Курс: Рекомендательные системы
Занятие 7:
Sequential-based подходы

Красильников Денис
09.03.24

А зачем нам вообще учитывать последовательность?

А зачем нам вообще учитывать последовательность?

Представьте, что вы рекомендуете человеку магазины и видите человека с такой упорядоченной историей:

зоомагазин, супермаркет, метрополитен, зоомагазин, кофейня, супермаркет, развлекательный сервис

Порекомендуете ли совершить следующую покупку в зоомагазине?

А зачем нам вообще учитывать последовательность?

Представьте, что вы рекомендуете человеку магазины и видите человека с такой упорядоченной историей:

зоомагазин, супермаркет, метрополитен, зоомагазин, кофейня, супермаркет, развлекательный сервис, зоомагазин

Порекомендуете ли теперь совершить следующую покупку в зоомагазине?

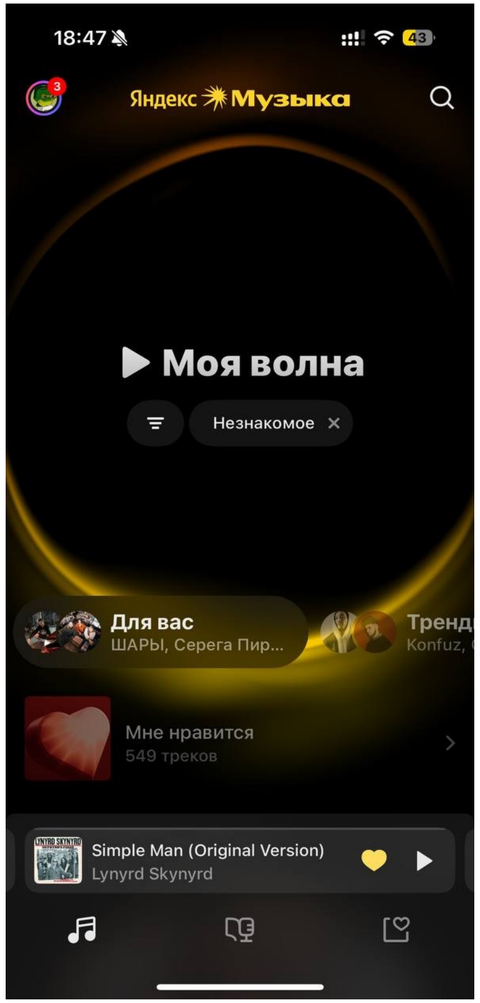
А зачем нам вообще учитывать последовательность?

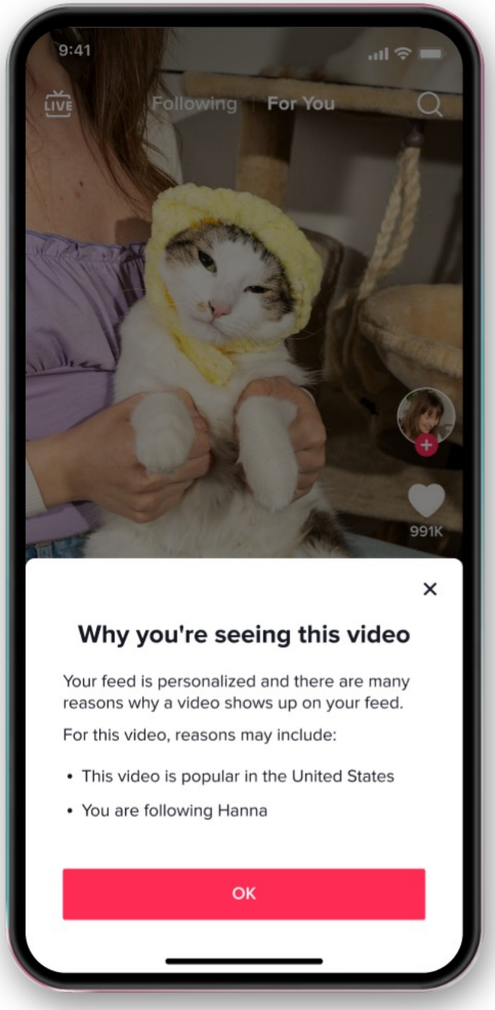
Представим теперь, что у наша последовательность не отсортирована и мы просто знаем, что человек покупал в магазинах

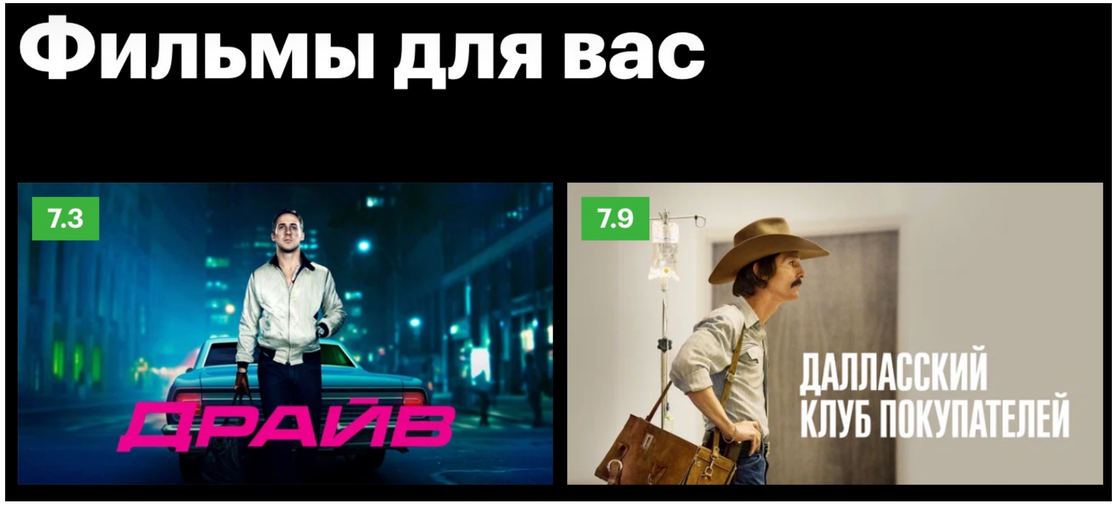
зоомагазин, супермаркет, кофейня, развлекательный сервис

Пора ли ему рекомендовать зоомагазин теперь?

Примеры где это работает







Как было раньше

User_id	Item_id	Date
5	4	2024-02-24
6	1	2024-02-24
5	3	2024-02-22
1	2	2024-02-13
5	1	2024-02-10
4	2	2024-02-08
2	3	2024-02-08
2	1	2024-02-07
….	….	….
1	4	2023-12-31

U/I	item 1	item2	item 3	item 4
User 1	0	1	0	1
User 2	1	0	1	0
User 3	0	1	0	1
User 4	0	1	0	0
User 5	1	0	1	1
User 6	1	1	0	1
User 7	0	1	1	0
User 8	1	0	0	1

Как будет теперь

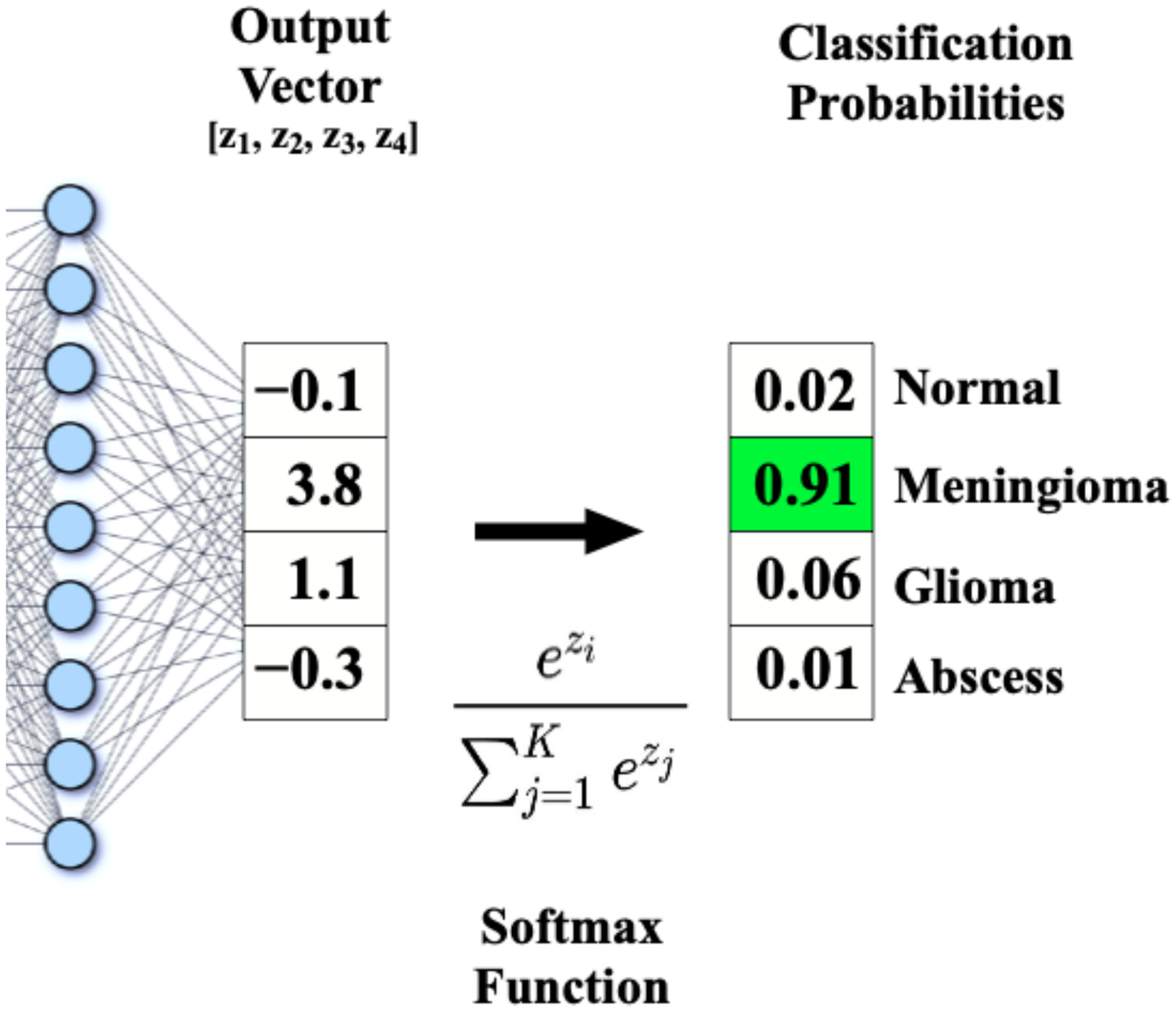
User_id	Item_id	Date
5	4	2024-02-24
6	1	2024-02-24
5	3	2024-02-22
1	2	2024-02-13
5	1	2024-02-10
4	2	2024-02-08
2	3	2024-02-08
2	1	2024-02-07
….	….	….
1	4	2023-12-31

User1: [4,1]
User2: [2,3]
User3: [2, 4]
User4: [2]
User5: [1,3,4]
User5: [2,3,1]
User7: [2,3]
User8: [1,4]

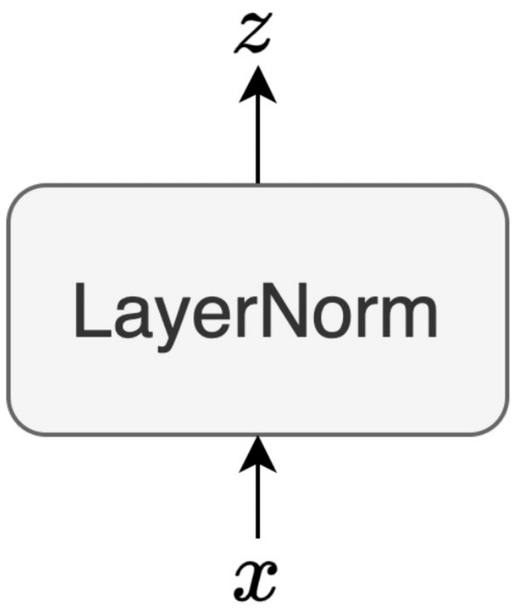
Время

Повторим базу

Softmax


Output
Vector
[z1, z2, z3, z4]
Classification
Probabilities
−0.1
3.8
1.1
−0.3
$\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$
0.02
Normal
0.91
Meningioma
0.06
Glioma
0.01
Abscess
Softmax
Function

LayerNorm



$$z_i = \alpha \cdot \frac{x_i - \hat{\mu}}{\hat{\sigma}} + \beta, \quad i \in \{1, 2, \ldots, d\},$$

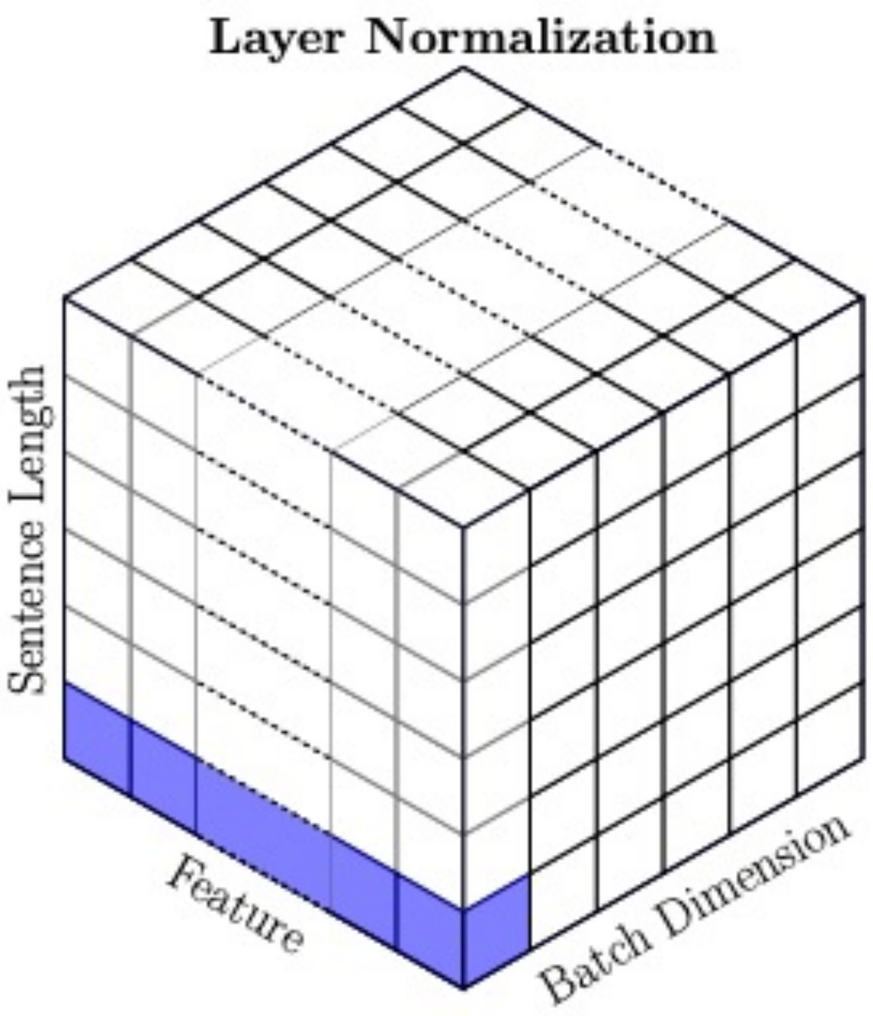
$$\hat{\mu} = \frac{\sum_{i=1}^{d} x_i}{d},$$

