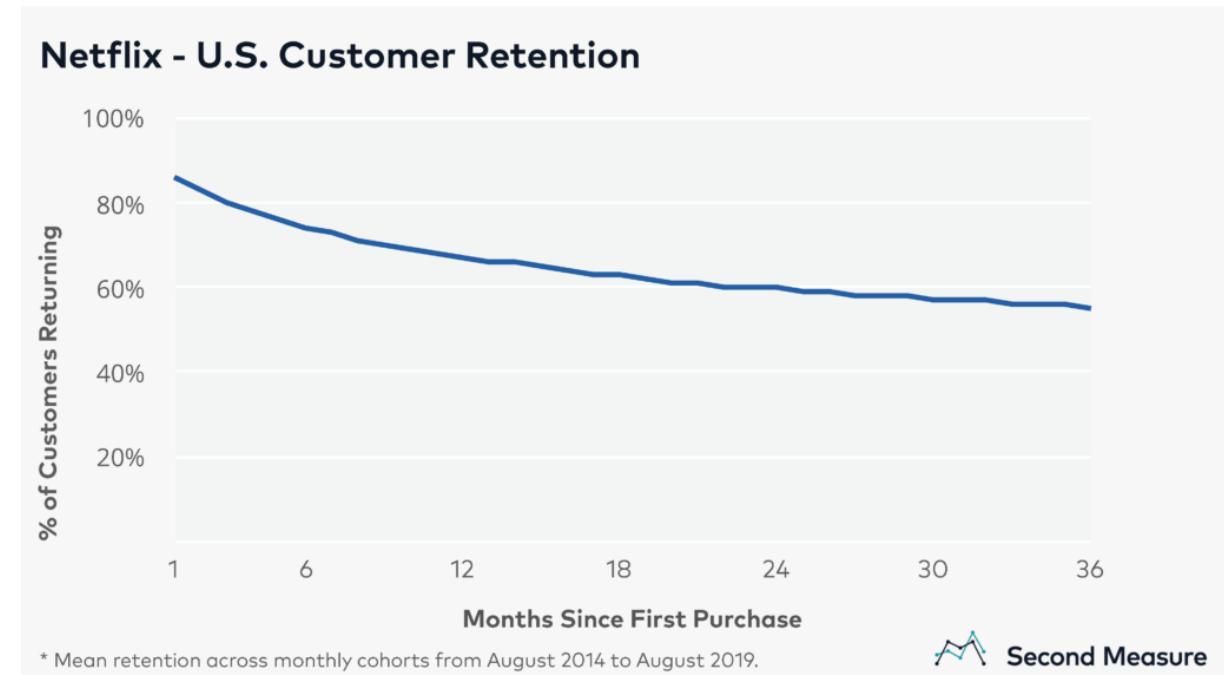
Подписочные сервисы

Цель платформы – предоставлять достаточно ценности, чтобы:

- Удерживать текущих подписчиков (retention)
- Привлекать новых



Проблема холодного старта
(пользователей): трудно угадывать интересы новых пользователей.



Маркетплейсы

Платформа зарабатывает за счет комиссии с каждой продажи товара.

Цель платформы – максимизировать GMV:

- Вместо лучших рекомендаций выдаем те, которые в среднем приносят больше денег
- $Score = P(\text{покупка}) \times \text{цена}$

Динамическое ценообразование – один и тот же товар стоит по-разному в зависимости от контекста.



a

\$790 млрд в год (GMV)

Alibaba Group

Taobao: \$790 млрд
Tmall: \$683 млрд

\$554 млрд

wildberries

\$7.2 млрд

Largest Internet Companies

List of the largest Internet companies [hide]										
Rank	Company	Revenue USD billions	F.Y.	Employees	Market cap. USD billions	Headquarters	Founded	Industry	Refs	
1	Amazon	\$620.12	2021	1,608,000	\$2,490	🇺🇸 Seattle	1994	Ecommerce	[1][2]	
2	Alphabet	\$339.85	2021	156,500	\$2,270	🇺🇸 Mountain View	1998	Internet	[3][4]	
3	JD.com	\$260.4	2021	385,357	\$390.35	🇨🇳 Beijing	1998	Ecommerce	[5][6]	
4	Ma	\$156.22	2021	71,970	\$1,803	🇺🇸 Menlo Park	2004	Social Media	[7][8]	
5	Alibaba	\$109.48	2021	251,462	\$330.67	🇨🇳 Hangzhou	1999	Ecommerce	[9][10]	
6	Tencent	\$87.85	2021	112,771	\$562.84	🇨🇳 Shenzhen	1998	Internet	[11][12]	
7	ByteDance	\$58	2021	110,000	\$353	🇨🇳 Beijing	2012	Social Media	[13][14]	
8	Netflix	\$39	2021	12,135	\$434.46	🇺🇸 Los Gatos	1997	Entertainment	[15][16]	
9	Meituan	\$27.77	2021	100,033	\$177.34	🇨🇳 Beijing	2010	Ecommerce	[17][18]	
10	PayPal	\$25.37	2021	30,900	\$220.26	🇺🇸 San Jose	1998	Financial Services	[19][20]	
11	Wildberries	\$22.26	2022	154,000	\$14.52	🇷🇺 Moscow	2004	Ecommerce		

Delightful Ads



in • 2nd
Senior Engineering Manager at Google Inc, improving YouTube ...
3mo • 

+ Follow ⋮

We're looking for passionate ML Engineers to join our team and revolutionize how users discover **delightful ads** on YouTube, directly impacting the success of small and medium businesses!

Are you skilled in building recommendation models with hands-on experience in information retrieval, transformers, and LLMs? Do you thrive on challenging tasks that blend cutting-edge technology with real-world impact?

If you're ready to make a difference and help businesses thrive, we want to hear from you!

Apply here: <https://lnkd.in/ga3BAJUT> 

DM me if you are already past the interview loop.

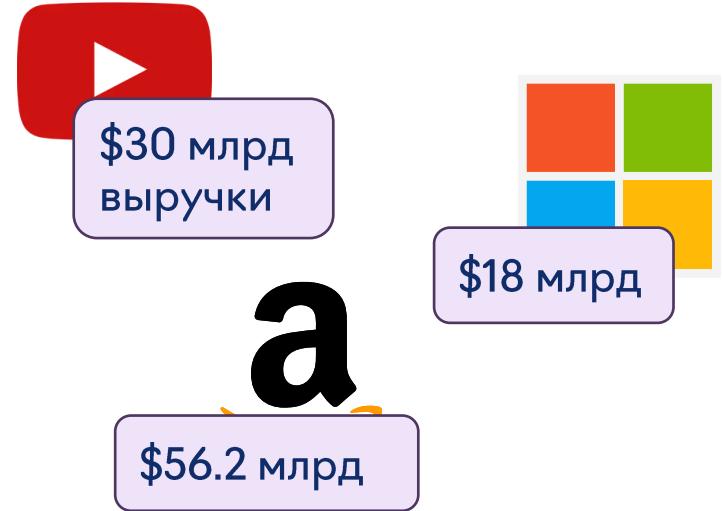
#MachineLearning #MLEngineer #YouTubeAds #Recommendations #Hiring
#CareerOpportunity #AI #LLM #Transformers

Delightful для
кого?

Монетизация через рекламу

YouTube:

- Во время просмотра видео, пользователю показывается реклама
- Чем больше watch time, тем больше показов рекламы



Больше пользовательской активности
→ больше рекламы → больше денег.



Монетизация поиска: показы
рекламных документов рядом
с "органикой".

