

Двухбашенные нейросети

Нейросетевые рекомендательные
системы 2025/26, лекция 3

Кирилл Хрыльченко

Старший преподаватель,
ФКН ВШЭ

Яндекс

План занятия

На лекции:

1. Введение в двухбашенные модели
2. Софтмакс модель
3. Функции близости и температура
4. Сэмплирование негативов

На семинаре:

1. Рекомендательные системы глазами исследователя
2. Поиск информации и чтение статей

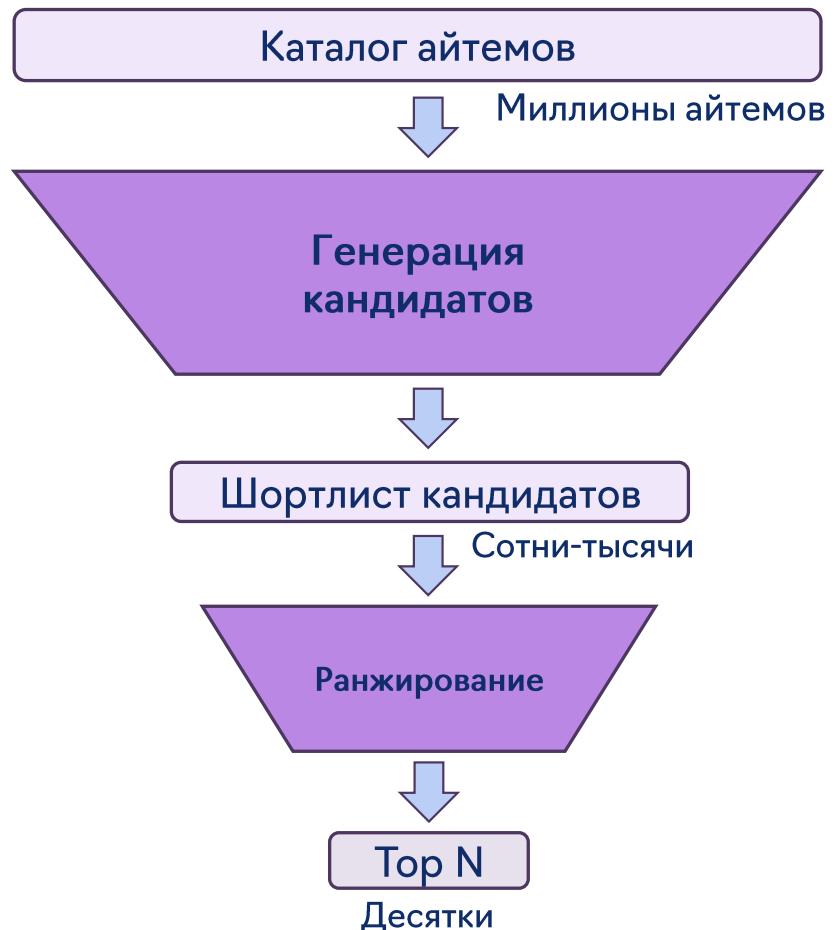
Введение в двухбашенные модели

Часть 1

Многостадийные рексистемы

Рекомендации – многостадийный процесс:

- На ранних стадиях – более быстрые и эффективные алгоритмы
 - Дальше – более тяжелые модели



Классические генераторы кандидатов

- **Топ популярного:** нет персонализации, не рекомендует новые айтемы
- **История пользователя:** нет discovery
- **Item-to-item:** мало персонализации и discovery, не учитывает контекст
- **User-to-item:** плохо работает на новых пользователях; не рекомендует новые айтемы
- **Матричная факторизация:** плохо для новых айтемов и пользователей, не учитывает контекст
- У всех методов кроме item-to-item – плохое качество на тяжелом хвосте

Нужны нейросети!

Двухбашенные модели

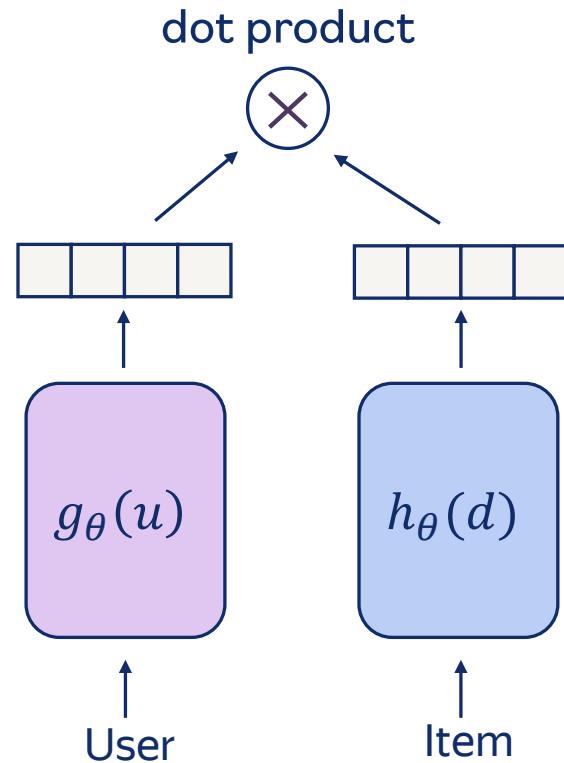
- Пользователи и айтемы раздельно кодируются в эмбеддинги:
 $g_\theta(u), u \in \mathcal{U}$ и $h_\theta(d), d \in \mathcal{D}$
- Позднее связывание: мера похожести – скалярное произведение

$$s_\theta(u, d) = \langle g_\theta(u), h_\theta(d) \rangle$$



В 2013 году появился DSSM – первая двухбашенная нейросеть для информационного поиска.