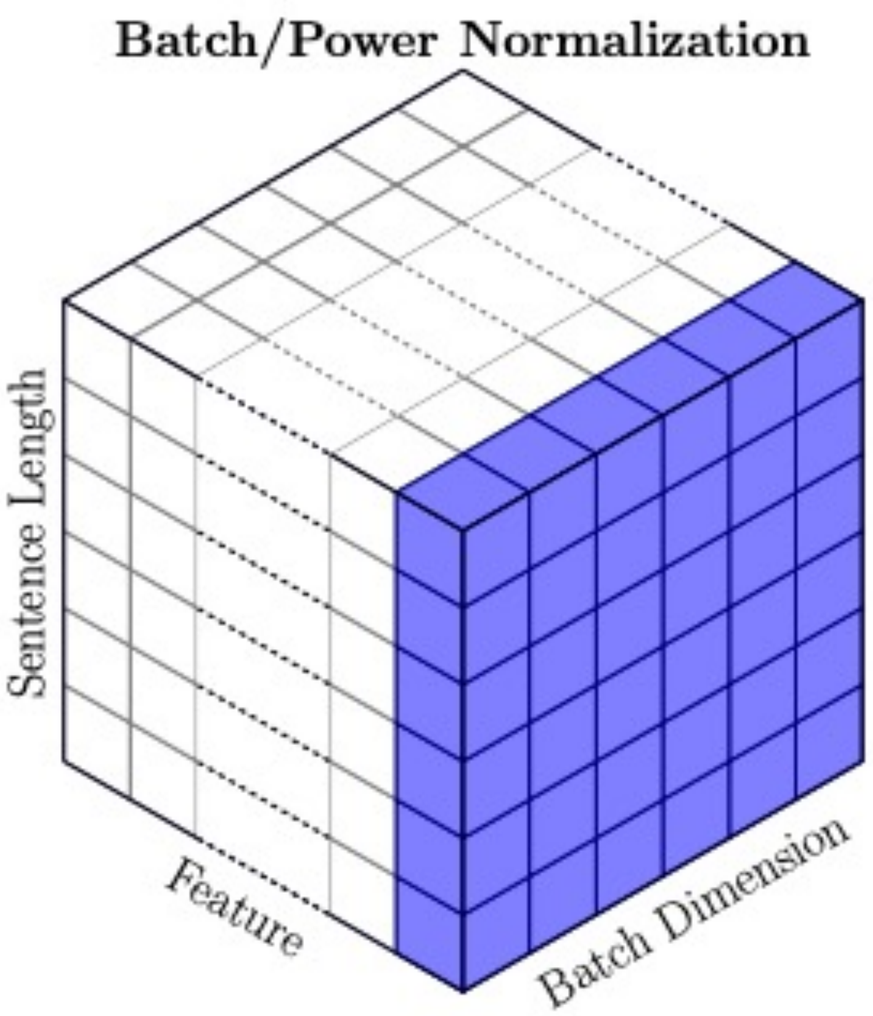
$$\hat{\sigma} = \sqrt{\frac{\sum_{i=1}^{d} (x_i - \hat{\mu})^2}{d}}$$

$$x, z \in \mathbb{R}^{1 \times d},$$

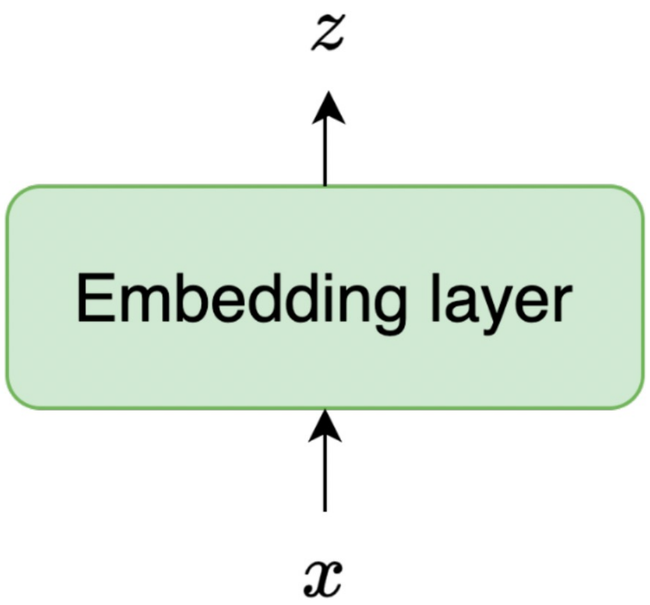
$$\alpha, \beta \in \mathbb{R}$$

BatchNorm vs LayerNorm





Эмбеддинги



$$z = (W_{x,1}, W_{x,2}, \ldots, W_{x,d}),$$

$$x \in \{0, 1, 2, \ldots, V-1\},$$
$$z \in \mathbb{R}^{1 \times d},$$
$$W \in \mathbb{R}^{V \times d}$$

Пусть x может принимать одно из V значений (от 0 до $V-1$).

Слой эмбеддингов превращает x в вектор.

То есть мы каждому айтему сопоставляем некоторый вектор из d-мерного пространства.

Эмбеддинги

Айтем	Индекс
бананы	0
молоко	1
хлеб	2
колбаса	3

0.32	-0.21	0.22	-0.23	0.01	-0.12
0.13	0.172	0.12	0.45	-0.42	0.65
0.01	0.89	0.65	-0.75	-0.31	-0.64
-0.98	1.23	-0.93	-0.85	0.31	0.33

Какой эмбеддинг у хлеба?

Эмбеддинги

Айтем	Индекс
бананы	0
молоко	1
хлеб	2
колбаса	3

0.32	-0.21	0.22	-0.23	0.01	-0.12
0.13	0.172	0.12	0.45	-0.42	0.65
0.01	0.89	0.65	-0.75	-0.31	-0.64
-0.98	1.23	-0.93	-0.85	0.31	0.33

Какой эмбеддинг у хлеба?

Эмбеддинги

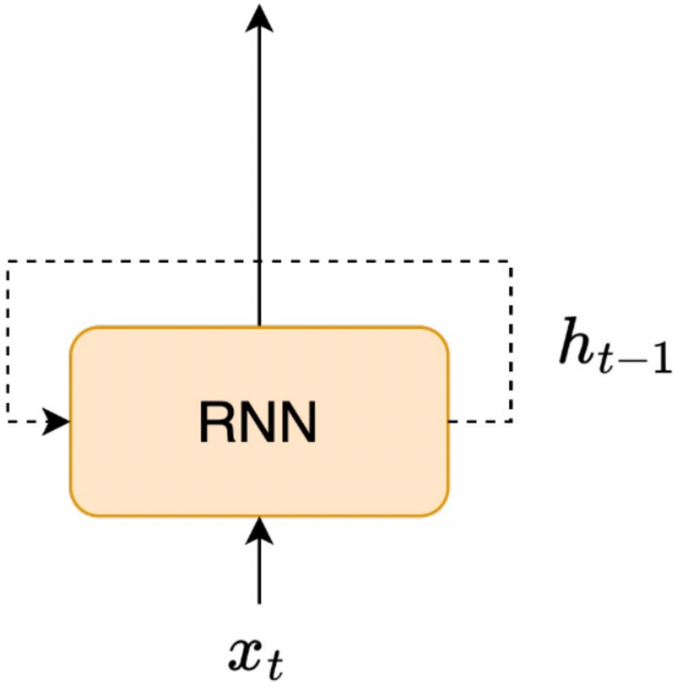
Айтем	Индекс
хлеб	0
колбаса	1
бананы	2
молоко	3

0.32	-0.21	0.22	-0.23	0.01	-0.12
0.13	0.172	0.12	0.45	-0.42	0.65
0.01	0.89	0.65	-0.75	-0.31	-0.64
-0.98	1.23	-0.93	-0.85	0.31	0.33

ПОРЯДОК **НЕВАЖЕН**!! ГЛАВНОЕ ЧТОБЫ ОН **НЕ МЕНЯЛСЯ**

Recurrent Neural Network (RNN)

Цель: обработать последовательность $x_1, x_2, \ldots, x_L$ зависимых наблюдений **нефиксированной** длины.



Recurrent Neural Network (RNN)

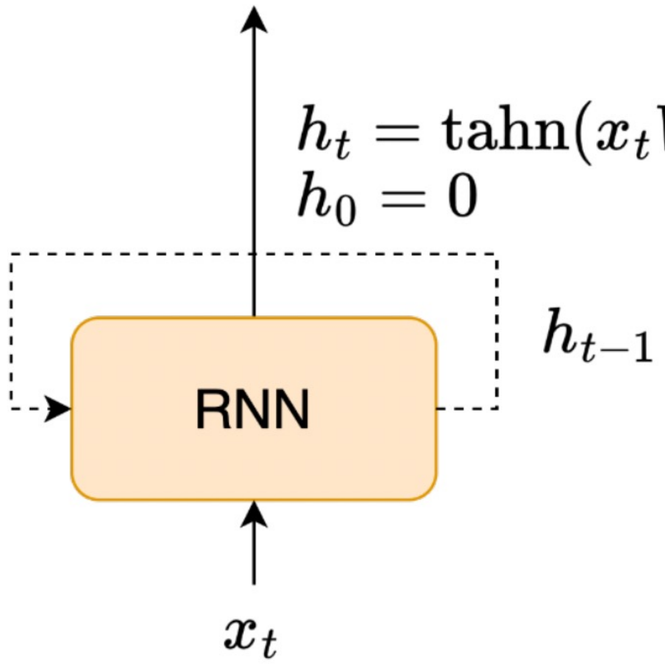
Цель: обработать последовательность $x_1, x_2, \ldots, x_L$ зависимых наблюдений **нефиксированной** длины.

$x_t \in \mathbb{R}^{1 \times d_x}$,

$h_t, b \in \mathbb{R}^{1 \times d_h}$,

$W_x \in \mathbb{R}^{d_x \times d_h}$,

$W_h \in \mathbb{R}^{d_h \times d_h}$



$h_t = \tanh(x_t W_x + h_{t-1} W_h + b)$,
$h_0 = 0$

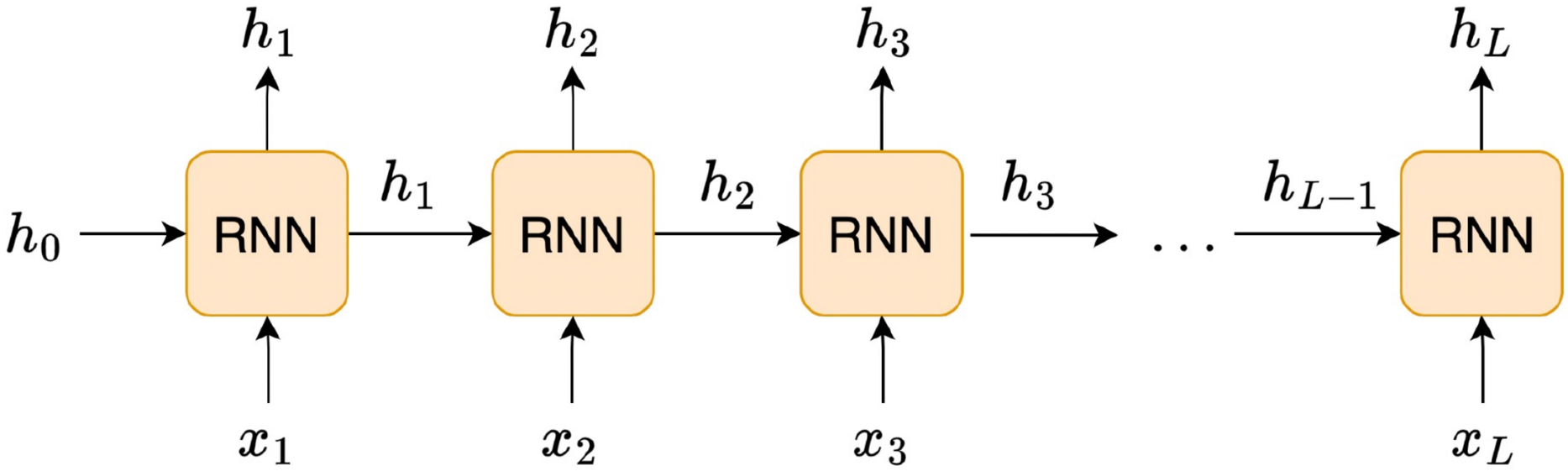
} Скрытые состояния

h_{t-1}

x_t

- Состояние h_t можно рассматривать как внутреннюю память модели на шаге t.
- Состояние h_t неявно зависит от всех $x_1, \ldots, x_t$, то есть хранит информацию о входах до шага t включительно.