Google Search: \$198 млрд выручки

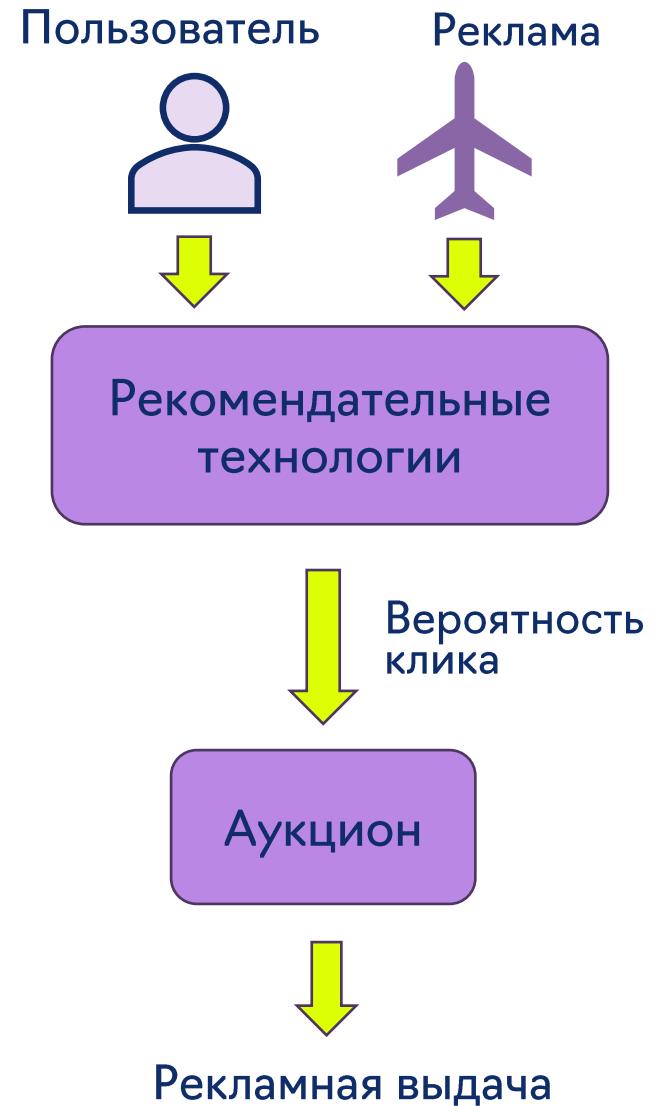
Персонализация рекламы

Рекламу показывают персонализированно:

- Чтобы меньше портить пользователю experience
- Чтобы она приносила больше пользы рекламодателям (и они продолжали за неё платить)

Для рекламы используются те же самые рексистемы, но:

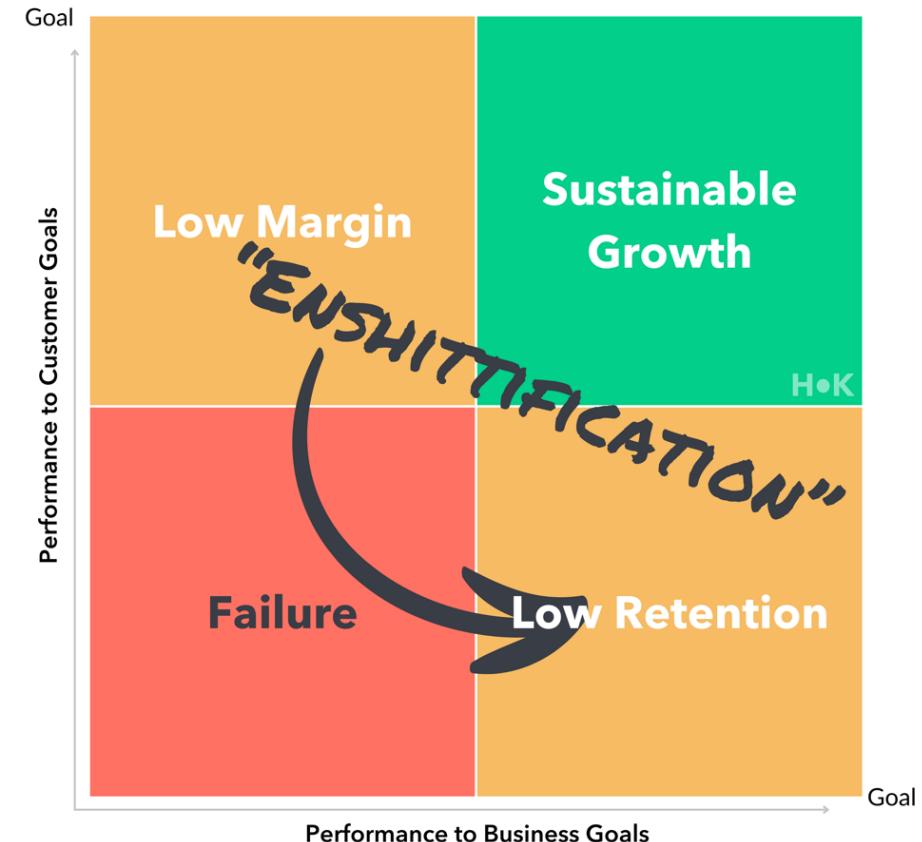
- Есть аукцион – учитываем сколько готов заплатить рекламодатель



Enshittification

Три стадии развития онлайн-платформ по Cory Doctorow:

- Изначально компания фокусируется на ценности для пользователей
- Затем фокус смещается на авторов / продавцов / рекламодателей
- Когда пользователям и авторам уже сложно сменить платформу, максимизируется прибыль для инвесторов



Google



Изначальный девиз Google:

- **Don't be evil**

После корпоративной реструктуризации:

- **Do the right thing**

Telegram



Free

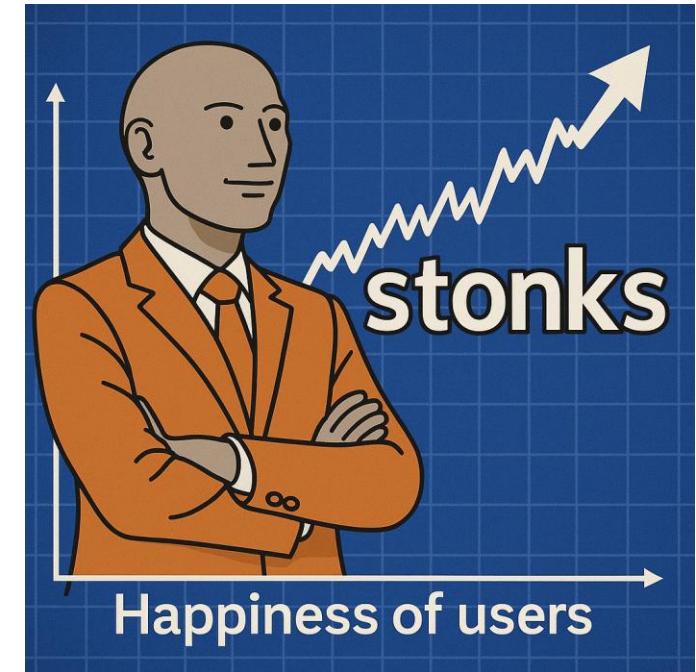
Telegram is free forever. No ads.

No subscription fees.

Счастье пользователей и авторов как долгосрочная метрика успеха

Если приносить пользователям и авторам ценность, деньги придут сами. Вместо краткосрочной прибыли, нужно:

- максимизировать **счастье пользователей**
- давать шанс всем авторам быть рекомендованными, включая менее популярных и новых



Самое ценное у рексистемы – это пользователи и авторы. Их гораздо сложнее приобрести, чем обеспечить монетизацию.



Монетизация – **следствие** хорошего продукта, а не его **причина**.

Как устроены рекомендательные алгоритмы

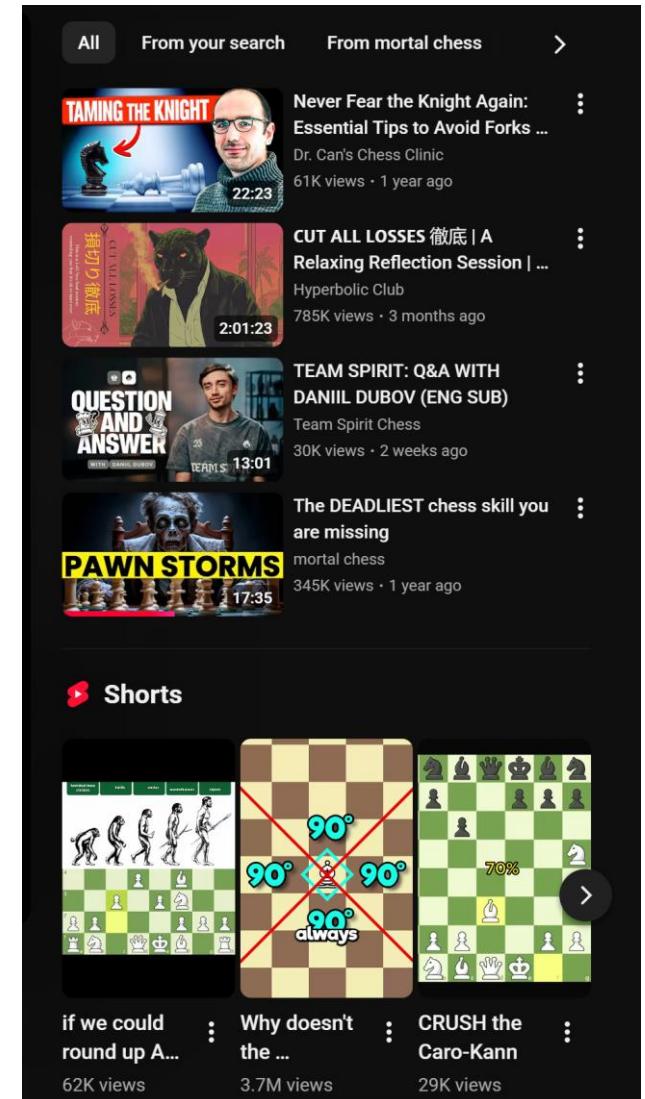
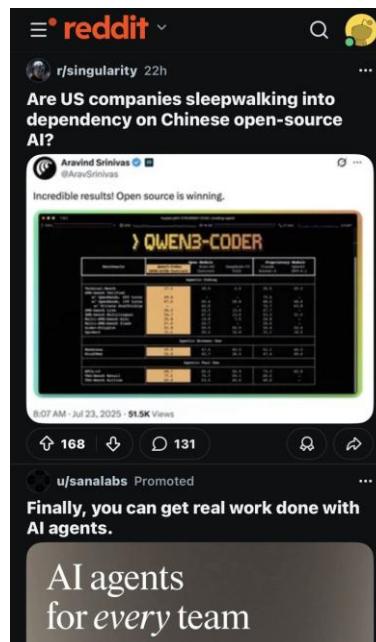
Часть 4



Результат работы рекомендательной системы

- Следующий трек / видео / пост
- Лента новостей или рекомендаций
- Список похожих товаров
- Смесь разных форматов контента (и рекламы)

Интерфейсы разные, ранжирование
одинаковое (почти).