1. [Learning deep structured semantic models for web search using clickthrough data](#)
2. [Deep Neural Networks for YouTube Recommendations](#)

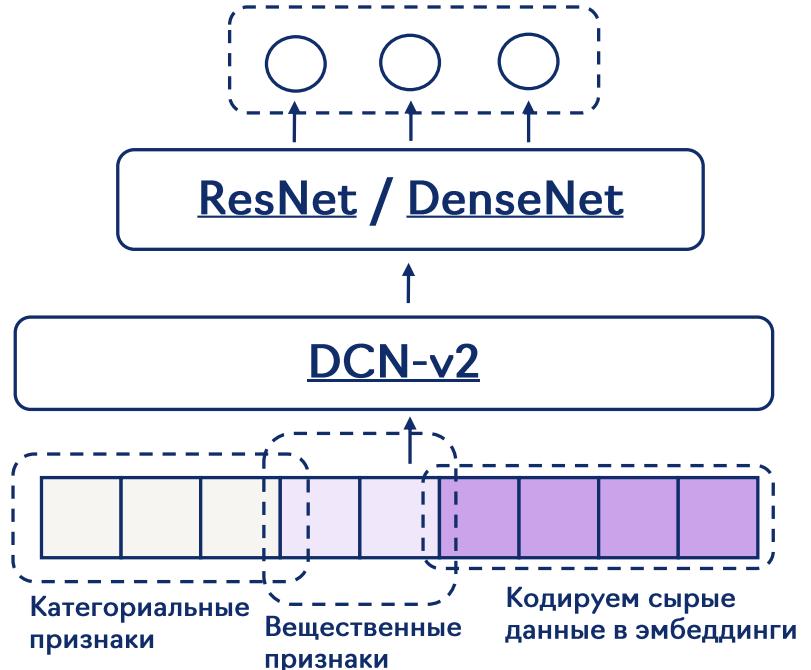


Зачем нам двухбашенность?

На стадии ранжирования используется раннее связывание:

- Подаём на вход единой нейросети всю информацию про пользователя и айтемы
- Используем специализированные нейросетевые слои для моделирования взаимодействия признаков

Не можем применить такую модель над всем каталогом.



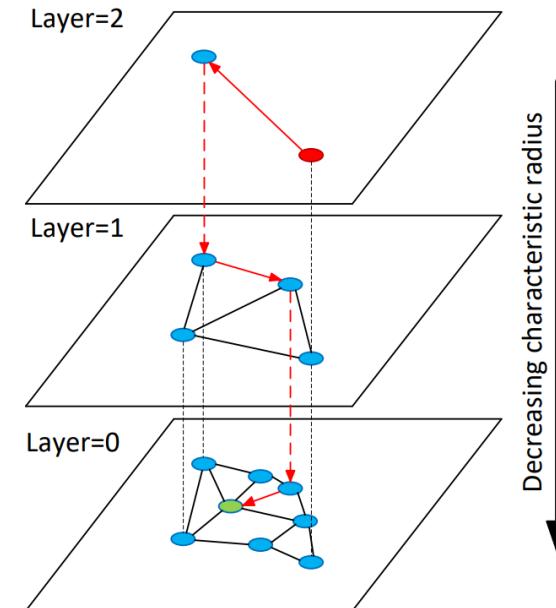
Архитектура ранжирующей нейросети.

Эффективность двухбашенных моделей

- Двухбашенность – это факторизация вычислений по пользователю и айтему:

$$s_\theta(u, d) = \langle g_\theta(u), h_\theta(d) \rangle$$

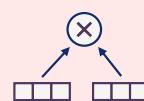
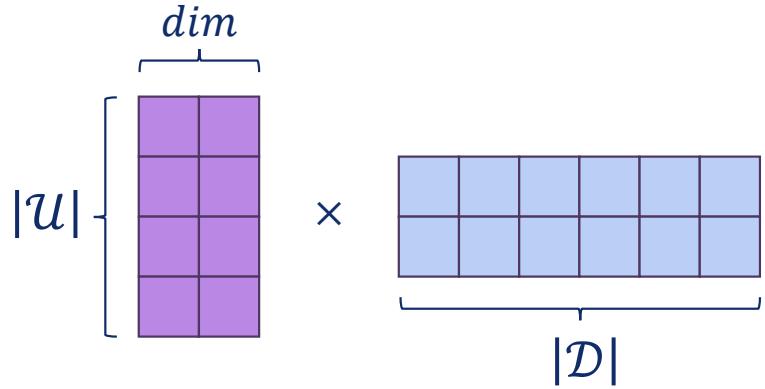
- Чтобы найти для пользователя top K кандидатов:
 1. Кодируем пользователя в эмбеддинг
 2. Идём в индекс, заранее построенный по эмбеддингам айтемов
 3. Получаем приближённый top K элементов из индекса с максимальным скалярным произведением



[Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs](#)

Двухбашенная модель как низкоранговое разложение

- Пусть есть матрица релевантности $\mathcal{R} \in \mathbb{R}^{|U| \times |D|}$, определяющая интерес пользователей к айтемам
- Двухбашенная модель учит низкоранговое разложение $\mathcal{R} \approx V_u \times V_d^T$, в котором
 - V_u – эмбеддинги пользователей
 - V_d – эмбеддинги айтемов
- Размерность эмбеддинга – это информационный бottлнек
 - ограничивает ранг разложения



Матричная факторизация – это частный случай двухбашенной модели, в которой башни – это просто таблицы обучаемых векторов.

Двухбашенная модель как низкоранговое разложение

Достаточно широкий эмбеддинг может выучить любую функцию похожести (матрицу релевантности):

- Вектор пользователя – one hot представление

$$v_u = (0, \dots, 1, \dots, 0) \in \mathbb{R}^{|\mathcal{U}|}$$

- Вектор айтема – столбец из R , соответствующая айтему

$$v_d = (\mathcal{R}_{ud})_{u=1}^{|\mathcal{U}|}$$

- Тогда $s(u, d) = \langle v_u, v_d \rangle = \mathcal{R}_{ud}$

На практике не можем позволить себе очень большие размерности из-за ограничений по памяти и скорости

- Используем сотни-тысячи

Обучение

Часть 2

Как учатся ранжирующие модели

- **Impression-aware:** учимся на показанных рекомендациях
 - **Pointwise / pairwise** обучение
 - Соответствует ли область применения области обучения?