Развернутая RNN



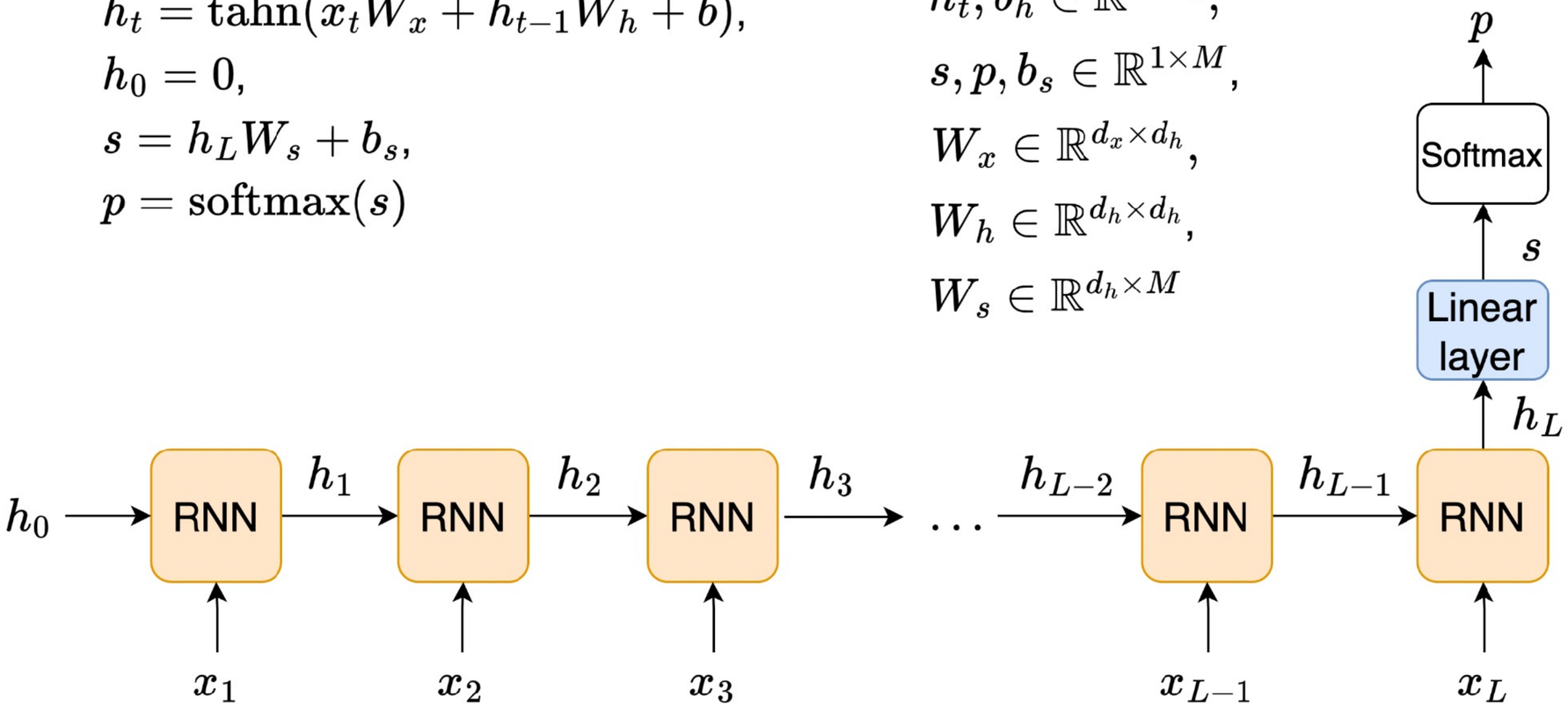
$$x_t \in \mathbb{R}^{1\times d_x},$$
$$h_t, b \in \mathbb{R}^{1\times d_h},$$
$$W_x \in \mathbb{R}^{d_x\times d_h},$$
$$W_h \in \mathbb{R}^{d_h\times d_h}$$

$$h_t = \tanh(x_t W_x + h_{t-1} W_h + b),$$
$$h_0 = 0$$

RNN для рекомендаций

$$h_t = \tanh(x_t W_x + h_{t-1} W_h + b),$$
$$h_0 = 0,$$
$$s = h_L W_s + b_s,$$
$$p = \mathrm{softmax}(s)$$

$$x_t \in \mathbb{R}^{1 \times d_x},$$
$$h_t, b_h \in \mathbb{R}^{1 \times d_h},$$
$$s, p, b_s \in \mathbb{R}^{1 \times M},$$
$$W_x \in \mathbb{R}^{d_x \times d_h},$$
$$W_h \in \mathbb{R}^{d_h \times d_h},$$
$$W_s \in \mathbb{R}^{d_h \times M}$$



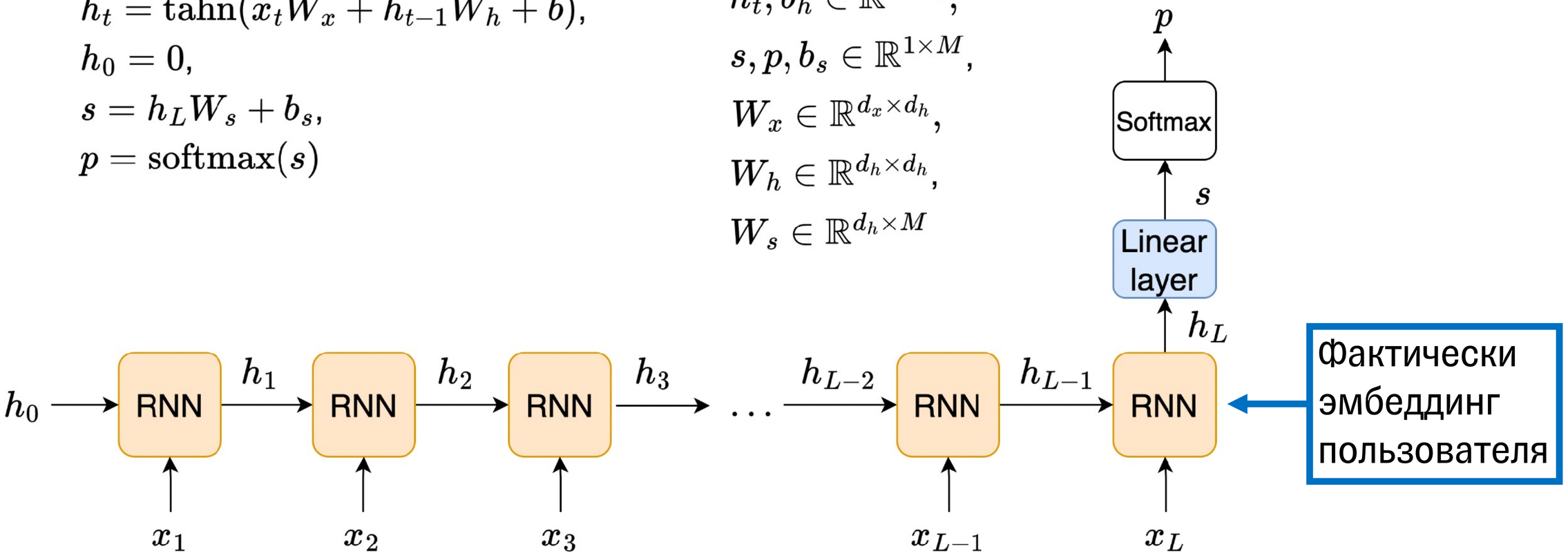
s - логиты,

p - предсказанное распределение (M классов)

RNN для рекомендаций

$$h_t = \tanh(x_t W_x + h_{t-1} W_h + b),$$
$$h_0 = 0,$$
$$s = h_L W_s + b_s,$$
$$p = \text{softmax}(s)$$

$$x_t \in \mathbb{R}^{1 \times d_x},$$
$$h_t, b_h \in \mathbb{R}^{1 \times d_h},$$
$$s, p, b_s \in \mathbb{R}^{1 \times M},$$
$$W_x \in \mathbb{R}^{d_x \times d_h},$$
$$W_h \in \mathbb{R}^{d_h \times d_h},$$
$$W_s \in \mathbb{R}^{d_h \times M}$$



s - логиты,

p - предсказанное распределение (M классов)

Серебряная пуля? Нет

- Не бесконечное capacity

Серебряная пуля? Нет

- Не бесконечное capacity
- Плохо параллелится

Серебряная пуля? Нет

- Не бесконечное capacity
- Плохо параллелится
- Затухают градиенты

Серебряная пуля? Нет

- Не бесконечное capacity
- Плохо параллелится
- Затухают градиенты
- Взрываются градиенты

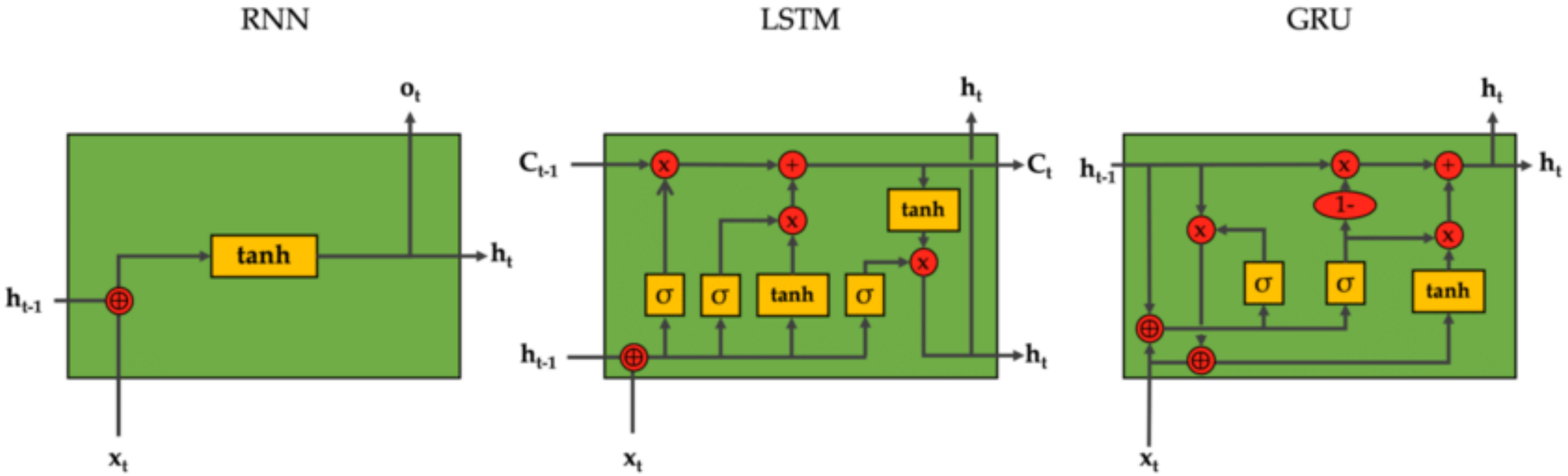
Серебряная пуля? Нет

- Не бесконечное capacity
- Плохо параллелится
- Затухают градиенты
- Взрываются градиенты
- Не может запоминать длинные последовательности

А можно как-то исправить?

- Не бесконечное capacity
- Плохо параллелится – можно упороться и распараллелить
- Затухают градиенты – есть модификации классической RNN
- Взрываются градиенты – можем их просто обрезать
- Не может запоминать длинные последовательности