Рекомендательная система – это сортировка



- Рекомендательная система считает функцию сортировки $f(u, c, i)$
- По значению f ранжирует весь каталог
- Вид f зависит от целей рекомендательной системы:
 - Вероятность клика / лайка
 - Вероятность покупки, помноженная на цену
 - Ожидаемое время просмотра

Рекомендательная система – это сортировка



Контекст:

- Общий контекст – время, устройство, браузер, пользовательские настройки
- В рекомендациях похожих товаров – товар, на карточке которого находимся
- В поиске – поисковый запрос
- В рекламе – веб-страница, на которую зашёл пользователь

Скорость рекомендаций



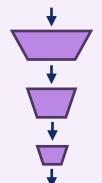
Рекомендации должны формироваться за
десятки-сотни миллисекунд.

Многостадийные рексистемы



Фокус на скорость и эффективность – эвристики и хитрые структуры данных (поиск ближайших соседей, обратный индекс), легкие модели

Тяжелые модели, использующие **максимум** информации



Многостадийность – это факторизация вычислений ради скорости.



Watch to watch next

Классический рекомендательный стек на примере YouTube до 2016 года – две стадии:

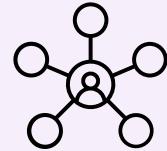
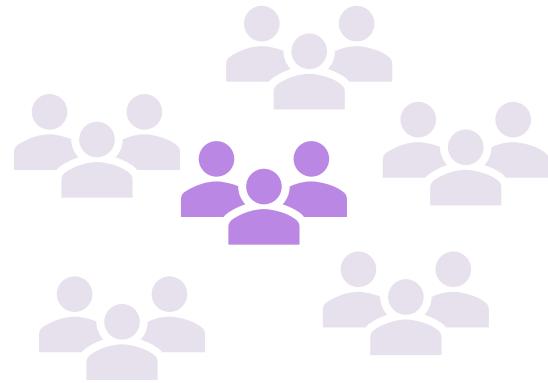
1. **Генерация кандидатов:** матричная факторизация
2. **Ранжирование:** линейная модель, предсказывающая expected watch time

Для того времени – это уже достаточно сложная рекомендательная система.

Коллаборативная фильтрация

Начнём с самых простых рекомендательных алгоритмов:

- Предлагаем пользователю то, что понравилось похожим пользователям
- В жизни: спрашиваем друзей, «что посмотреть/прочитать?»
- Рекомендательная система делает то же самое для миллионов пользователей

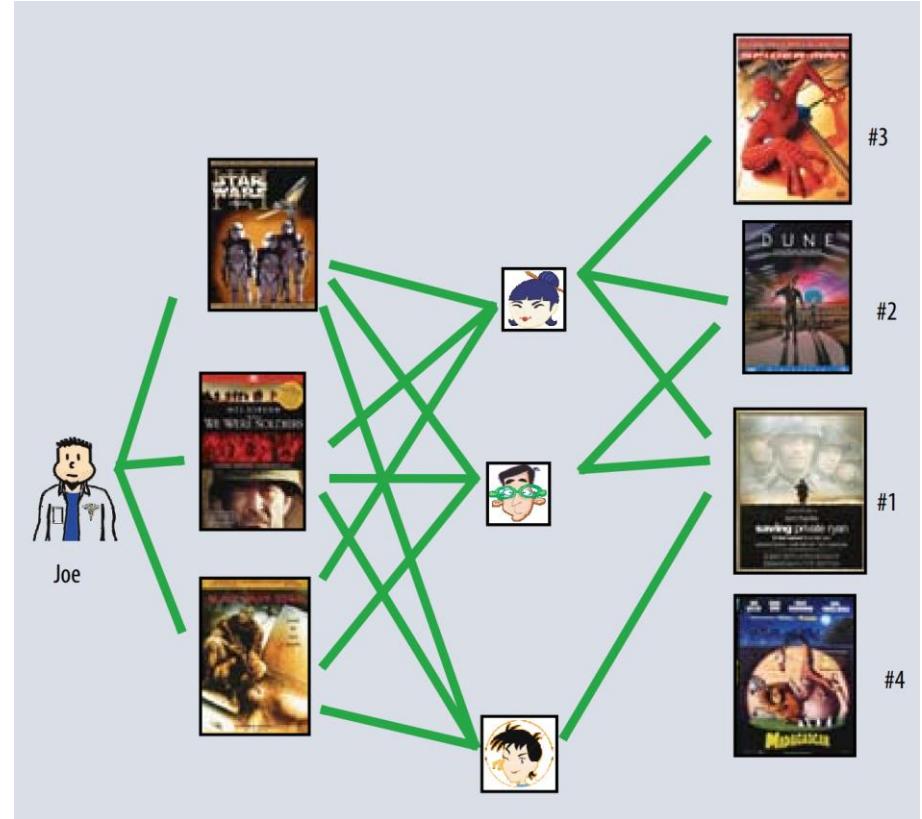


Коллаборативность – используем фидбек **всех** пользователей для рекомендаций **каждому**.

Как это работает (user-based CF)

- Смотрим, какие фильмы/песни/товары вам понравились
- Находим пользователей с похожими вкусами
- Берём то, что им понравилось, а вы ещё не видели
- Рекомендуем это вам

“Users like you also enjoyed X”



[Matrix Factorization Techniques for Recommender Systems](#),
Yehuda Koren et al

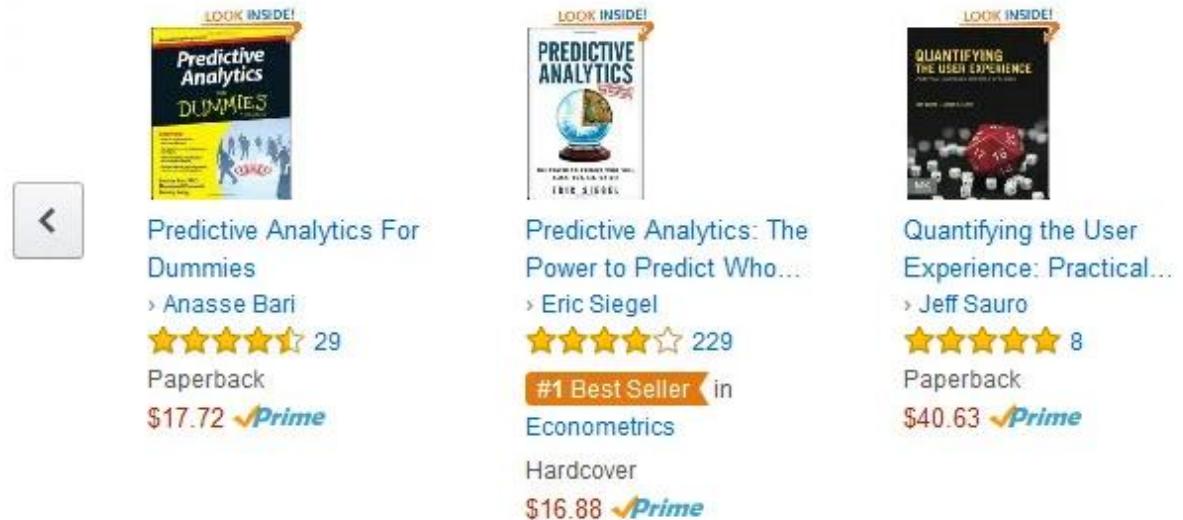
Item-based CF: похожие айтемы

Можно смотреть не на похожих людей, а на похожие айтемы.