Pointwise

$$P(y|u, d) = \begin{cases} \text{logit} \hat{p}(f(u, d)), & y = 1 \\ \hat{p}(-f(u, d)), & y = 0 \end{cases}$$

$$L(u, d, y) = -y \log \hat{p}(f(u, d)) - (1-y) \log \hat{p}(-f(u, d))$$

Paizweise

$(u, d_1, y_1), (u, d_2, y_2): y_1 > y_2$

(u, d_+, d_-)

$P(y_1 > y_2 | u, \underline{d_1, d_2}) =$

$$= \beta(f(u, d_1) - f(u, d_2))$$

$\chi = -\log P$

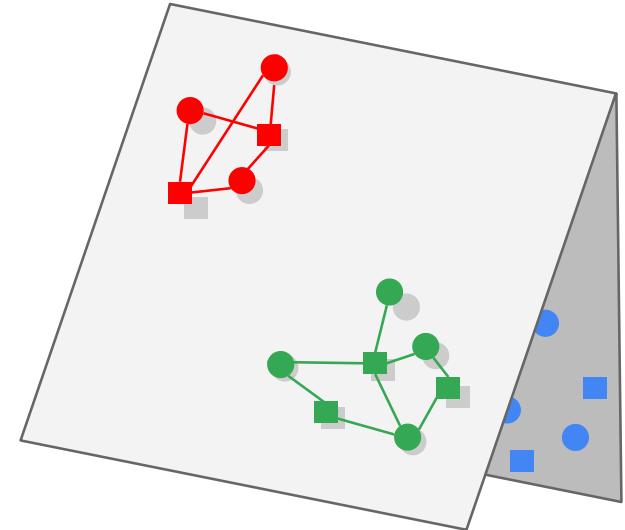
$P(d_+ | u, d_1, d_2) = \frac{e^{f(u, d_1)}}{e^{f(u, d_1)} + e^{f(u, d_2)} =$

$$= \frac{1}{1 + e^{-f(u, d_1) - f(u, d_2)}}$$

Folding

- Если рексистема работает хорошо, то в выдачах не встречается случайных айтемов
- Возникают непересекающиеся user-item группы
 - Испанцам – видеоролики на испанском, итальянцам – на итальянском
- Ранжирующий лосс учит модель делать сравнения только в рамках этих групп
- Возникает **folding** – разные user-item группы накладываются (англ. “to fold”) друг на друга в векторном пространстве

Нам нужна способность глобального сравнения: для пользователя сравнивать все айтемы, а не только внутри “фолдов”



Из [Google for Developers, Recommender Systems](#)

[Folding: Why Good Models Sometimes Make Spurious Recommendations](#)

Одностадийная рексистема

- Как должна обучаться одностадийная рексистема?
- Будет ли folding при обучении на ранжирование?

Softmax model

Перейдём от $P(y = 1 | user, item) \rightarrow P(item | user, y = 1)$

- **Раньше:** пользователю u показали видео i . Поставил ли он лайк ($y = 1$)?
- **Теперь:** знаем, что пользователь u поставил лайк ($y = 1$) какому-то видео i . Что это за видео i могло быть?

Это задача **экстремальной классификации**, в которой каждому айтему из каталога соответствует свой класс.

Softmax model

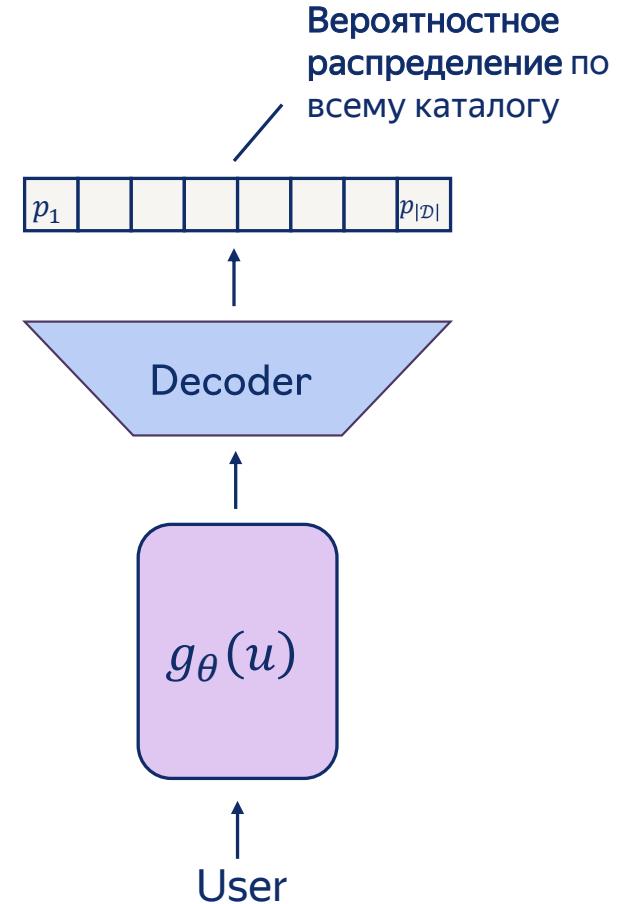
- На входе в модель – пользователь
- На выходе – вероятностное распределение по всем айтемам

$$P(d | u) := P(d | u, y = 1)$$

Для моделирования вероятностей используем softmax:

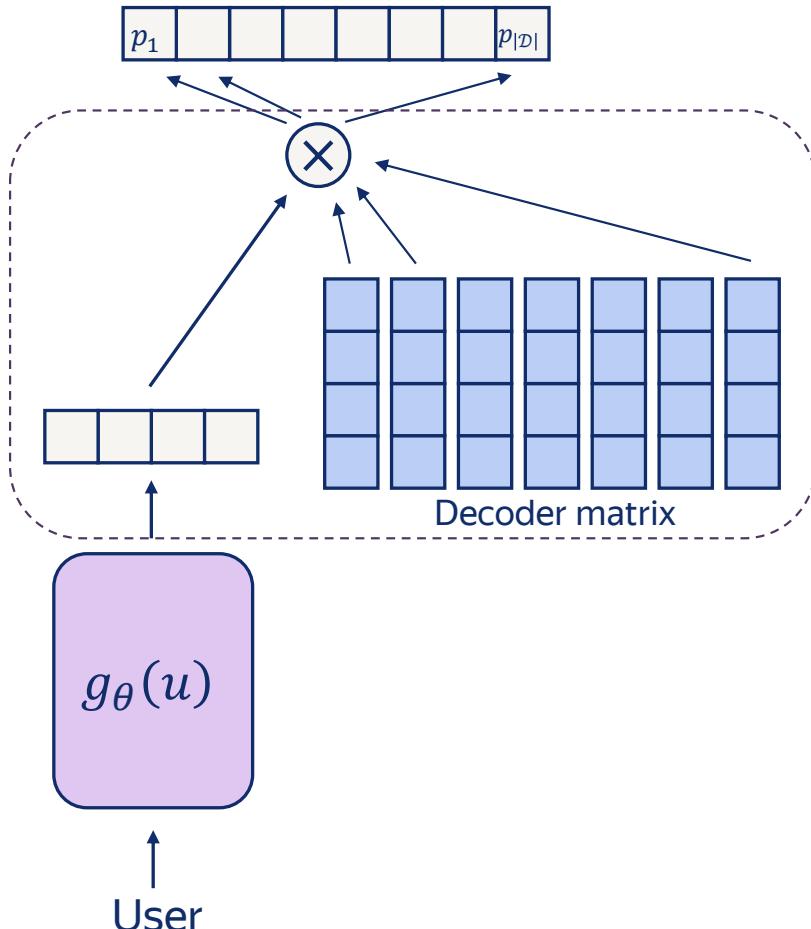
$$P(d | u) = \frac{e^{s_\theta(u, d)}}{\sum_{d' \in \mathcal{D}} e^{s_\theta(u, d')}}$$

где $s_\theta(u, d)$ – функция похожести



Softmax model as a two-tower model

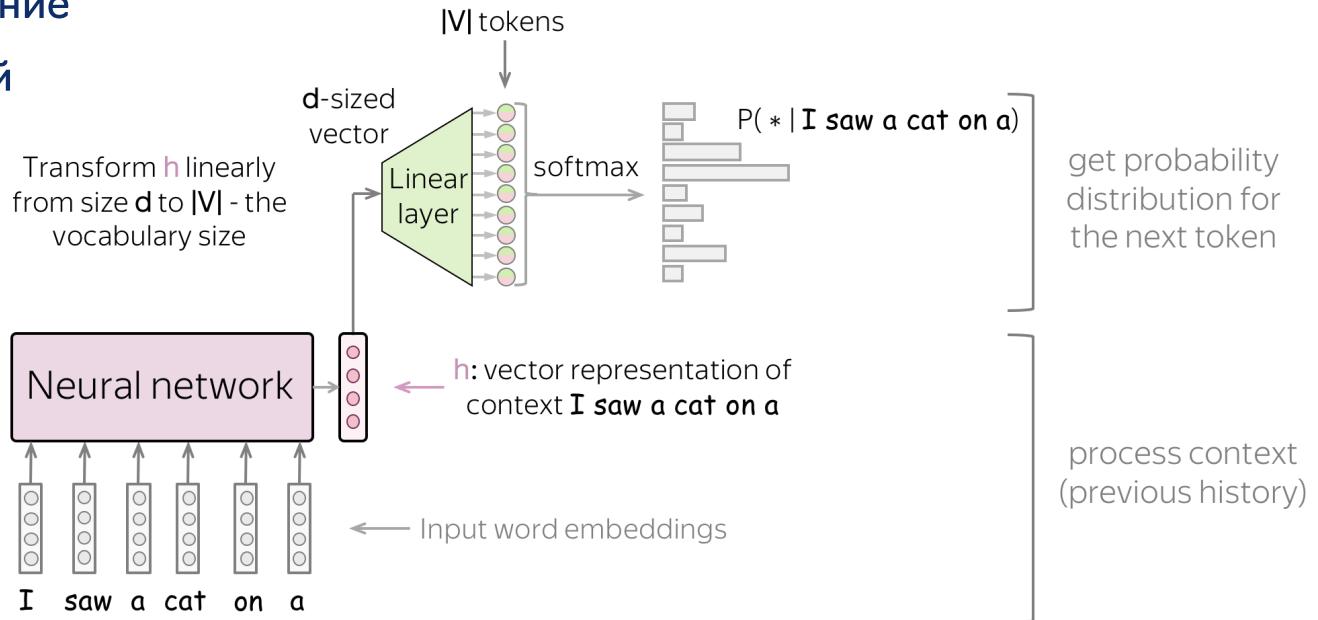
- Получаем распределение с помощью умножения на декодер матрицу V :
 $\bar{p} = \text{softmax}(g_\theta(u)V^T)$, то есть
 $s_\theta(u, d) = \langle g_\theta(u), v_d \rangle$, где $g_\theta(u)$ – вектор-строка
- Декодер – матрица обучаемых эмбеддингов айтемов
- Это двухбашенная модель:
 - Левая часть – энкодер пользователя $g_\theta(u), u \in \mathcal{U}$
 - Правая часть – обучаемые эмбеддинги айтемов $v_1, \dots, v_{|\mathcal{D}|}$



LLM as a two-tower model

LLM – это softmax model:

- Левая часть: уже обработанное предложение
- Правая часть: следующий токен



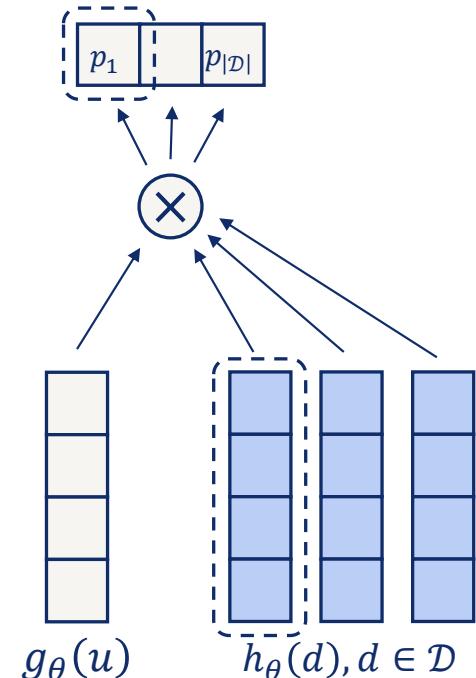
Softmax loss

Максимизируем правдоподобие реальных положительных взаимодействий:

$$\mathcal{L}_{\text{softmax}}(u, d_+) = -\log P_\theta(d_+ | u) = -\log \frac{e^{s_\theta(u, d_+)}}{\sum_{d \in \mathcal{D}} e^{s_\theta(u, d)}}$$