Long-Short Term Memory & Gated Recurrent Unit



σ sigmoid function

tanh hyperbolic tangent function

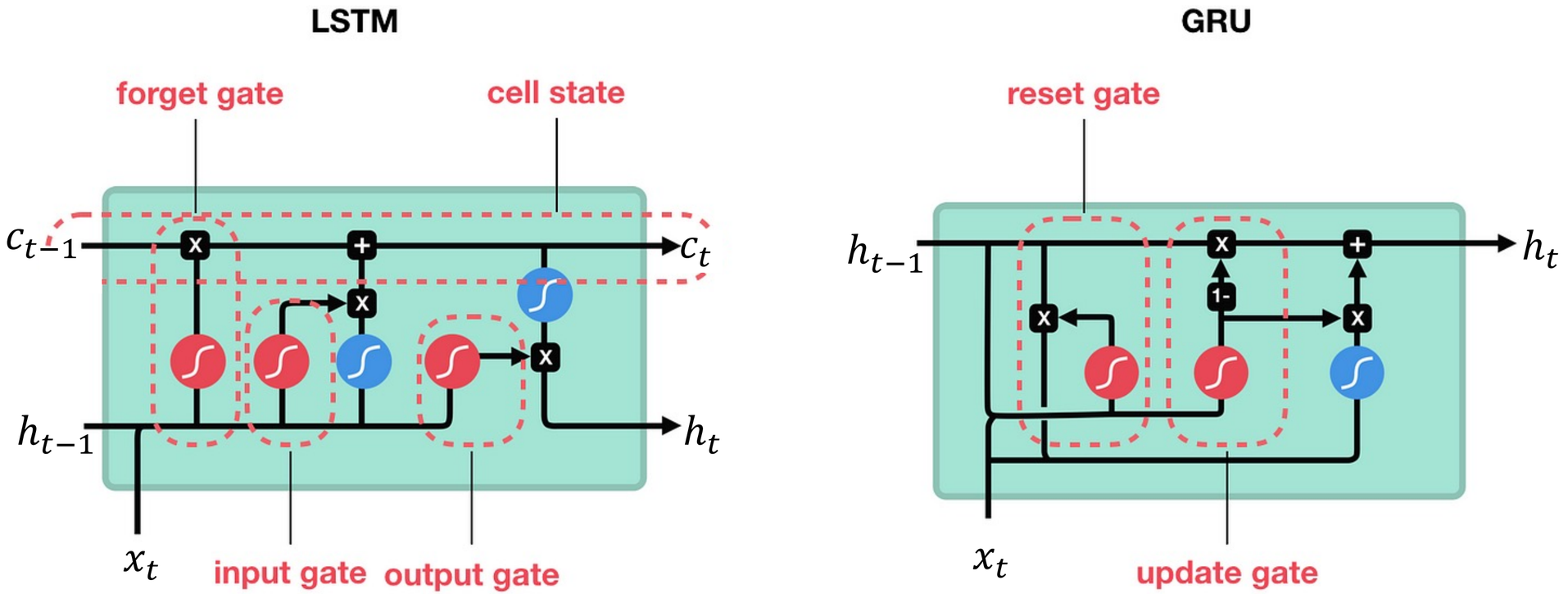
1- subtract from one

+ pointwise addition

x pointwise multiplication

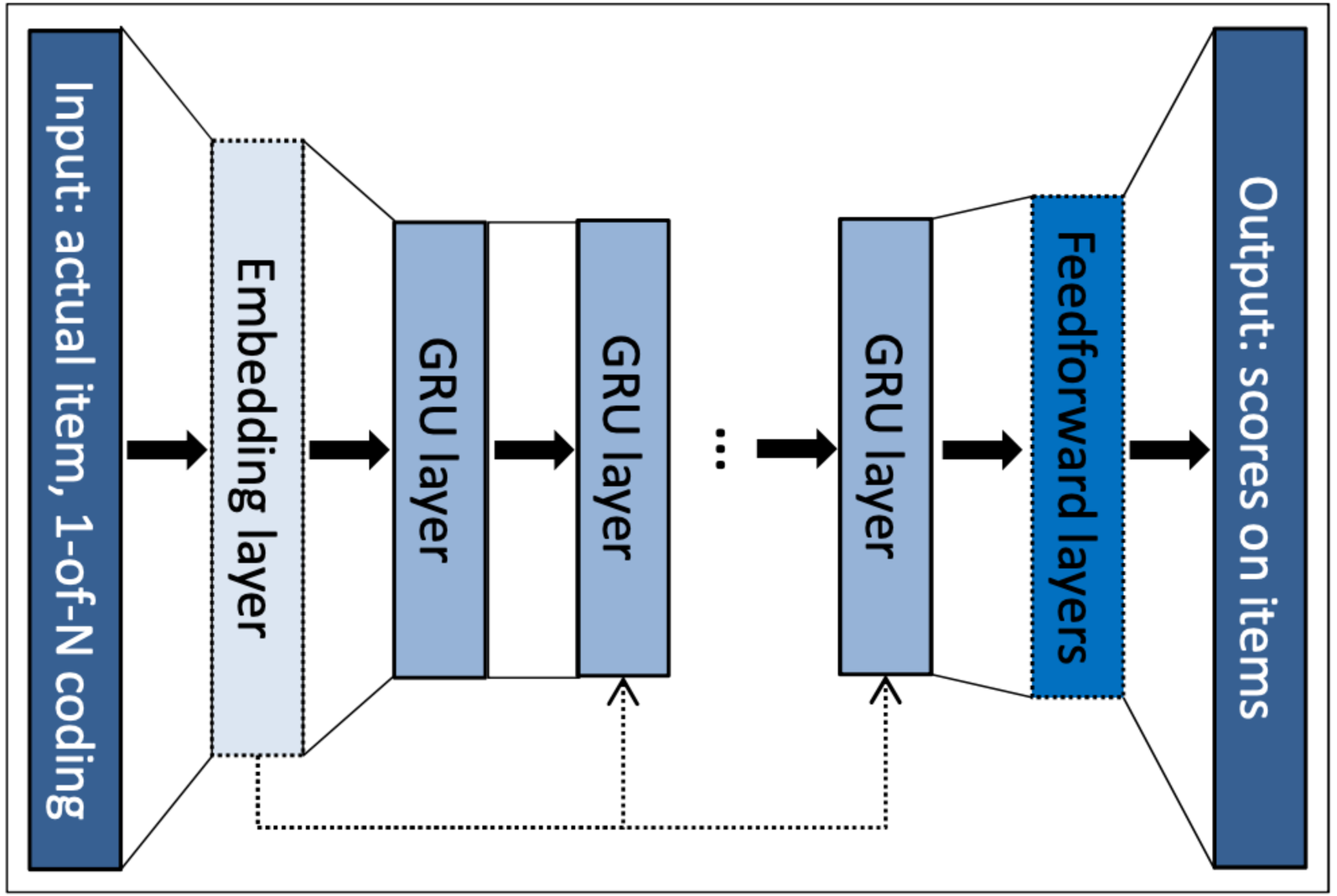
⊕ vector concatenation

LSTM & GRU


LSTM
forget gate
cell state
c_{t-1}
c_t
h_{t-1}
h_t
x_t
input gate
output gate
GRU
reset gate
h_{t-1}
h_t
x_t
update gate
1-

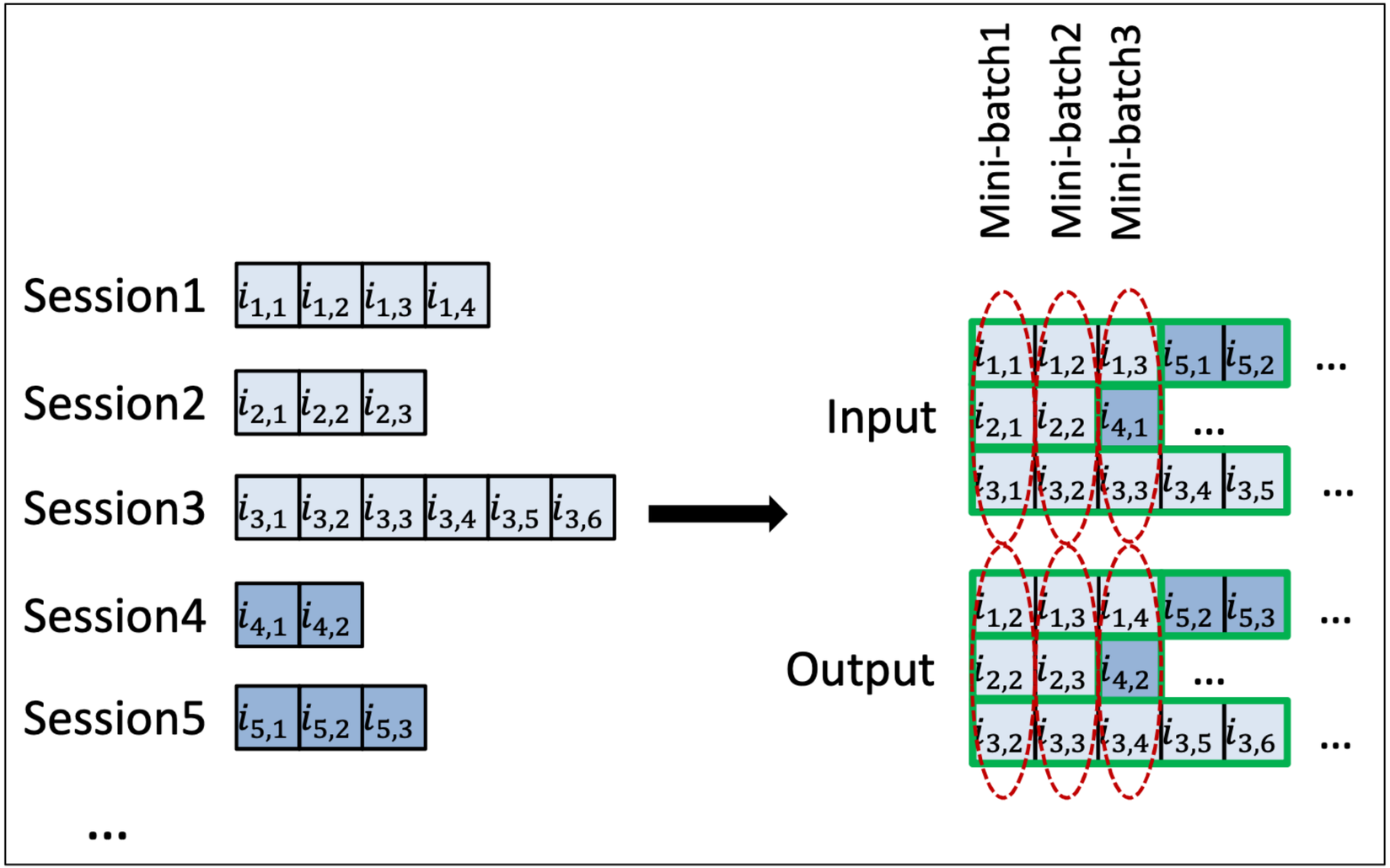

sigmoid
tanh
pointwise multiplication
pointwise addition
vector concatenation

GRU4Rec



Hidasi B, Karatzoglou A, Baltrunas L, Tikk D. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939. 2015 Nov 21.

GRU4Rec



Hidasi B, Karatzoglou A, Baltrunas L, Tikk D. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939. 2015 Nov 21.

GRU4Rec

Baseline	RSC15		VIDEO	
	Recall@20	MRR@20	Recall@20	MRR@20
POP	0.0050	0.0012	0.0499	0.0117
S-POP	0.2672	0.1775	0.1301	0.0863
Item-KNN	0.5065	0.2048	0.5508	0.3381
BPR-MF	0.2574	0.0618	0.0692	0.0374

Table 3: Recall@20 and MRR@20 for different types of a single layer of GRU, compared to the best baseline (item-KNN). Best results per dataset are highlighted.

Loss / #Units	RSC15		VIDEO	
	Recall@20	MRR@20	Recall@20	MRR@20
TOP1 100	0.5853 (+15.55%)	0.2305 (+12.58%)	0.6141 (+11.50%)	0.3511 (+3.84%)
BPR 100	0.6069 (+19.82%)	0.2407 (+17.54%)	0.5999 (+8.92%)	0.3260 (-3.56%)
Cross-entropy 100	0.6074 (+19.91%)	0.2430 (+18.65%)	0.6372 (+15.69%)	0.3720 (+10.04%)
TOP1 1000	0.6206 (+22.53%)	**0.2693 (+31.49%)**	**0.6624 (+20.27%)**	**0.3891 (+15.08%)**
BPR 1000	**0.6322 (+24.82%)**	0.2467 (+20.47%)	0.6311 (+14.58%)	0.3136 (-7.23%)
Cross-entropy 1000	0.5777 (+14.06%)	0.2153 (+5.16%)	–	–

Hidasi B, Karatzoglou A, Baltrunas L, Tikk D. Session-based recommendations with recurrent neural networks. arXiv preprint arXiv:1511.06939. 2015 Nov 21.

Внимание! Спасибо за внимание!

Внимание! Спасибо за внимание!

Внимание! Спасибо за внимание!

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

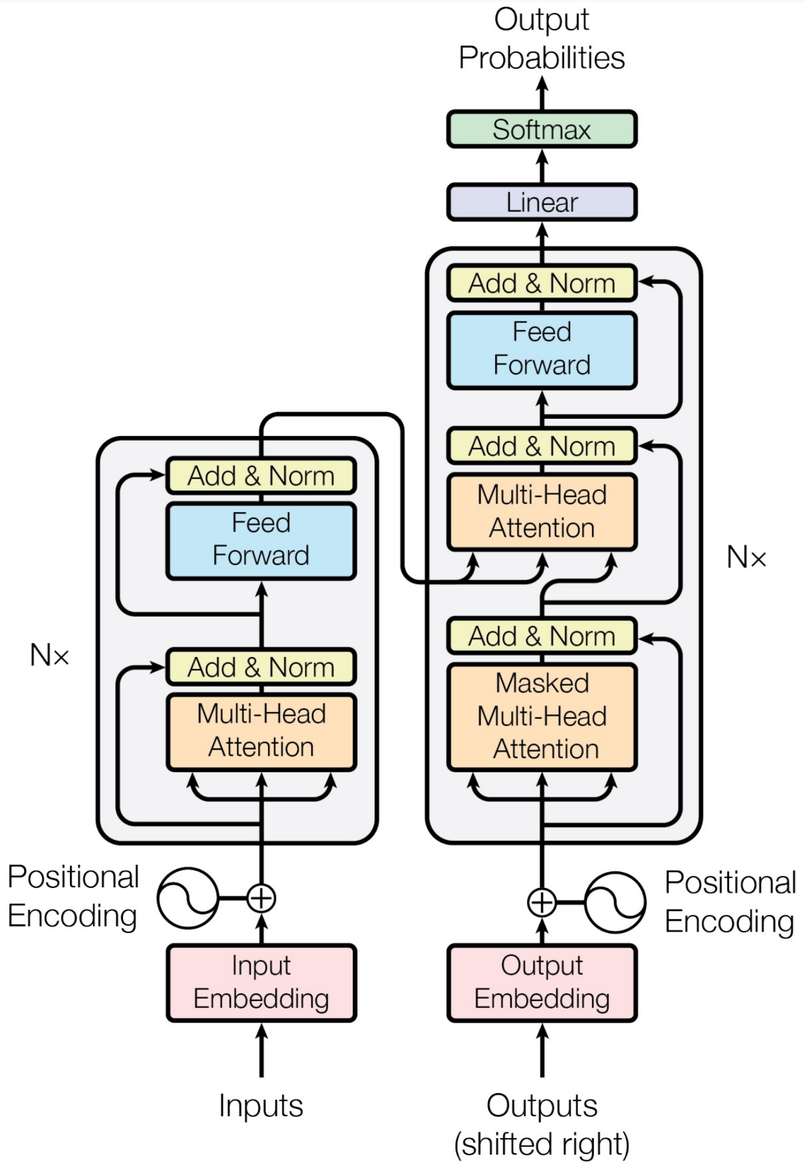
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

2017

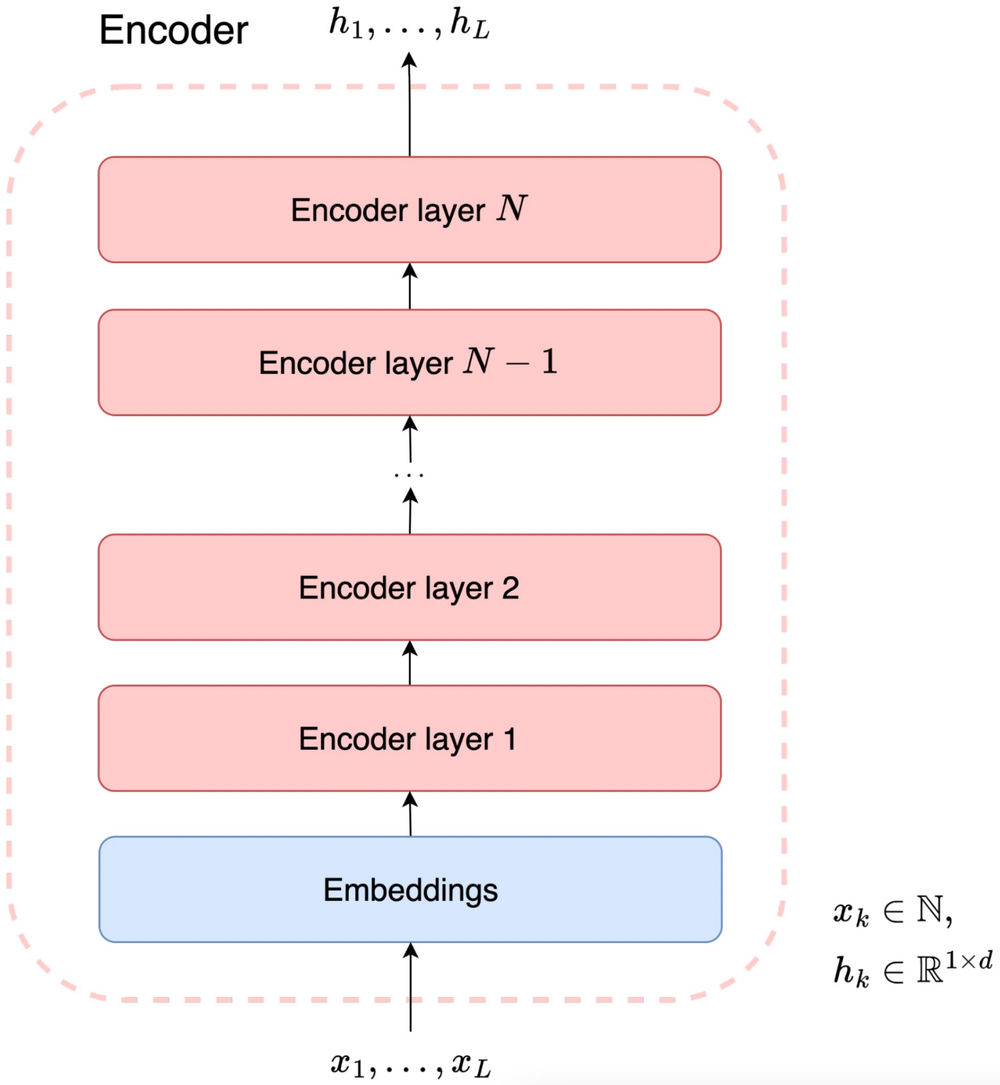
Внимание! Спасибо за внимание!