Как оценивать похожесть айтемов:

- **По содержимому – описания, жанры, тэги**
- **По поведению пользователей (коллаборативный сигнал):**
 - “80% слушавших трек X слушали и трек Y” → значит, X и Y похожи

Customers Who Bought This Item Also Bought



Холодный старт (айтемов):
сложно рекомендовать новинки,
потому что для них нет
пользовательского фидбека.

Как это видят алгоритмы: user-item матрица

- Строим большую матрицу релевантности R :
 - Строки – пользователи
 - Столбцы – айтемы
 - Элементы – user-item взаимодействия
- **User-based CF:** ищем похожие строки, переносим знания в пропуски
- **Item-based CF:** ищем похожие столбцы, используем их для рекомендаций

Айтем

Рейтинг

Пользователь

			Айтем	Рейтинг			
				3	1	5	4
			3	1	5	5	4
			•	4	•	•	•
			•	•	2	3	1
			•	•	•	•	•
1	2	•	•	•	•	•	•
3	•	•	•	•	4	3	•
•	2	•	5	1	•	•	•
•	•	5	•	•	•	•	•

User-item interaction matrix

Разреженность

User-item матрица очень разреженная: 99+% элементов – нули.

Проблемы:

- Похожие пользователи могли ни разу не пересечься по лайкам → user-CF не видит их схожесть
- «Rich gets richer, poor gets poorer»:
 - Популярные айтемы часто встречаются в user-item матрице → их много рекомендуем
 - Хвостовые айтемы почти не видим → системно их недооцениваем



Netflix Prize:
99.82% sparsity



99.9999%
sparsity

Матричная факторизация

Представим user-item матрицу как произведение двух низкоранговых матриц:

	•	4	•	•	3
	•	•	2	3	1
	•	•	•	•	5
1	2	•	•	•	•
3	•	•	•	4	
•	2	•	5	1	
•	•	5	•	•	

≈

0.4	4.1	1.3
5.8	1.3	2.5
4.4	9.8	0.8
1.9	2.2	7.4
3.0	5.4	3.6
8.1	2.6	9.9
8.3	0.5	5.4

User Embeddings

×

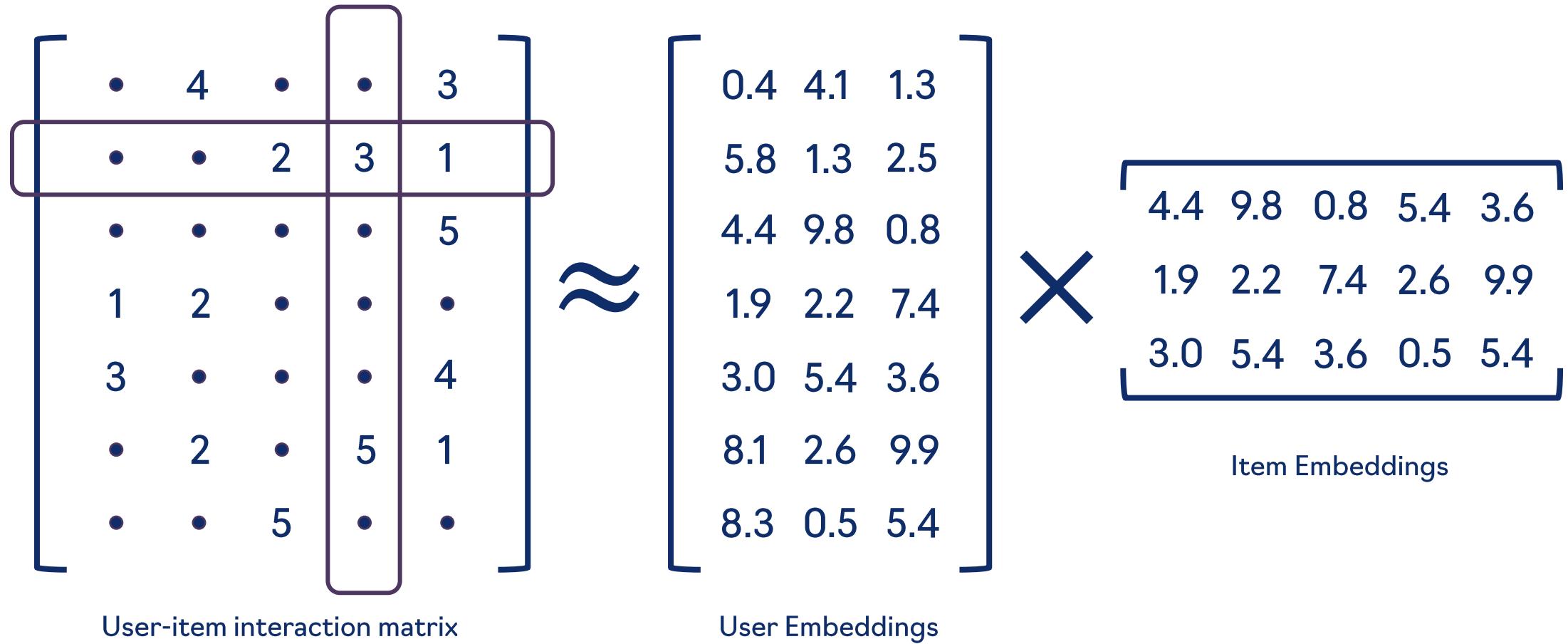
4.4	9.8	0.8	5.4	3.6
1.9	2.2	7.4	2.6	9.9
3.0	5.4	3.6	0.5	5.4

Item Embeddings

User-item interaction matrix

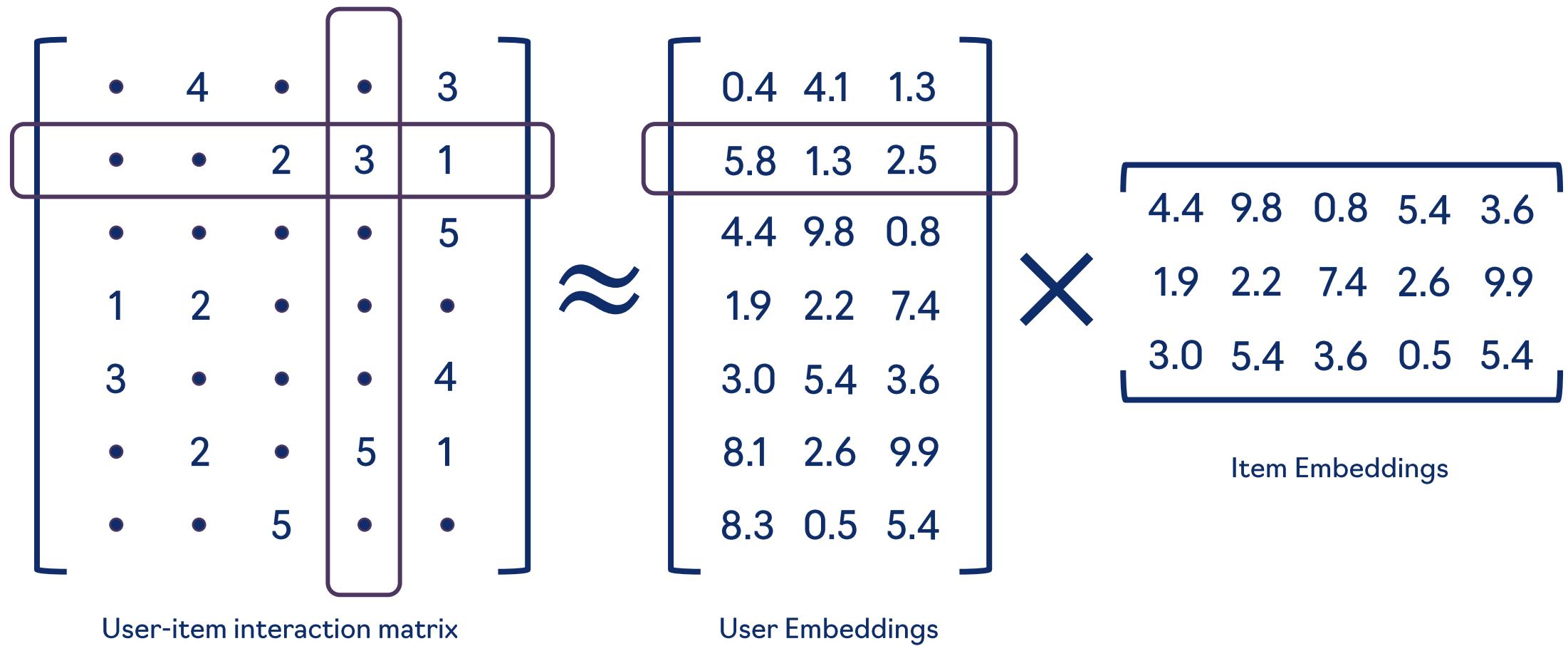
Матричная факторизация

Представим user-item матрицу как произведение двух низкоранговых матриц:



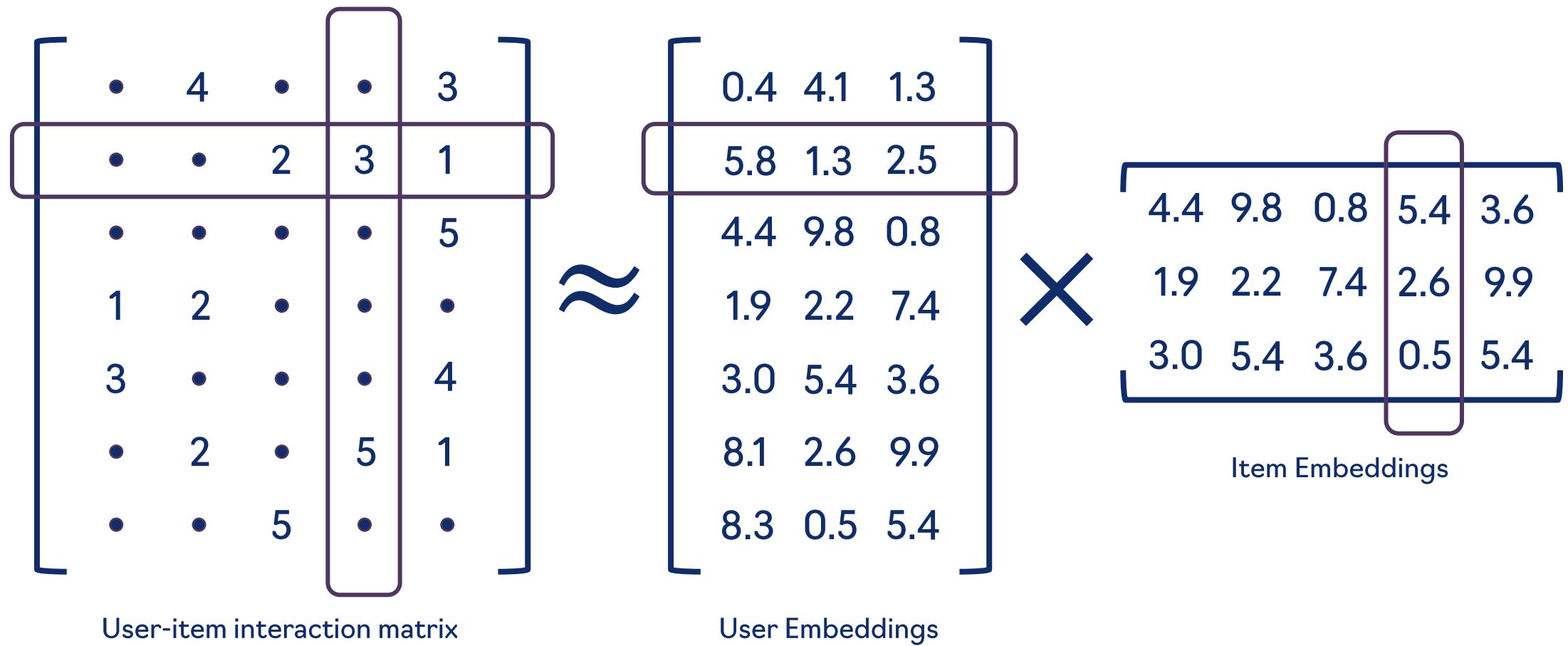
Матричная факторизация

Представим user-item матрицу как произведение двух низкоранговых матриц:



Матричная факторизация

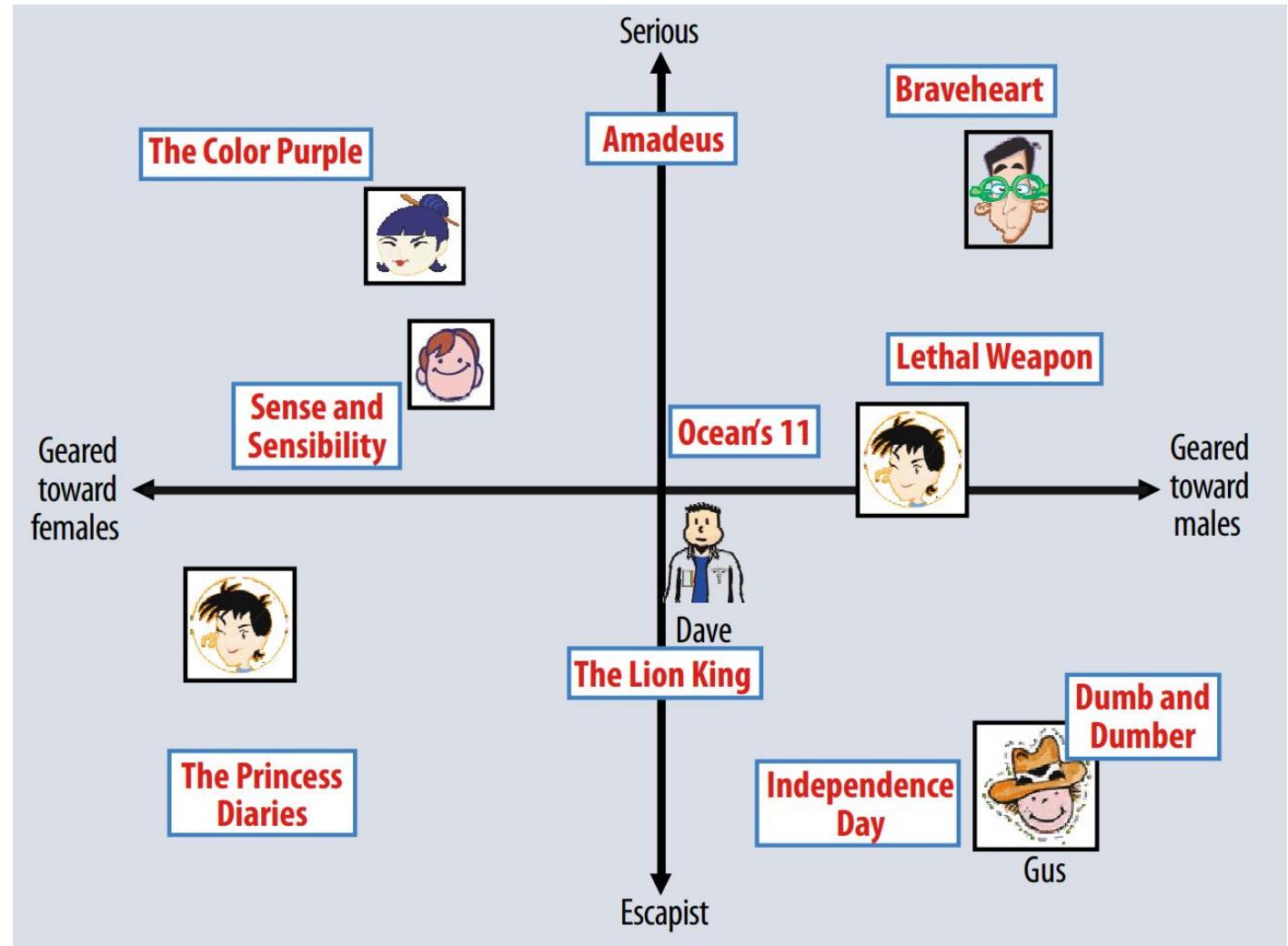
Представим user-item матрицу как произведение двух низкоранговых матриц:



Интерпретируемость матричной факторизации

Каждый элемент векторного представления отражает какую-то характеристику:

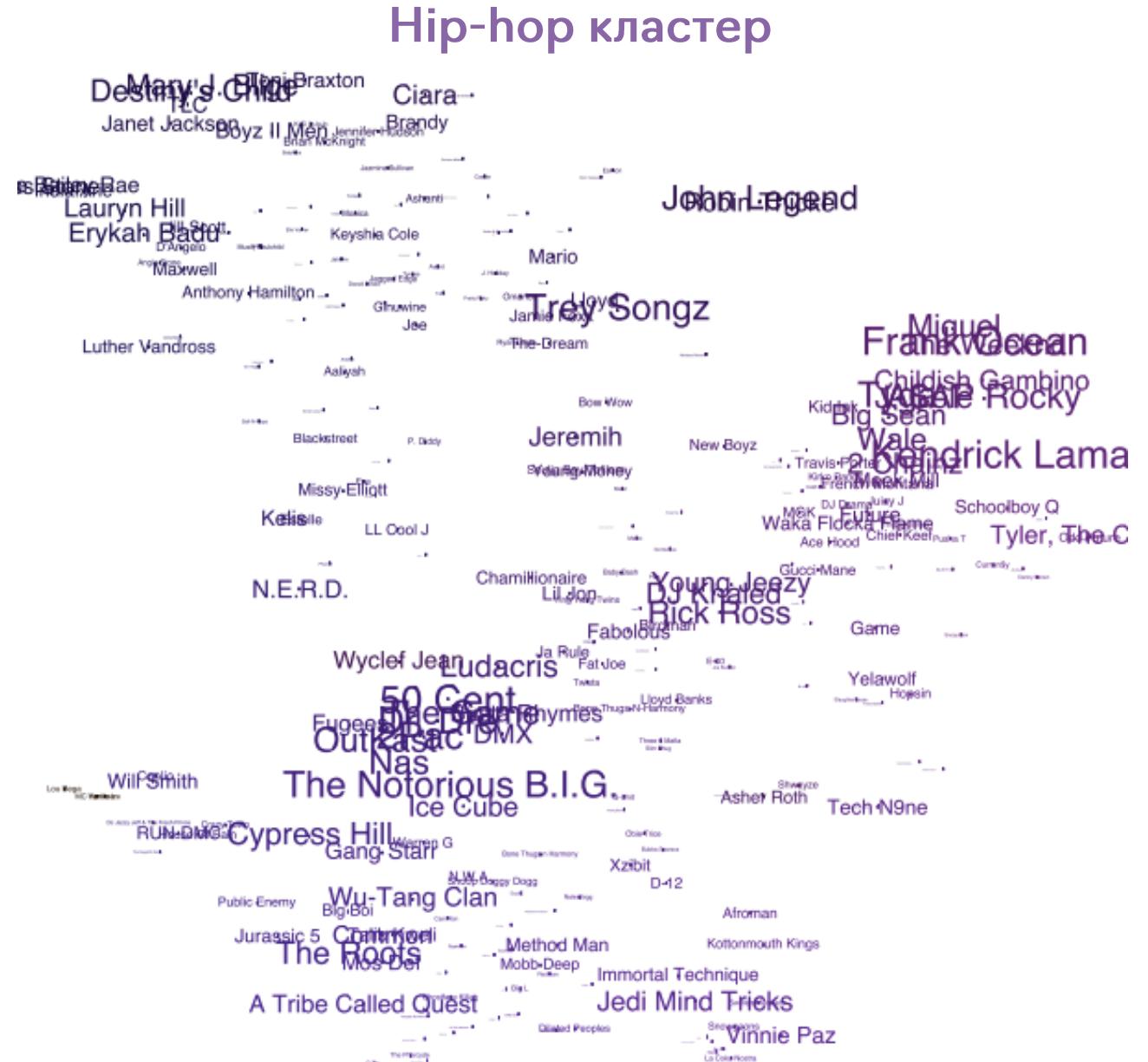
- Например, пользователь любит ужастики
- Для айтема – та же координата подсвечивает, является ли он ужастиком
- Элементы “совпадают” – высокая оценка





Вспомним Эрика из Spotify:

- Построил user-item матрицу прослушиваний треков
- С помощью матричной факторизации получил векторы длины 40



Рекомендации с помощью матричной факторизации

- Для всех пользователей и айтемов есть эмбеддинги (векторные представления), полученные из матричной факторизации
- Можем найти для пользователя айтемы с максимальным значением скалярного произведения (MIPS – maximum inner product search)
- Выдать их в качестве рекомендаций или заиспользовать как кандидатов для следующий стадий “рекомендательной воронки”

Преимущества матричной факторизации

- Побороли разреженность – у пользователей не пересекавшихся по айтемам могут быть похожие векторы
- Тяжелому хвосту стало лучше
- Научились находить абстрактные “вкусы” пользователей и характеристики айтемов
- Сжали информацию из огромной user-item матрицы в две небольших матрицы:
 - Описываем объекты небольшим количеством чисел вместо широких разреженных векторов
 - Подход хорошо масштабируется

Netflix Prize

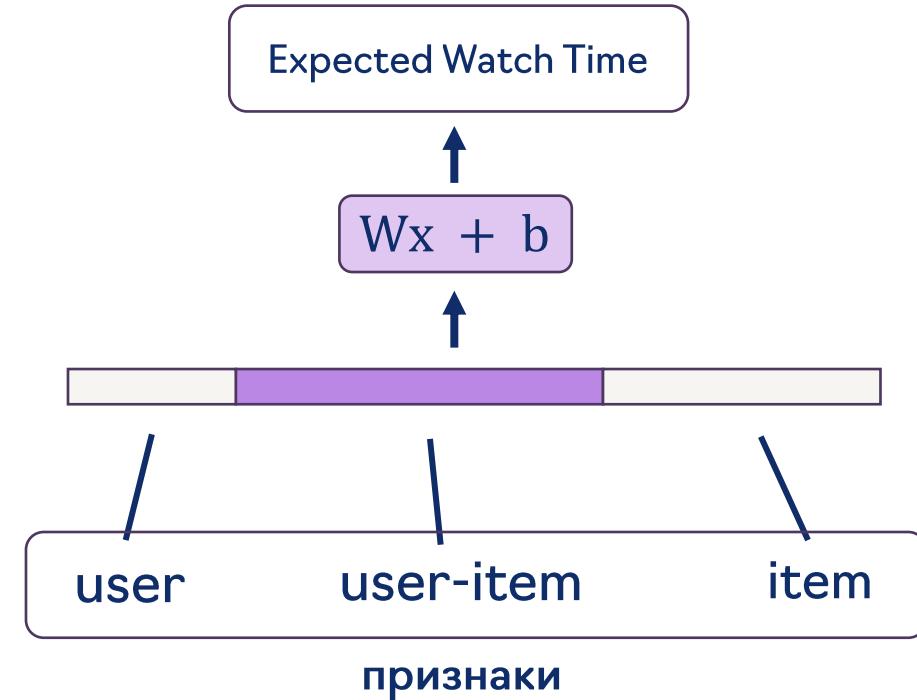
- В 2006 году Netflix опубликовали датасет с 100 млн проставленных фильмах оценок;
 - 480 тысяч клиентов, 17 тысяч фильмов
- Цель соревнования: улучшить точность относительно алгоритма Netflix на 10%
- Приз: 1 млн долларов
- Матричная факторизация – ключевой подход для победы



№	Команда	СКО	Улучшение в %	Время отправки
1	The Ensemble	0.8553	10.10	2009-07-26 18:38:22
2	BellKor's Pragmatic Chaos	0.8554	10.09	2009-07-26 18:18:28

Линейная модель ранжирования

- Цель: показывать видео, которые будут смотреть дольше
- Модель предсказывает **expected watch time**
- **Признаковое пространство:**
 - признаки пользователя
 - признаки видео
 - user-item признаки (например, наличие подписки на автора)



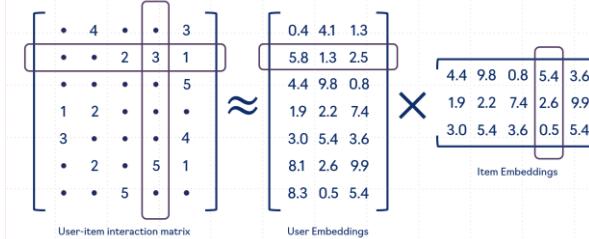


Watch to watch next

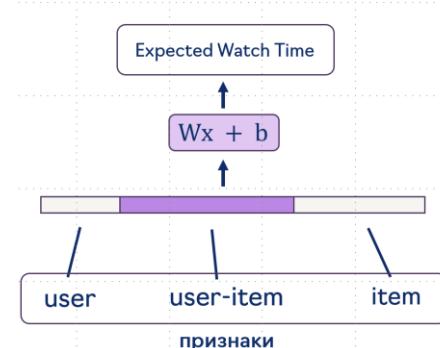
Классический рекомендательный стек на примере YouTube до 2016 года – две стадии:

1. Генерация кандидатов: матричная факторизация
2. Ранжирование: линейная модель, предсказывающая expected watch time

Отбираем тысячи кандидатов с помощью матричной факторизации



Ранжируем их с помощью линейной модели и берем top 10



Зачем для рекомендаций нужны нейросети

Часть 5



Зачем нужны нейросети

- У нас очень много данных – триллионы user-item взаимодействий
 - Классическому ML столько не нужно
- В других областях DL победил
- В рекомендациях есть свои челленджи:
 - холодный старт и тяжелые хвосты
 - distribution drift



Bitter lesson (Richard Sutton)