Полный софтмакс не вычислим на больших масштабах:

- Каталоги больше 10^5 – слишком дорогой знаменатель
- И по вычислениям, и по памяти невозможно суммировать по всему каталогу



Sampled softmax loss

Sampled softmax loss аппроксимирует знаменатель с помощью сэмплирования негативов:

$$\mathcal{L}_{\text{sampled}}(u, d_+, N) = -\log \frac{e^{s_\theta(u, d_+)}}{e^{s_\theta(u, d_+)} + \sum_{d \in N} e^{s_\theta(u, d)}}$$

- Где $N = \{d_1, \dots, d_{|N|} : d_i \sim Q(d)\}$ – сэмплированные негативы
- Это тоже задача классификации, но с подмножеством каталога в качестве классов

Contrastive learning & InfoNCE

Contrastive learning – сближаем похожие объекты и отдаляем непохожие в рамках обучения модели.

Пусть есть батч из положительных пар вида $(u_1, d_1), \dots, (u_n, d_n)$.
Тогда:

$$\mathcal{L}_{\text{InfoNCE}}(u_i, d_i) = -\log \frac{e^{\cos(u_i, d_i)/\tau}}{\sum_{j=1}^n e^{\cos(u_j, d_j)/\tau}}$$

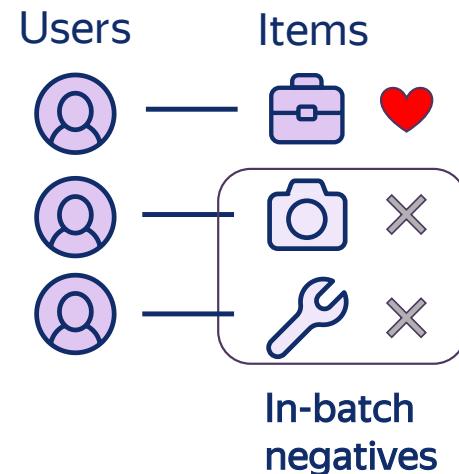
InfoNCE:

- $s_\theta(u, d) = \cos(u, d)/\tau$, где $\tau > 0$ – это температура
- In-batch negatives

In-batch negatives

В батче n положительных пар ($user, item$):

- Для каждой пары переиспользуем в качестве негативов позитивы из других пар
- Это очень эффективно:
 - Не нужно перевычислять эмбеддинги для негативов
 - Можно переиспользовать негативы с прошлых батчей (накапливать очередь, **memory queue**)
 - При распределенном обучении можно собирать **cross-device** негативы с других видеокарт



1. [Momentum Contrast for Unsupervised Visual Representation Learning](#)
2. [Cross-Batch Negative Sampling for Training Two-Tower Recommendations](#)

Про косинус

- Обучение с косинусом – более стабильное
 - Логиты ограничены по масштабу
- Если объекты достаточно хорошо кластеризованы, появляется линейная сепарабельность
- Нормы айтемов имеют тенденцию выучивать популярность
 - Косинус убирает влияние нормы (понижаем popularity bias)

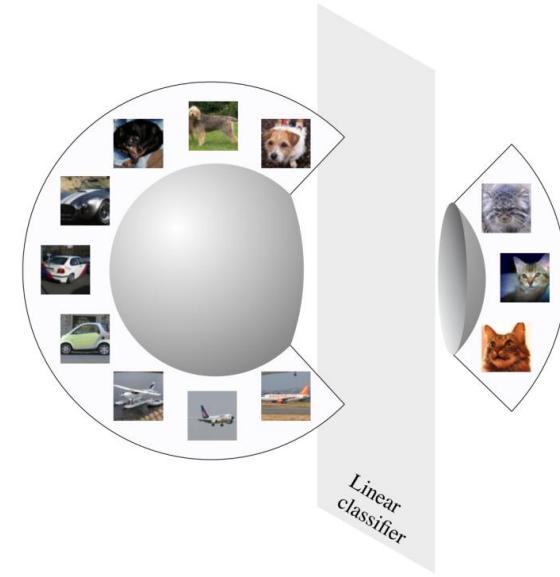


Figure 2: **Hypersphere**: When classes are well-clustered (forming spherical caps), they are linearly separable. The same does not hold for Euclidean spaces.

[Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere](#)

Температура

$$s_\theta(u, d) = \cos(u, d) / \tau$$

- Если используем косинус, то логиты ограничены $[-1; 1]$
- Такой софтмакс будет очень “гладким”
- Температура позволяет отмасштабировать логиты до большего интервала значений
- Чем меньше температура, тем больше интервал значений

Логиты	τ	P(positive)	P(negative)
1, -1, -1, -1, -1	1	0.65	0.088
1, -1, -1, -1, -1	0.1	~1.0	~0

Температура как мера неопределенности

- Для заданных (s_1, \dots, s_n) распределение с температурой можно записать следующим образом:

$$p_k = \frac{e^{s_k/\tau}}{\sum_{i=1}^K e^{s_i/\tau}} = \frac{e^{-E_k/\tau}}{\sum_{i=1}^K e^{-E_i/\tau}} = \frac{e^{-E_k/\tau}}{Z}$$

- Это Boltzmann / Gibbs distribution
- Температура – мера неопределенности
 - Чем больше температура, тем больше энтропия распределения