Архитектура, пришедшая на смену RNN:

- Хорошо параллелится
- Внутри нет рекуррентности и связанной с ней проблем, используется только механизм внимания
- Имеем доступ ко всей информации (нет узкого места, как в RNN)

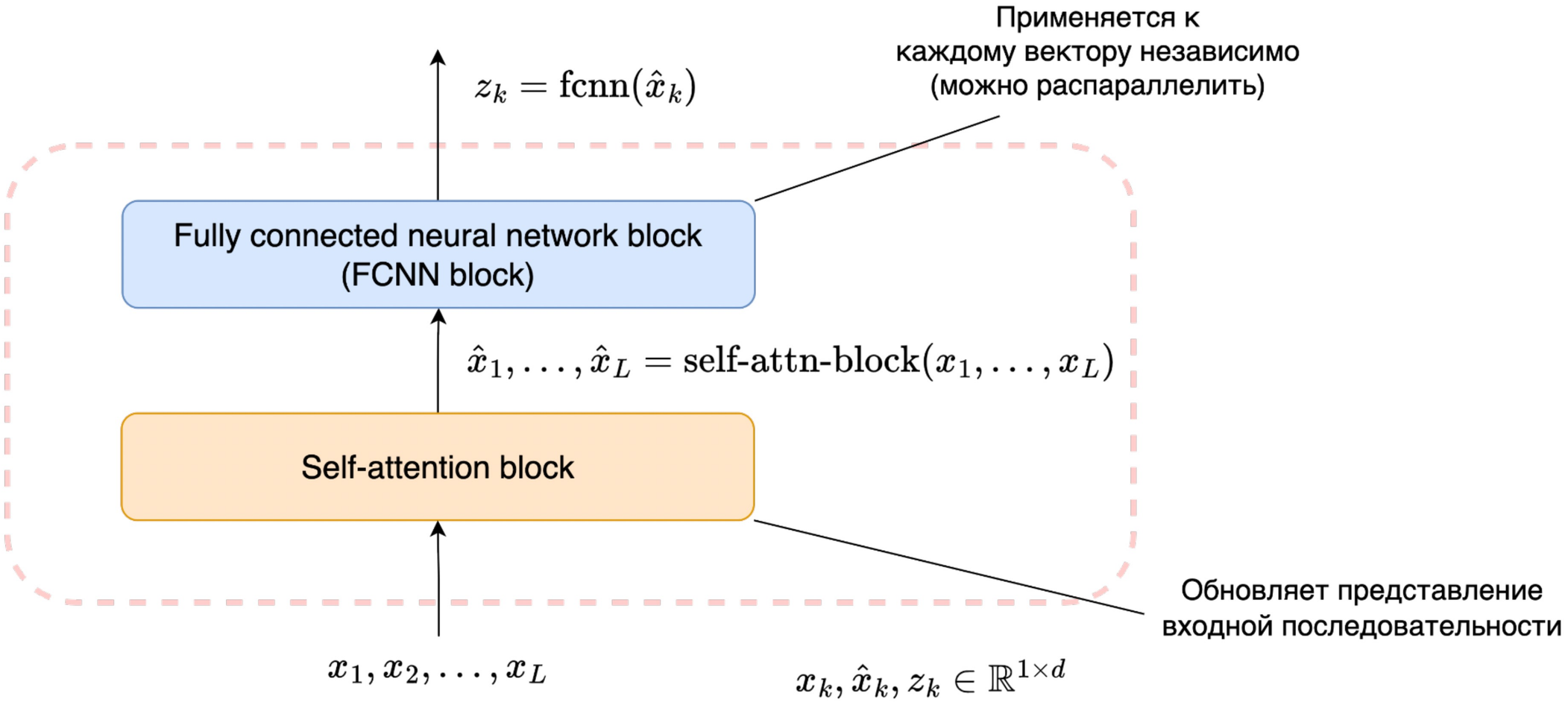
Кодировщик



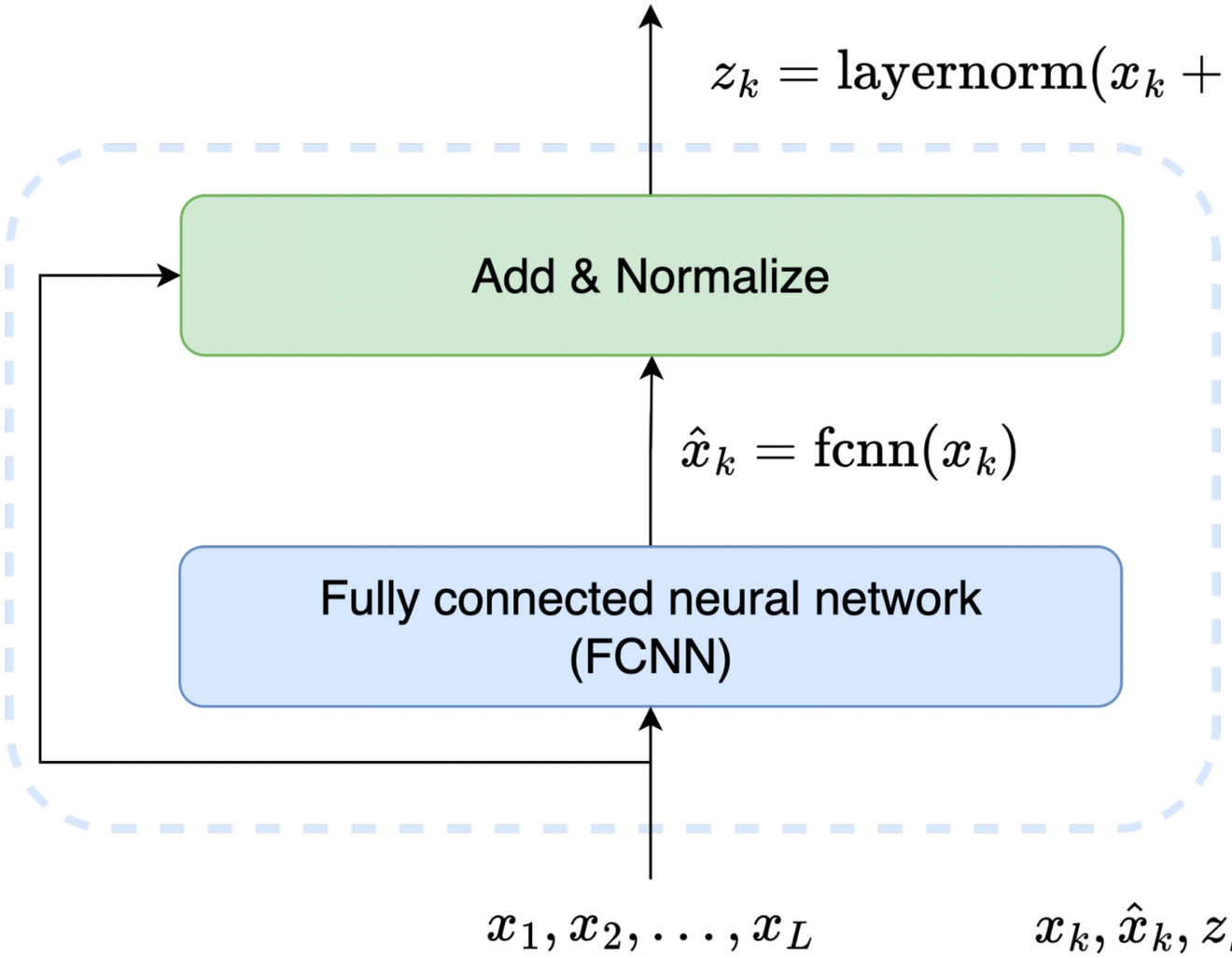
Кодировщик:

- Принимает на вход последовательность токенов
- Вычисляет эмбеддинги входных токенов (слой embeddings)
- Применяет последовательно N однотипных преобразований (encoder layer)