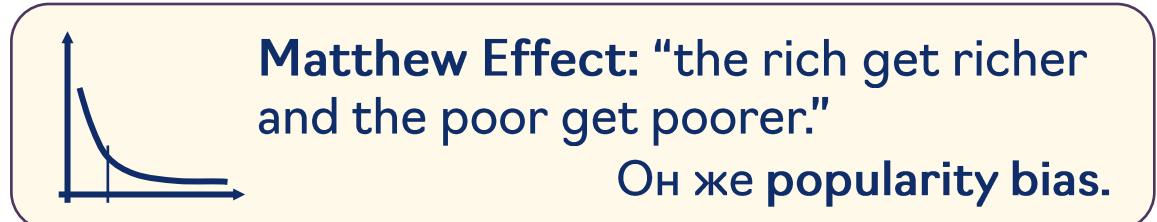
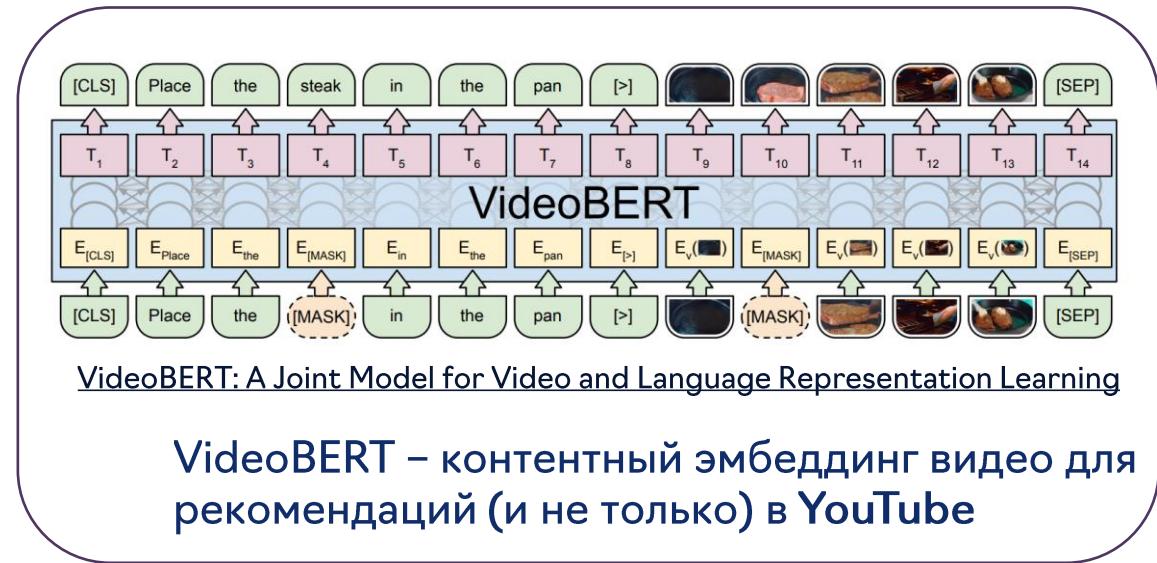
“The biggest lesson that can be read from 70 years of AI research is that general methods that leverage computation are ultimately the most effective, and by a large margin.”



Scaling hypothesis – чем больше данных и сама модель, тем лучше результат.

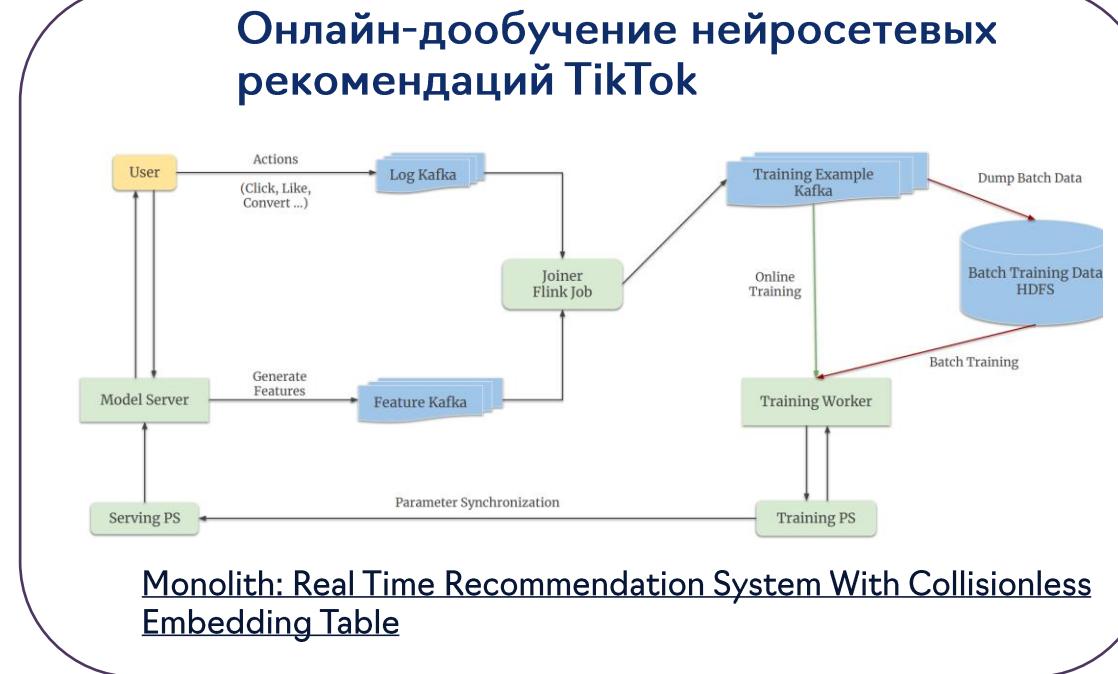
Холодный старт и тяжёлый хвост

- В новостях, рекламе, маркетплейсах каталог постоянно обновляется:
 - для новых айтемов нет фидбека
- Нужен content understanding:
 - видео/текст/картишка → эмбеддинг
- Тяжелый хвост:
 - популярные айтемы “забивают” обучение
 - у контентных признаков хвост легче, больше шанс вытянуть объекты из хвоста



Distribution Drift

- Распределения постоянно меняются:
 - появляются новые тренды, обновляется каталог
 - меняются интересы пользователей
- Нужны алгоритмы, которые постоянно дообучаются на новых данных



Feedback Loop



- Модель обучается на данных, которые породила
- Данные отражают её плохие особенности
- Обучение на этих данных эти особенности усиливает

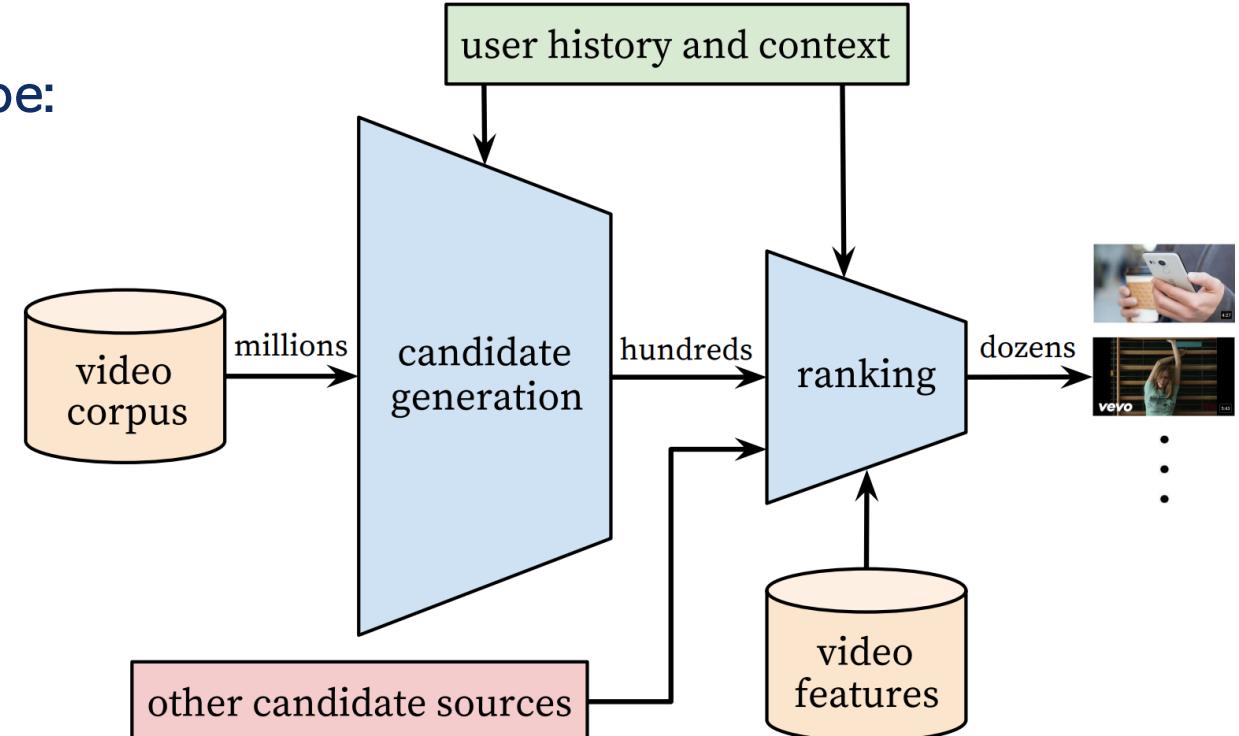
YoutubeDNN

В 2016 году Google представили нейросетевые рекомендации в Youtube:

- **Двухбашенная нейросеть для генерации кандидатов**
- **Полносвязная нейросеть для ранжирования**

До этого:

- **Матричная факторизация** для генерации кандидатов
- **Линейная модель** для ранжирования

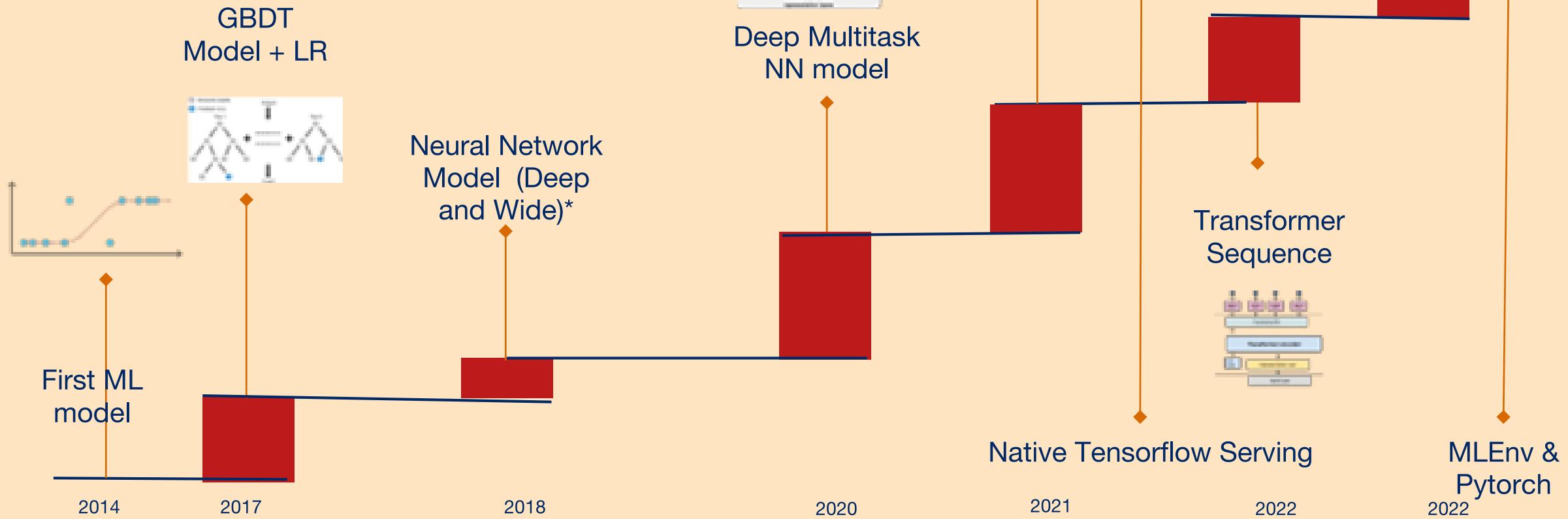


[Deep Neural Networks for YouTube Recommendations](#)

Ветки развития рексистемы

- Улучшение моделей для генерации кандидатов
- Улучшение ранжирующих моделей
- Концептуально другие подходы
 - делаем больше или меньше стадий в рексистеме
 - меняем интерфейс рекомендаций
- Про всё это поговорим в рамках курса

Эволюция нейросетевого ранжирования в Pinterest



Из выступления [Evolution of Ads Ranking at Pinterest by Aayush Mudgal](#)

Трансформеры для рекомендаций

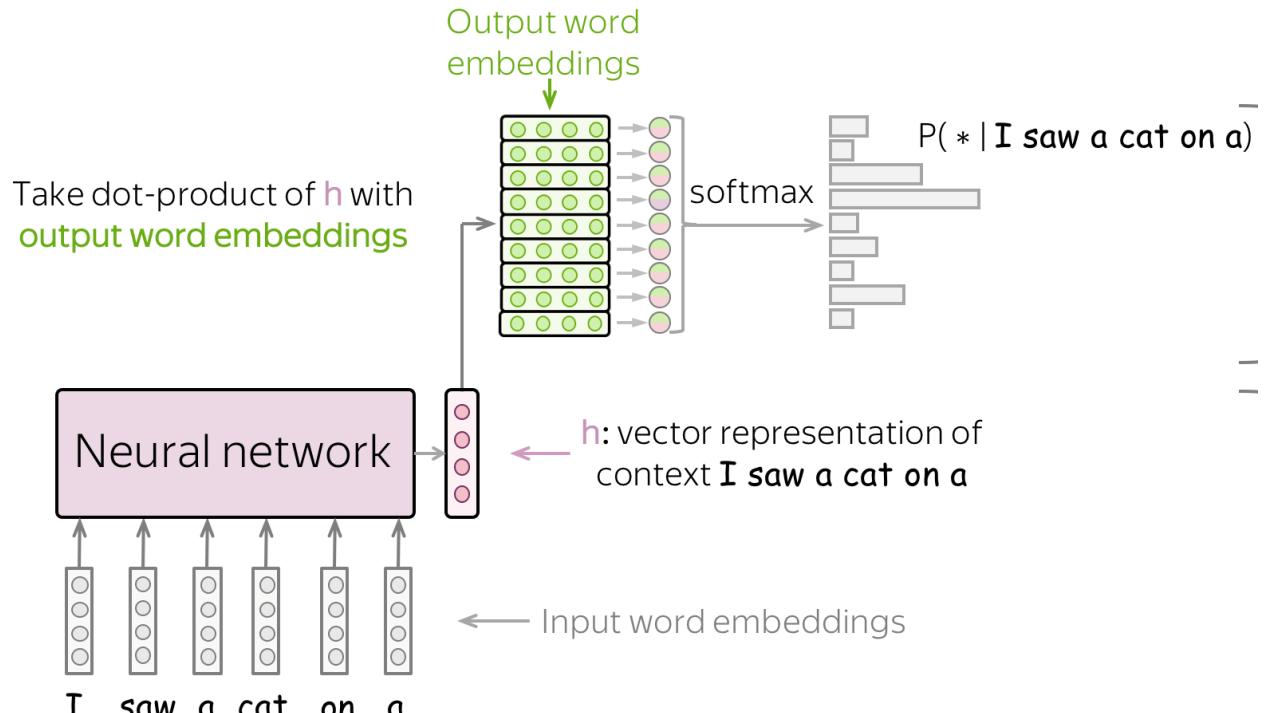
Часть 6



Языковые модели

ChatGPT, Gemini и другие
большие языковые модели учатся
**предсказывать следующее
слово в тексте:**

- Extreme multi-task learning:
учим грамматику, логику,
факты, здравый смысл, даже
математику и рекомендации



Из Lena Voita's [YSDA NLP Course](#)



LLM – это модель мира: она
учится понимать мир,
продолжая тексты.

Next Item Prediction

Заменим слова на действия пользователей:

- было (NLP): “Я пошёл в [token1] [token2] ...”
- стало (RecSys): “Пользователю понравились [track1] [track2] ...”

С помощью задачи
предсказания следующего айтема
 обучаем генеративную модель пользователя.



В отличие от NLP, пространство токенов очень большое:

- NLP: $10^4 - 10^5$, RecSys: $10^6 - 10^9$ токенов
- больше похоже на генерацию следующего кадра видео, чем на текст



Это модель мира! В которой мир – это пользователь.

Трансформеры в Яндекс Музыке



Уже есть и новые модели, смотрите
[выступление Петра Зайделя на PML Conf.](#)

Трансформеры в Яндекс Музыке

Внедрение	Длина истории	Конфигурация	Параметры	TLT	Like Likelihood
Offline V1	512	L6 H512	18.9M	+0.52%	+1.11%
Offline V2	1024	L6 H512	18.9M	+1.00%	+0.73%
Offline V3	1024	L6 H512	18.9M	+0.73%	+5.00%
Real-time V1	1024	L4 H256	3.2M	+0.32%	+1.38%
Offline V4 (ARGUS)	8192	L10 H1024	126.0M	+2.26%	+6.37%

Summary

Обсудили:

1. Тяжелые хвосты и возникновение рекомендаций в Netflix, Spotify, соцсетях
2. Подписочные сервисы, маркетплейсы, реклама
3. Многостайдийность, коллаборативную фильтрацию, матричную факторизацию, линейные модели
4. Зачем нужны нейросети для рекомендаций
5. Трансформеры

На семинаре

1. Как оценивать качество рекомендательной системы в оффлайне, до А/Б тестов
2. Как выглядят рекомендательные данные на практике и как мы с ними работаем для обучения моделей