Температура как мера неопределенности

$$H(p) = - \sum_k p_k \underbrace{\log p_k}_{\text{surprisal}}. \quad \max_p H(p) = H(\text{Uniform}) = \log Z. \quad \min_p H(p) = H(S) = 0$$

$$T \downarrow \frac{\partial H}{\partial \tau} > 0$$

$$\gamma = \tau^{-1}$$

$$H(p) = - \sum_k \frac{e^{\gamma s_k}}{Z} \log \frac{e^{\gamma s_k}}{Z} = - \sum_k \frac{e^{\gamma s_k}}{Z} (\gamma s_k - \log Z) = -\gamma \sum_k \underbrace{\frac{e^{\gamma s_k}}{Z} s_k}_{\mathbb{E}[S]} + \log Z =$$

$$= -\gamma \mathbb{E}[S] + \log Z$$

$$\frac{\partial}{\partial \gamma} (Z) = \frac{\partial}{\partial \gamma} \left(\sum_k e^{\gamma s_k} \right) = \sum_k e^{\gamma s_k} s_k \frac{\gamma}{Z} = Z \cdot \mathbb{E}[S]$$

$$2) \frac{\partial}{\partial \gamma} (\log Z) = \mathbb{E}[S]$$

$$3) \frac{\partial}{\partial \gamma} (\mathbb{E}[S]) = \frac{\partial}{\partial \gamma} \left(\sum_k \frac{e^{\gamma s_k}}{Z} s_k \right) = \sum_k \frac{e^{\gamma s_k} \cdot s_k \cancel{Z} - e^{\gamma s_k} \cdot \cancel{Z} \mathbb{E}[S]}{Z \cdot \cancel{Z}} \quad s_k = \sum_i \frac{e^{\gamma s_k}}{Z} (s_k - \mathbb{E}[S]) s_i =$$

$$= D[S] > 0$$

$$\frac{\partial \mathcal{H}}{\partial \alpha}(\mathcal{A}) = -\cancel{[E[S]]} - \alpha \cdot \mathbb{D}[S] + \cancel{[E[S]]} = -\alpha \mathbb{D}[S]$$

Температура как мера неопределенности

$$\frac{\partial \mathcal{H}}{\partial \tau} = \frac{\partial \mathcal{H}}{\partial \alpha} \cdot \frac{\partial \alpha}{\partial \tau} = -\frac{1}{\tau} \mathbb{D}[S] \cdot \left(-\frac{1}{\tau^2}\right) = \frac{1}{\tau^3} \mathbb{D}[S] > 0$$

$$\underline{T_h} \quad m = \arg \max_k S_k \quad (\text{и } \exists \text{ on e}^{\text{данных}})$$

$$\lim_{\tau \rightarrow 0} p_k = \begin{cases} 1, & k = m \\ 0, & k \neq m \end{cases}$$

► $\lim_{\tau \rightarrow 0} \frac{e^{s_k/\tau}}{\sum_{i=1}^k e^{s_i/\tau}} = \lim_{\tau \rightarrow 0} \frac{e^{s_k/\tau}}{e^{s_m/\tau} + \sum_{i \neq m} e^{s_i/\tau}} = \lim_{\tau \rightarrow 0} \frac{e^{(s_k - s_m)/\tau}}{1 + \sum_{i \neq m} e^{(s_i - s_m)/\tau}}$

$\lim \frac{1}{1+0}$

$\frac{0}{1+0}$

Свойства софтмакса

- Софтмакс неявно майнит хард негативы
- Негативы конкурируют между собой за вклад в градиент

$$s_+, \underbrace{s_1, \dots, s_n}_{\text{негативы}}$$

$$\mathcal{L}(s) = -\log \frac{e^{s_+/\tau}}{e^{s_+/\tau} + \sum_{i=1}^n e^{s_i/\tau}} = -s_+/\tau + \log \left(e^{s_+/\tau} + \sum_{i=1}^n e^{s_i/\tau} \right)$$

$$\frac{\partial \mathcal{L}}{\partial s_i} = \frac{1}{e^{s_+/\tau} + \sum_{i=1}^n e^{s_i/\tau}} \cdot e^{s_i/\tau} \cdot \frac{1}{\tau} = p_i \cdot \frac{1}{\tau} = \left| \frac{\partial \mathcal{L}}{\partial s_i} \right|$$

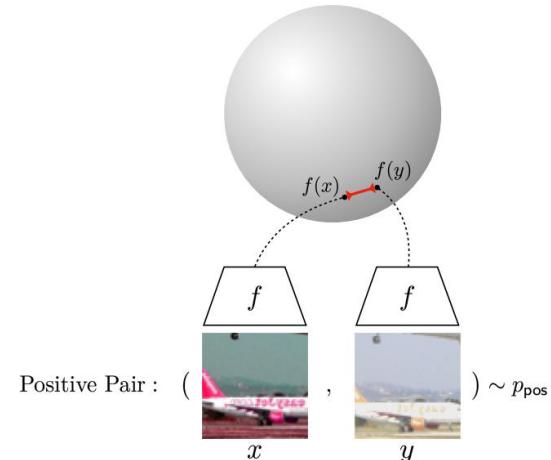
$$\left. \sum_{i=1}^n \left| \frac{\partial \mathcal{L}}{\partial s_i} \right| = \sum_{i=1}^n p_i \frac{1}{\tau} = (1 - p_+) \cdot \frac{1}{\tau} \right\} \frac{\left| \frac{\partial \mathcal{L}}{\partial s_i} \right|}{\left| \frac{\partial \mathcal{L}}{\partial s_j} \right|} = \frac{p_i}{p_j} = \frac{e^{s_i/\tau}}{e^{s_j/\tau}} = \boxed{\frac{(s_i - s_j)/\tau}{\tau}}$$

s_i - hard neg: $s_i > s_j$

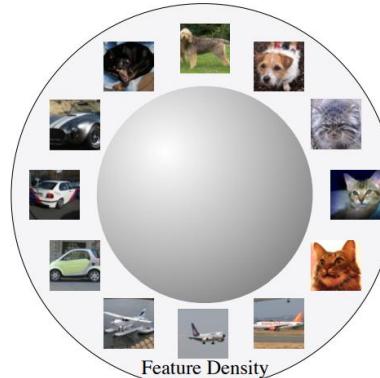
Свойства температуры

- **Alignment** – похожие объекты должны быть близко в пространстве
- **Uniformity** – объекты должны равномерно покрывать сферу

При уменьшении температуры увеличиваем uniformity (и уменьшаем alignment).