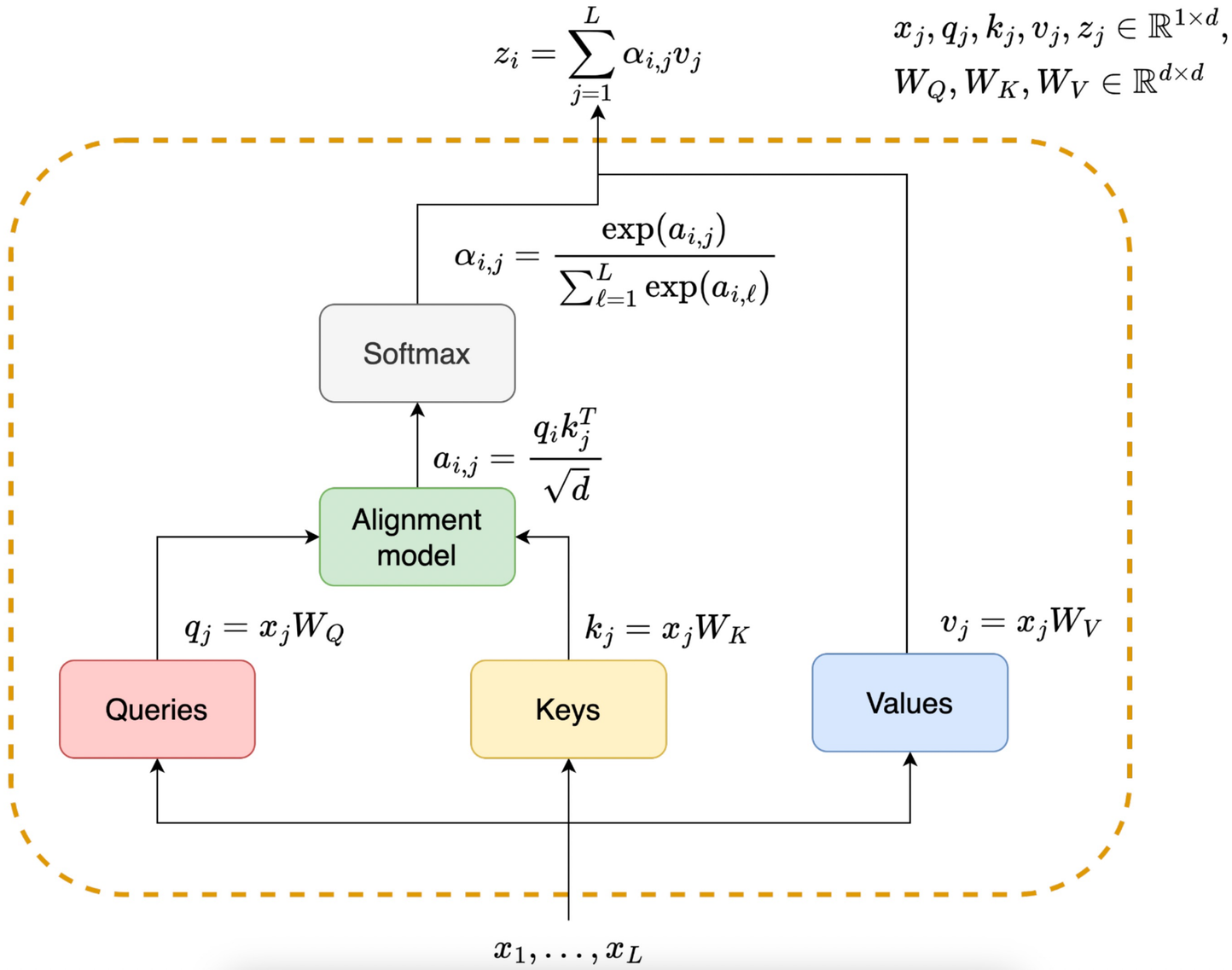
Слой кодировщика



Fully connected neural network

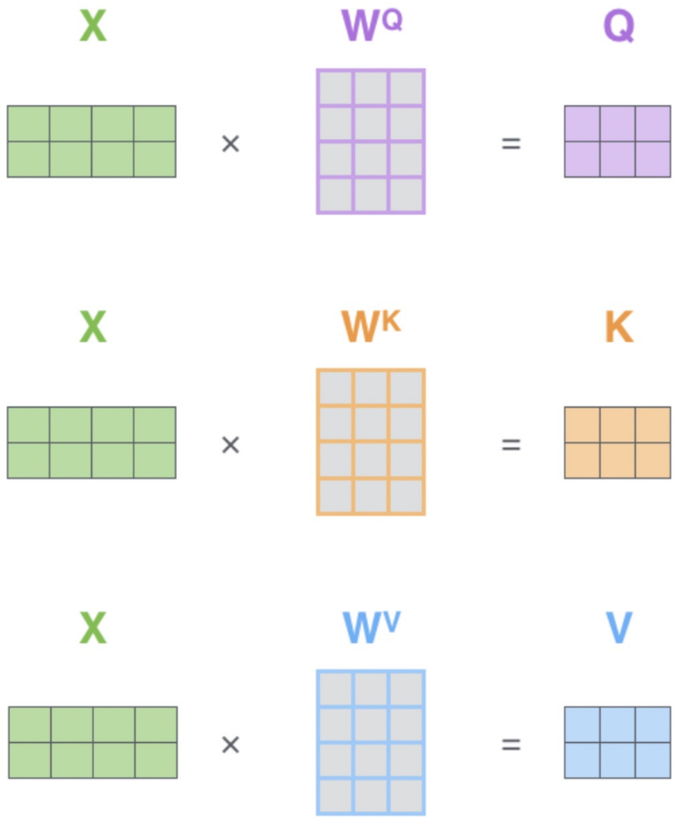


Self Attention



Self Attention

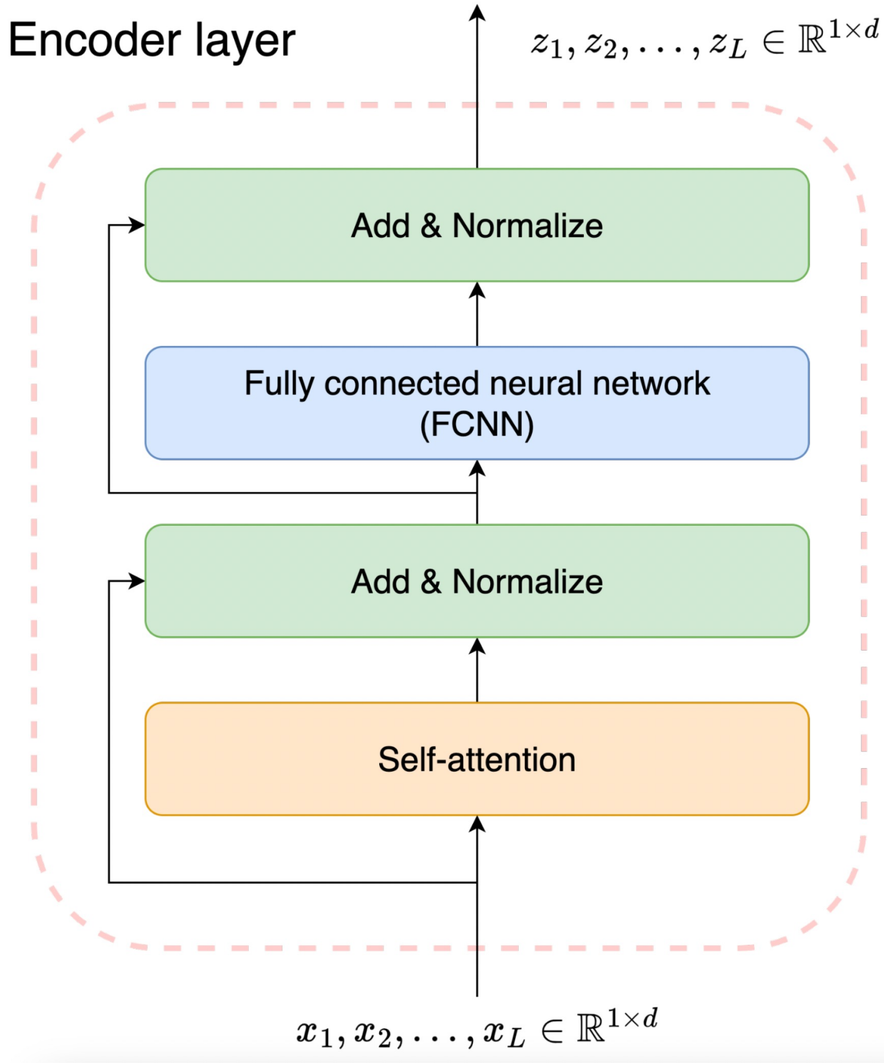
Получаем матрицы Q, K, V:



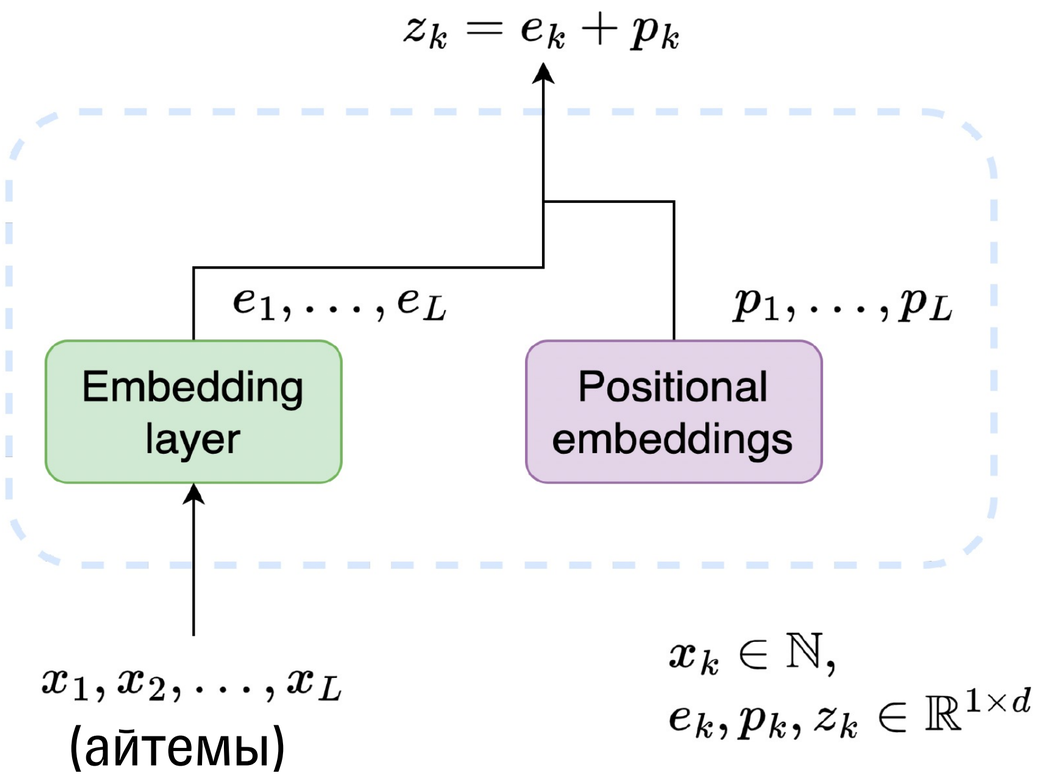
Считаем self-attention в матричной форме:

$$\text{softmax}\left(\frac{Q \times K^T}{\sqrt{d_k}}\right) V = Z$$

Слой кодировщика



А как учесть позицию?



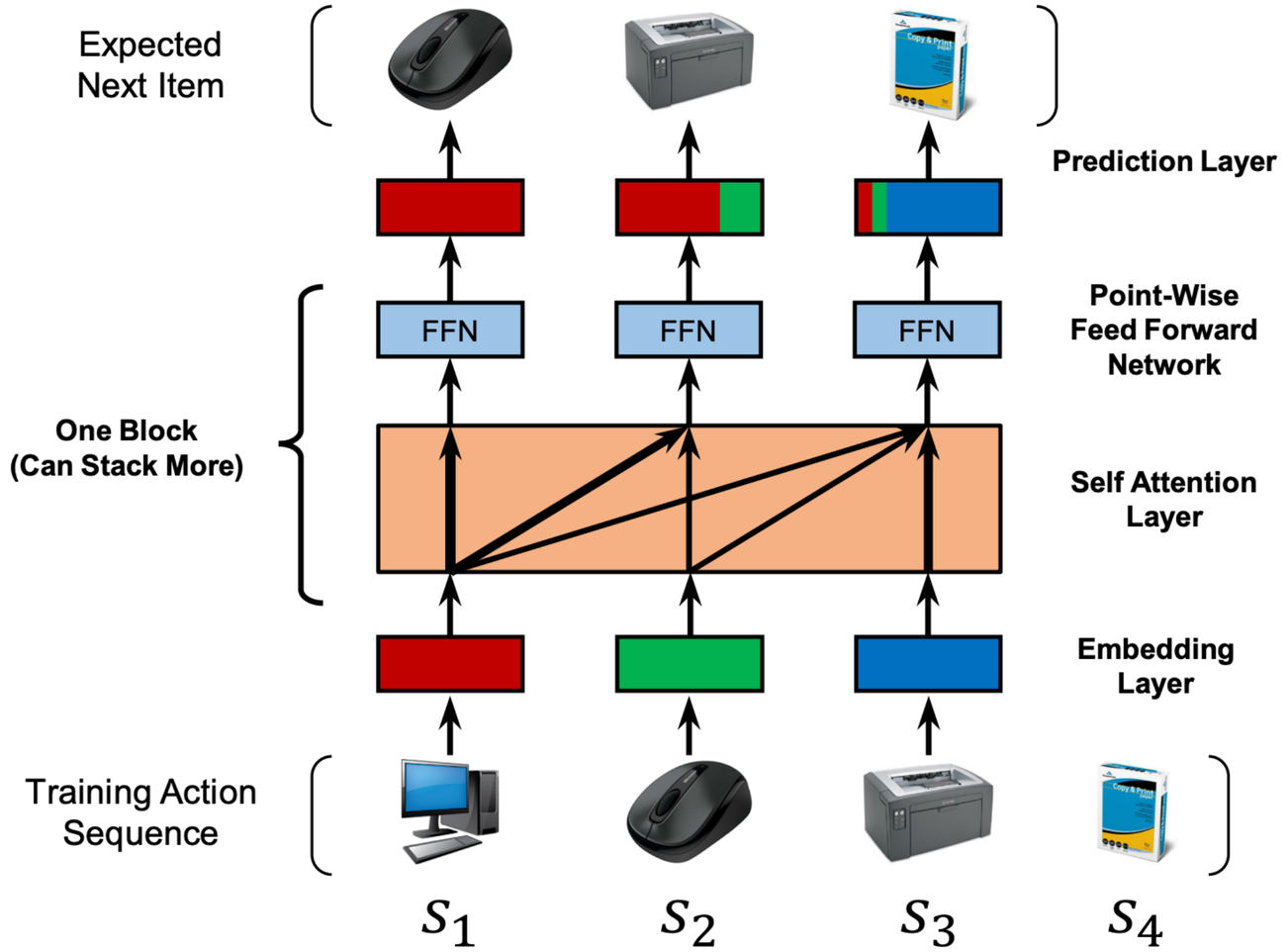
SASRec

Self-Attentive Sequential Recommendation

Wang-Cheng Kang, Julian McAuley
UC San Diego
{wckang,jmcauley}@ucsd.edu

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec



Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec: Детали

- Shared эмбеддинги айтемов на **входе** и на **выходе**
- BCELoss
- Обучаемые positional embeddings
- Маскируем аттеншн: не можем смотреть в будущее

$$L_{BCE} = -\sum_{\mathcal{S}^u \in \mathcal{S}} \sum_{t \in [1,2,\ldots,n]} \left[\log(\sigma(r_{o_t,t})) + \sum_{j \notin \mathcal{S}^u} \log(1 - \sigma(r_{j,t})) \right].$$

Note that we ignore the terms where $o_t =$ <pad>.

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

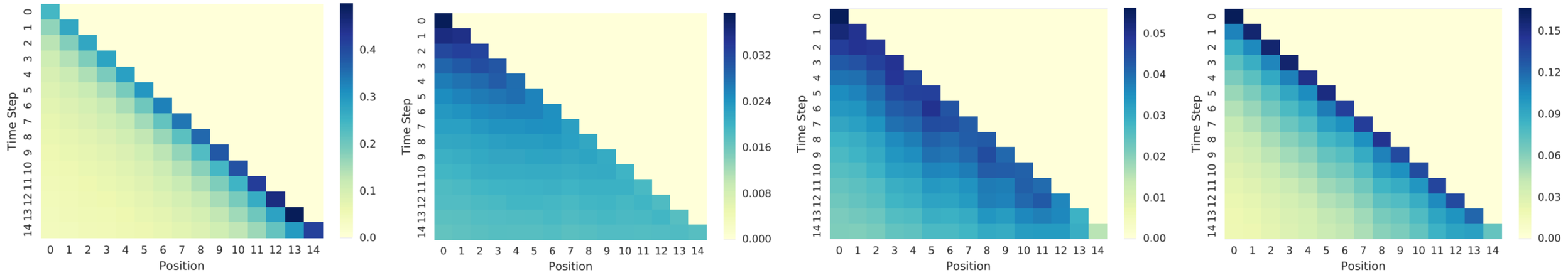
SASRec: Детали

Как посчитать релевантность i-го айтема:

- Прогоняем всю последовательность через модель
- Берем последний hidden layer
- Умножаем на $embedding_i$

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec: Интересности



(a) *Beauty*, Layer 1 (b) *ML-1M*, Layer 1, w/o PE (c) *ML-1M*, Layer 1 (d) *ML-1M*, Layer 2

Figure 4: Visualizations of average attention weights on positions at different time steps. For comparison, the heatmap of a first-order Markov chain based model would be a diagonal matrix.