Alignment: Similar samples have similar features.
(Figure inspired by Tian et al. (2019).)



Uniformity: Preserve maximal information.

[Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere](#)

Обучаемая температура

- Можно зафиксировать температуру
- Можно сделать расписание для температуры (постепенно её уменьшать) – аналог curriculum learning
- Можно сделать обучаемую температуру – тогда это неуверенность модели

$$\begin{aligned}\tau &= \tau^{-1} \\ \mathcal{L} &= -\log \frac{e^{s_+ \tau}}{e^{s_+ \tau} + \sum_{i=1}^n e^{s_i \tau}} = -s_+ \tau + \log \mathcal{Z} \\ \frac{\partial \mathcal{L}}{\partial \tau} &= -s_+ + \frac{1}{\mathcal{Z}} \left(e^{s_+ \tau} \cdot s_+ + \sum_{i=1}^n e^{s_i \tau} \cdot s_i \right) = \\ &= -s_+ + \mathbb{E}[s]. \quad \text{Если } s_+ > \mathbb{E}[s] \Rightarrow \tau \uparrow \begin{array}{l} \text{(уверенность} \\ \text{растёт)} \end{array}\end{aligned}$$

Negative sampling

Softmax loss “майнит” хард негативы по всему каталогу – вклад в градиент больше у тех айтемов, в которых модель ошибается

- Если не можем взять все айтемы, надо взять те, у которых будет большой вклад в градиент

Основные источники негативов:

- Uniform
- In-batch negatives
- Mixed negative sampling

Uniform negatives

Uniform negatives, $d \sim \text{Uniform}(\mathcal{D})$:

- Равномерное сэмплирование из каталога
- Очень "простые" для модели
- Дают небольшой вклад в градиент
- Нужно большое количество, чтобы выучить модель

In-batch negatives

In-batch негативы эквивалентны сэмплированию из униграммного (частотного) распределения

- Частота попадания айтема в батч пропорциональна его общей популярности
- Это более сложные негативы по сравнению с равномерными, $f_\theta(u, d)$ в среднем больше для случайных пользователей
- Нужно меньше негативов, чтобы обучить модель

LogQ Correction

Часть 3

Bias of in-batch negatives

In-batch негативы приводят к систематическому смещению:

- Популярные айтемы часто появляются в негативах
- один раз как позитив и $batch\ size - 1$ раз как негатив
- Модель видит корреляцию между популярностью и "негативностью" → начинает штрафовать популярные объекты

Смещение оценки градиента

$$\nabla_{\theta} \mathcal{L}_{\text{full}}$$

$$\nabla_{\theta} \mathcal{L}_{\text{sampled}}$$

$$\mathbb{E}_N[\nabla_{\theta} \mathcal{L}_{\text{sampled}}] \neq \nabla_{\theta} \mathcal{L}_{\text{full}}$$

Importance sampling

$\int p(x), f(x)$

$$\mathbb{E}_{p(x)}[f(x)] \approx \frac{1}{n} \sum_{i=1}^n f(x_i), \quad x_i \sim p(x)$$

$q_r(x)$

$$\mathbb{E}_{p(x)}[f(x)] = \int p(x) f(x) dx = \int p(x) \frac{q_r(x)}{q_r(x)} f(x) dx = \mathbb{E}_{q_r(x)} \left[\frac{p(x)}{q_r(x)} f(x) \right]$$

$$\approx \frac{1}{n} \sum_{i=1}^n \frac{p(x_i)}{q_r(x_i)} f(x_i), \quad x_i \sim q_r(x) \leftarrow \text{proposal distribution}$$

$$\mathbb{E} = \sum_{d \in \mathcal{D}} e^{s(u, d)} = |\mathcal{D}| \sum_{d \in \mathcal{D}} \frac{1}{|\mathcal{D}|} e^{s(u, d)} = |\mathcal{D}| \mathbb{E}_{\text{Unif}(\mathcal{D})} [e^{s(u, d)}] =$$

$$= \cancel{|\mathcal{D}|} \cdot \frac{1}{n} \sum_{i=1}^n \frac{1}{\cancel{|\mathcal{D}|}} e^{s(u, d_i)} = \frac{1}{n} \sum_{i=1}^n e^{s(u, d_i) - \log Q(d_i)}, \quad d_i \sim Q(d)$$

Вывод logQ коррекции

$$\begin{aligned}
 1) \nabla_{\theta} \mathcal{L}_{full}(u, d_+) &= \nabla_{\theta} \left(-\log \frac{e^{s(u, d_+)}}{\sum e^{s(u, d)}} \right) = \\
 &= -\nabla_{\theta} s(u, d_+) + \frac{1}{2} \sum_{d \in \mathcal{W}} e^{s(u, d)} \nabla_{\theta} s(u, d) = \\
 &= -\nabla_{\theta} s(u, d_+) + \underbrace{\mathbb{E}_{P_{\theta}(d|u)} [\nabla_{\theta} s(u, d)]}_{\text{1}} = \\
 2) \text{(1)} &= \frac{1}{n} \sum_{i=1}^n \frac{P_{\theta}(d_i|u)}{Q(d_i)} \\
 &\nabla_{\theta} s(u, d_i) = \frac{1}{n} \sum_{i=1}^n e^{s(u, d_i) - \log Q(d_i)} \nabla_{\theta} s(u, d_i) = \\
 &= \frac{1}{n} \sum_{i=1}^n \frac{e^{s(u, d_i) - \log Q(d_i)}}{\sum_{j \neq i} e^{s(u, d_j) - \log Q(d_j)}} \nabla_{\theta} s(u, d_i) \leftarrow \text{2}
 \end{aligned}$$

LogQ коррекция

Sampled softmax loss с logQ коррекцией:

$$\mathcal{L}_{\log Q}(\underbrace{\mu, d_+}_N, N) = -\log \frac{e^{\underbrace{s_\theta(u, d_+)}_{-\log Q(d_+)}}}{e^{\underbrace{s_\theta(u, d_+)}_{-\log Q(d_+)}} + \sum_{d \in N} e^{s_\theta(u, d)} - \log Q(d)}$$