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec: Интересности

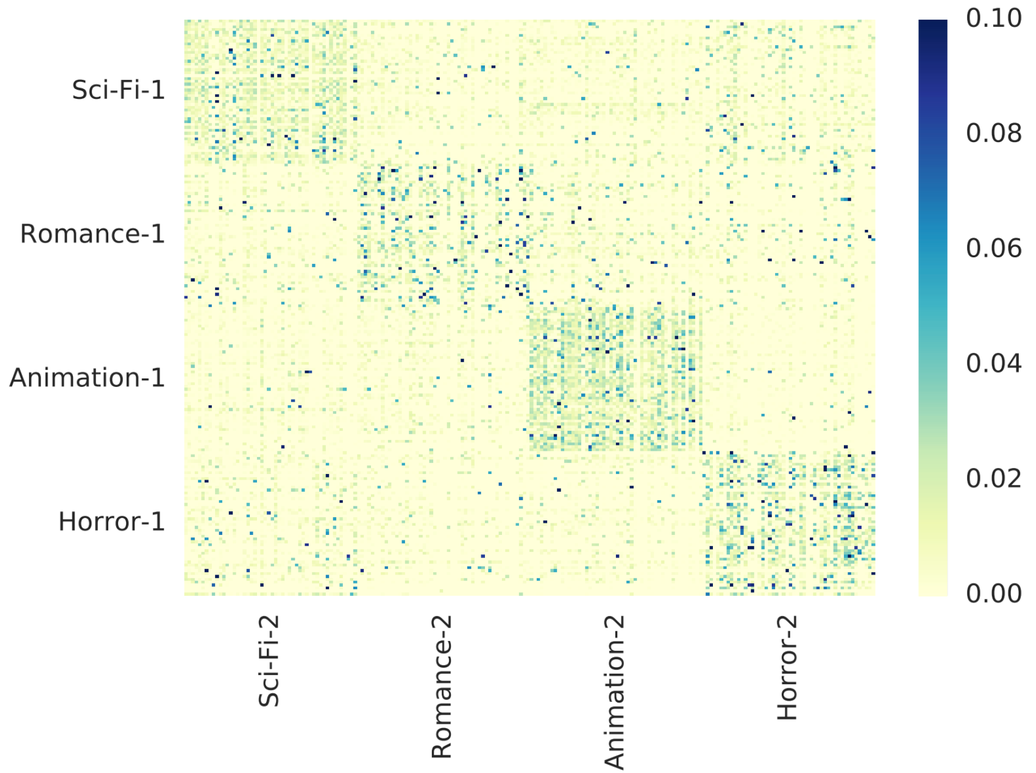


Figure 5: Visualization of average attention between movies from four categories. This shows our model can uncover items' attributes, and assigns larger weights between similar items.

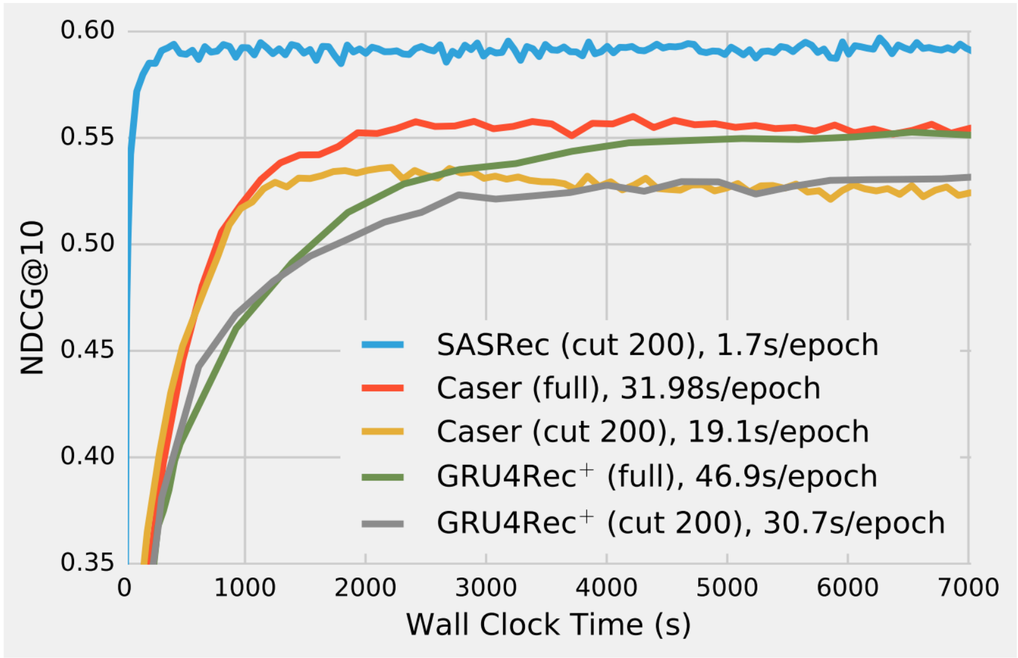


Figure 3: Training efficiency on *ML-1M*. SASRec is an order of magnitude faster than CNN/RNN-based recommendation methods in terms of training time per epoch and in total.

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec: Интересности

Architecture	*Beauty*	*Games*	*Steam*	*ML-1M*
(0) Default	0.3142	0.5360	0.6306	0.5905
(1) Remove PE	**0.3183**	0.5301	0.6036	0.5772
(2) Unshared IE	0.2437↓	0.4266↓	0.4472↓	0.4557↓
(3) Remove RC	0.2591↓	0.4303↓	0.5693	0.5535
(4) Remove Dropout	0.2436↓	0.4375↓	0.5959	0.5801
(5) 0 Block (b=0)	0.2620↓	0.4745↓	0.5588↓	0.4830↓
(6) 1 Block (b=1)	0.3066	**0.5408**	0.6202	0.5653
(7) 3 Blocks (b=3)	0.3078	0.5312	0.6275	**0.5931**
(8) Multi-Head	0.3080	0.5311	0.6272	0.5885

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec: Качество

Dataset	Metric	(a) PopRec	(b) BPR	(c) FMC	(d) FPMC	(e) TransRec	(f) GRU4Rec	(g) GRU4Rec$^+$	(h) Caser	(i) SASRec	Improvement vs. (a)-(e)	Improvement vs. (f)-(h)
Beauty	Hit@10	0.4003	0.3775	0.3771	0.4310	0.4607	0.2125	0.3949	0.4264	**0.4854**	5.4%	13.8%
	NDCG@10	0.2277	0.2183	0.2477	0.2891	0.3020	0.1203	0.2556	0.2547	**0.3219**	6.6%	25.9%
Games	Hit@10	0.4724	0.4853	0.6358	0.6802	0.6838	0.2938	0.6599	0.5282	**0.7410**	8.5%	12.3%
	NDCG@10	0.2779	0.2875	0.4456	0.4680	0.4557	0.1837	0.4759	0.3214	**0.5360**	14.5%	12.6%
Steam	Hit@10	0.7172	0.7061	0.7731	0.7710	0.7624	0.4190	0.8018	0.7874	**0.8729**	13.2%	8.9%
	NDCG@10	0.4535	0.4436	0.5193	0.5011	0.4852	0.2691	0.5595	0.5381	**0.6306**	21.4%	12.7%
ML-1M	Hit@10	0.4329	0.5781	0.6986	0.7599	0.6413	0.5581	0.7501	0.7886	**0.8245**	8.5%	4.6%
	NDCG@10	0.2377	0.3287	0.4676	0.5176	0.3969	0.3381	0.5513	0.5538	**0.5905**	14.1%	6.6%

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

SASRec: Минусы

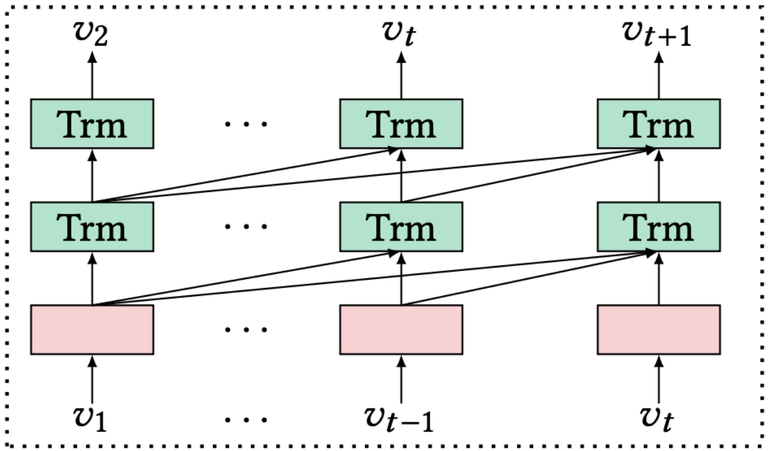
- Сомнительный протокол тестирования

> To avoid heavy computation on all user-item pairs, we followed the strategy in [14], [48]. For each user u, we randomly sample 100 negative items, and rank these items with the ground-truth item. Based on the rankings of these 101 items, Hit@10 and NDCG@10 can be evaluated.

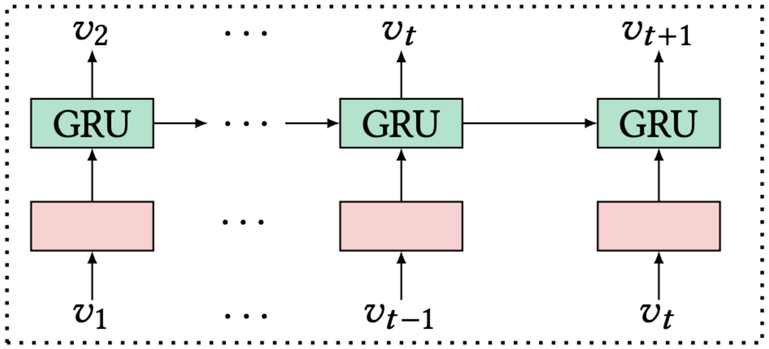
- «Слабая» функция потерь

Kang WC, McAuley J. Self-attentive sequential recommendation. In2018 IEEE international conference on data mining (ICDM) 2018 Nov 17 (pp. 197-206). IEEE.

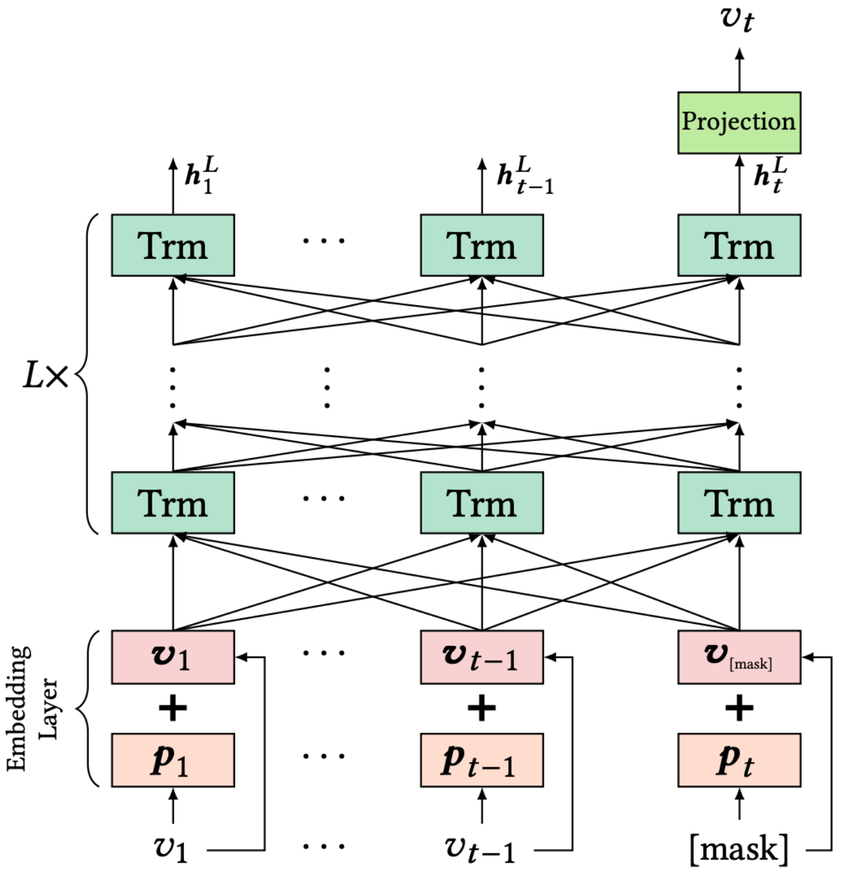
Introducing: BERT4Rec



(c) SASRec model architecture.



(d) RNN based sequential recommendation methods.



Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

Introducing: BERT4Rec

- Можем смотреть на всю последовательность
- Маскируем случайные элементы в ней
- Для предсказания добавляем маску в конец последовательности
- функция потерь – softmax

Input: $[v_1, v_2, v_3, v_4, v_5] \xrightarrow{\text{randomly mask}} [v_1, [\text{mask}]_1, v_3, [\text{mask}]_2, v_5]$

Labels: $[\text{mask}]_1 = v_2, \quad [\text{mask}]_2 = v_4$

The final hidden vectors corresponding to "[mask]" are fed into an output softmax over the item set, as in conventional sequential recommendation. Eventually, we define the loss for each masked input $\mathcal{S}'_u$ as the negative log-likelihood of the masked targets:

$$\mathcal{L} = \frac{1}{|\mathcal{S}^m_u|} \sum_{v_m \in \mathcal{S}^m_u} -\log P(v_m = v^*_m | \mathcal{S}'_u) \tag{8}$$

where $\mathcal{S}'_u$ is the masked version for user behavior history $\mathcal{S}_u$, $\mathcal{S}^m_u$ is the random masked items in it, v^*_m is the true item for the masked item v_m, and the probability $P(\cdot)$ is defined in Equation 7.