- Q – распределение, из которого сэмплируем негативы
- N – negative pool
- Интуиция: logQ поправка штрафует айтемы за популярность при обучении

[1. Quick Training of Probabilistic Neural Nets by Importance Sampling](#)

[2. Sampling-Bias-Corrected Neural Modeling for Large Corpus Item Recommendations](#)

Ошибка в выводе logQ коррекции

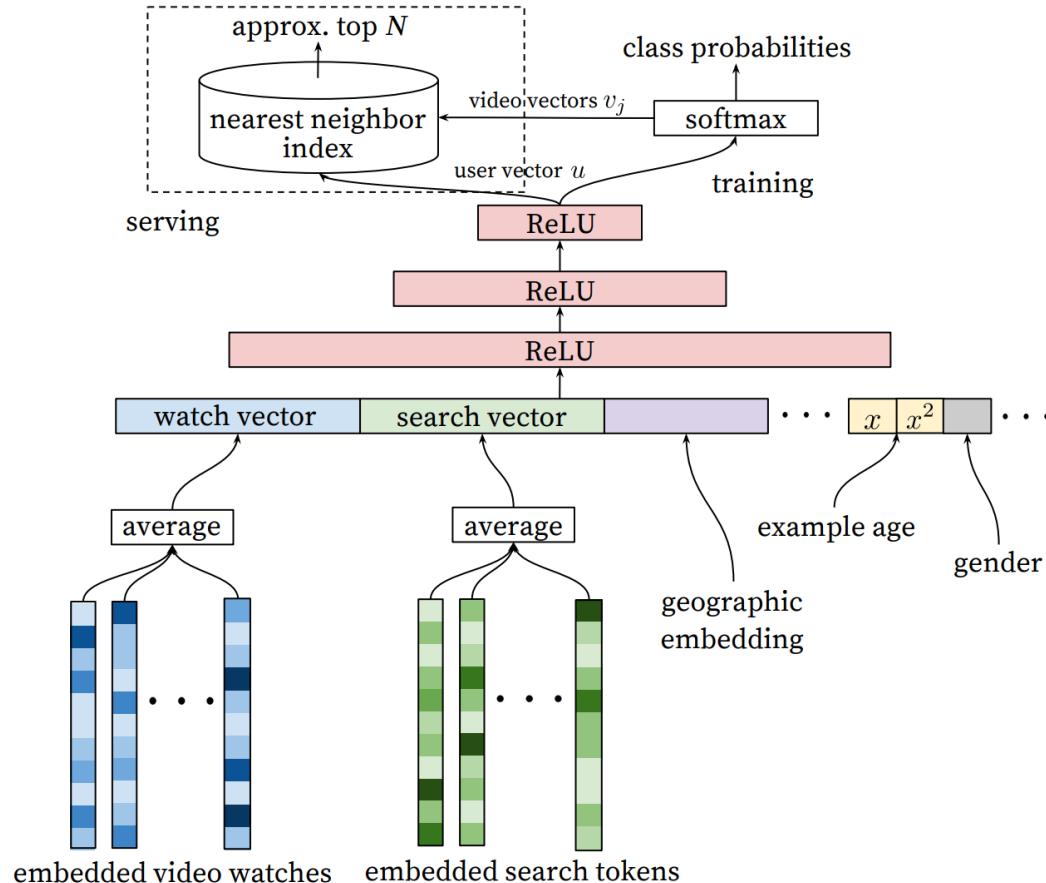
$$\begin{aligned}
 \nabla_{\theta} \mathcal{L}_{full} &= -\nabla_{\theta} S(u, d_+) + \sum_{d \in \mathcal{D}} \frac{e^{s(u, d)}}{\sum_{d' \in \mathcal{D}} e^{s(u, d')}} \nabla_{\theta} S(u, d) = \\
 &= -\nabla_{\theta} S(u, d_+) + \frac{e^{s(u, d_+)}}{2} \nabla_{\theta} S(u, d_+) + \sum_{\substack{d \in \mathcal{D} \\ d \neq d_+}} \frac{e^{s(u, d)}}{\sum_{\substack{d' \in \mathcal{D} \\ d' \neq d_+}} e^{s(u, d')}} \cdot \nabla_{\theta} S(u, d) \\
 &= (1 - P_{\theta}(d_+ | u)) \cdot (-\nabla_{\theta} S(u, d_+) + \sum_{\substack{d \in \mathcal{D} \\ d \neq d_+}} \frac{e^{s(u, d)}}{2 / \log 2} \nabla_{\theta} S(u, d)) \\
 &= (1 - P_{\theta}(d_+ | u)) (-\nabla_{\theta} S(u, d_+) + \mathbb{E}_{\substack{P_{\theta}(d | u, d \neq d_+)} \nabla_{\theta} S(u, d | u, d \neq d_+)})
 \end{aligned}$$

Correcting the LogQ Correction

$$\mathcal{L}_{\text{logQ}}^{\text{corrected}}(u, d_+, N) = \underbrace{w_{ud_+}}_{\substack{\text{вес сэмпла} \\ \text{вероятность, что модель} \\ \text{на нём ошибётся}}} \log \frac{\overbrace{e^{s_\theta(u, d_+)}}^{\text{вероятность ошибки}}}{\sum_{d \in N} e^{s_\theta(u, d)} - \log Q(d)}$$

- Где $w_{ud_+} = \text{stopgrad}(1 - P(d_+ | u))$ и $P(d_+ | u) \approx \frac{e^{s_\theta(u, d_+)}}{e^{s_\theta(u, d_+)} + \frac{1}{|N|} \sum_{d \in N} e^{s_\theta(u, d)} - \log Q(d)}$
- В знаменателе больше не участвует позитив
- Вес сэмпла = вероятность, что модель на нём ошибётся

YoutubeDNN



Summary

Обсудили:

- Как выглядят двухбашенные модели, зачем они нужны и как их обучать
- Свойства софтмакса, косинуса, температуры
- Источники негативов
- Проблемы униграммного сэмплирования негативов и $\log Q$ коррекцию

На семинаре

- Рекомендательные системы глазами исследователя
- Поиск информации и чтение статей