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Метрики

Datasets	Metric	POP	BPR-MF	NCF	FPMC	GRU4Rec	GRU4Rec$^+$	Caser	SASRec	BERT4Rec	Improv.
Beauty	HR@1	0.0077	0.0415	0.0407	0.0435	0.0402	0.0551	0.0475	0.0906	**0.0953**	5.19%
	HR@5	0.0392	0.1209	0.1305	0.1387	0.1315	0.1781	0.1625	0.1934	**0.2207**	14.12%
	HR@10	0.0762	0.1992	0.2142	0.2401	0.2343	0.2654	0.2590	0.2653	**0.3025**	14.02%
	NDCG@5	0.0230	0.0814	0.0855	0.0902	0.0812	0.1172	0.1050	0.1436	**0.1599**	11.35%
	NDCG@10	0.0349	0.1064	0.1124	0.1211	0.1074	0.1453	0.1360	0.1633	**0.1862**	14.02%
	MRR	0.0437	0.1006	0.1043	0.1056	0.1023	0.1299	0.1205	0.1536	**0.1701**	10.74%
Steam	HR@1	0.0159	0.0314	0.0246	0.0358	0.0574	0.0812	0.0495	0.0885	**0.0957**	8.14%
	HR@5	0.0805	0.1177	0.1203	0.1517	0.2171	0.2391	0.1766	0.2559	**0.2710**	5.90%
	HR@10	0.1389	0.1993	0.2169	0.2551	0.3313	0.3594	0.2870	0.3783	**0.4013**	6.08%
	NDCG@5	0.0477	0.0744	0.0717	0.0945	0.1370	0.1613	0.1131	0.1727	**0.1842**	6.66%
	NDCG@10	0.0665	0.1005	0.1026	0.1283	0.1802	0.2053	0.1484	0.2147	**0.2261**	5.31%
	MRR	0.0669	0.0942	0.0932	0.1139	0.1420	0.1757	0.1305	0.1874	**0.1949**	4.00%
ML-1m	HR@1	0.0141	0.0914	0.0397	0.1386	0.1583	0.2092	0.2194	0.2351	**0.2863**	21.78%
	HR@5	0.0715	0.2866	0.1932	0.4297	0.4673	0.5103	0.5353	0.5434	**0.5876**	8.13%
	HR@10	0.1358	0.4301	0.3477	0.5946	0.6207	0.6351	0.6692	0.6629	**0.6970**	4.15%
	NDCG@5	0.0416	0.1903	0.1146	0.2885	0.3196	0.3705	0.3832	0.3980	**0.4454**	11.91%
	NDCG@10	0.0621	0.2365	0.1640	0.3439	0.3627	0.4064	0.4268	0.4368	**0.4818**	10.32%
	MRR	0.0627	0.2009	0.1358	0.2891	0.3041	0.3462	0.3648	0.3790	**0.4254**	12.24%
ML-20m	HR@1	0.0221	0.0553	0.0231	0.1079	0.1459	0.2021	0.1232	0.2544	**0.3440**	35.22%
	HR@5	0.0805	0.2128	0.1358	0.3601	0.4657	0.5118	0.3804	0.5727	**0.6323**	10.41%
	HR@10	0.1378	0.3538	0.2922	0.5201	0.5844	0.6524	0.5427	0.7136	**0.7473**	4.72%
	NDCG@5	0.0511	0.1332	0.0771	0.2239	0.3090	0.3630	0.2538	0.4208	**0.4967**	18.04%
	NDCG@10	0.0695	0.1786	0.1271	0.2895	0.3637	0.4087	0.3062	0.4665	**0.5340**	14.47%
	MRR	0.0709	0.1503	0.1072	0.2273	0.2967	0.3476	0.2529	0.4026	**0.4785**	18.85%

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Метрики

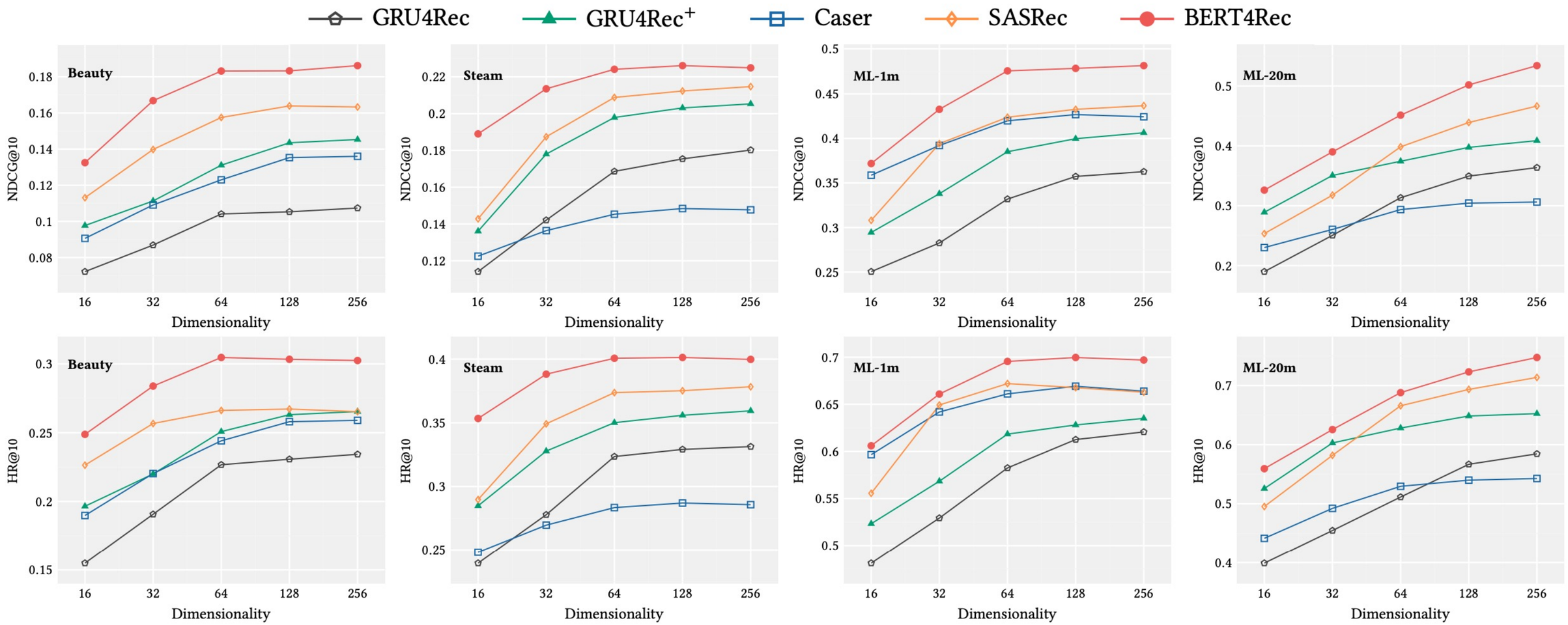
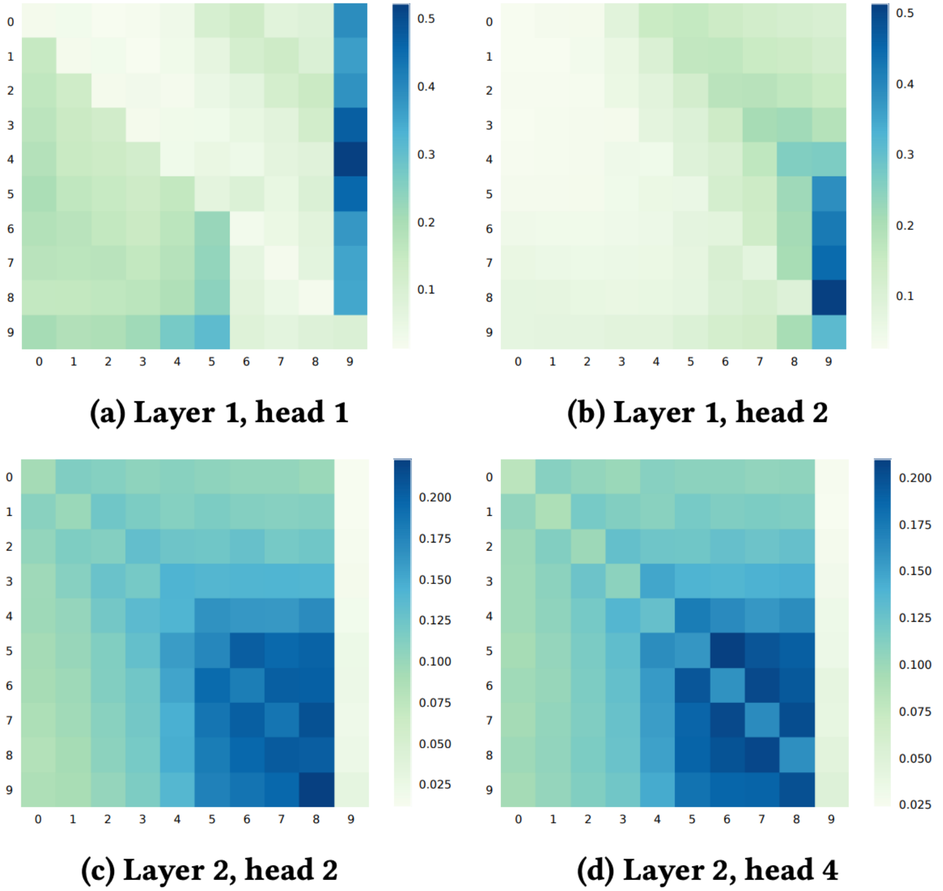


Figure 3: Effect of the hidden dimensionality d on HR@10 and NDCG@10 for neural sequential models.

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Детали



(a) Layer 1, head 1 (b) Layer 1, head 2

(c) Layer 2, head 2 (d) Layer 2, head 4

Figure 2: Heat-maps of average attention weights on Beauty, the last position "9" denotes "[mask]" (best viewed in color).

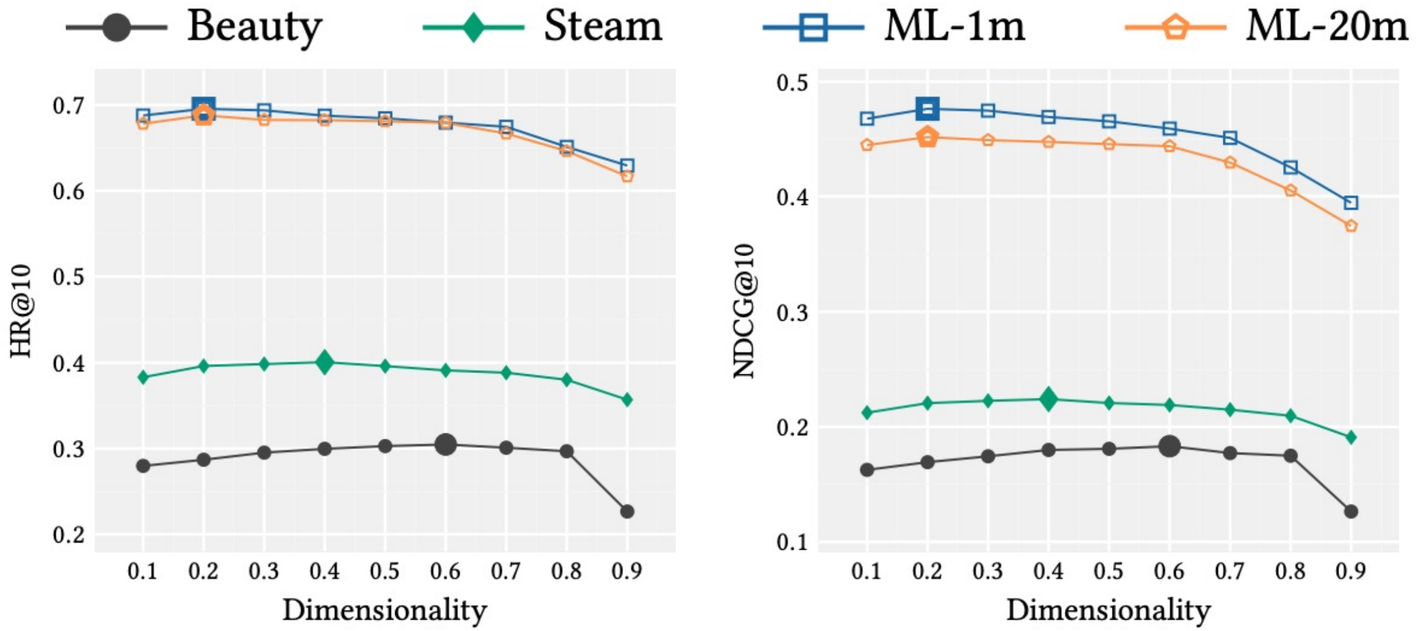


Figure 4: Performance with different mask proportion ρ on $d = 64$. Bold symbols denote the best scores in each line.

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Детали

Table 5: Ablation analysis (NDCG@10) on four datasets. Bold score indicates performance better than the default version, while ↓ indicates performance drop more than 10%.

Architecture	Dataset			
	Beauty	Steam	ML-1m	ML-20m
$L = 2, h = 2$	0.1832	0.2241	0.4759	0.4513
w/o PE	0.1741	0.2060	0.2155↓	0.2867↓
w/o PFFN	0.1803	0.2137	0.4544	0.4296
w/o LN	0.1642↓	0.2058	0.4334	0.4186
w/o RC	0.1619↓	0.2193	0.4643	0.4483
w/o Dropout	0.1658	0.2185	0.4553	0.4471
1 layer $(L = 1)$	0.1782	0.2122	0.4412	0.4238
3 layers $(L = 3)$	**0.1859**	**0.2262**	**0.4864**	**0.4661**
4 layers $(L = 4)$	**0.1834**	**0.2279**	**0.4898**	**0.4732**
1 head $(h = 1)$	**0.1853**	0.2187	0.4568	0.4402
4 heads $(h = 4)$	0.1830	**0.2245**	**0.4770**	**0.4520**
8 heads $(h = 8)$	0.1823	**0.2248**	0.4743	**0.4550**

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Не все так сладко

- Тяжело масштабировать. Почему?

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Не все так сладко

- Тяжело масштабировать. Почему? Из-за **Softmax**

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

BERT4Rec: Не все так сладко

- Тяжело масштабировать. Почему? Из-за **Softmax**
- Все еще **странный** протокол тестирования

4.2 Task Settings & Evaluation Metrics

To evaluate the sequential recommendation models, we adopted the *leave-one-out* evaluation (*i.e.*, next item recommendation) task, which has been widely used in [12, 22, 49]. For each user, we hold out the last item of the behavior sequence as the test data, treat the item just before the last as the validation set, and utilize the remaining items for training. For easy and fair evaluation, we follow the common strategy in [12, 22, 49], pairing each ground truth item in the test set with 100 randomly sampled *negative* items that the user has not interacted with. To make the sampling reliable and representative [19], these 100 negative items are sampled according to their popularity. Hence, the task becomes to rank these negative items with the ground truth item for each user.

Sun F, Liu J, Wu J, Pei C, Lin X, Ou W, Jiang P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer.

Какие мы выделили проблемы?

1. Сомнительный протокол тестирования
2. Плохая масштабируемость
3. Завышенные скоры у топовых айтемов

Завышенные скоры у топовых айтемов ведут к тому, что модель не учится отличать эти айтемы друг от друга (все их
Скоры приблизительно равны) и фокусируется на умение отличить лучшие от худших.

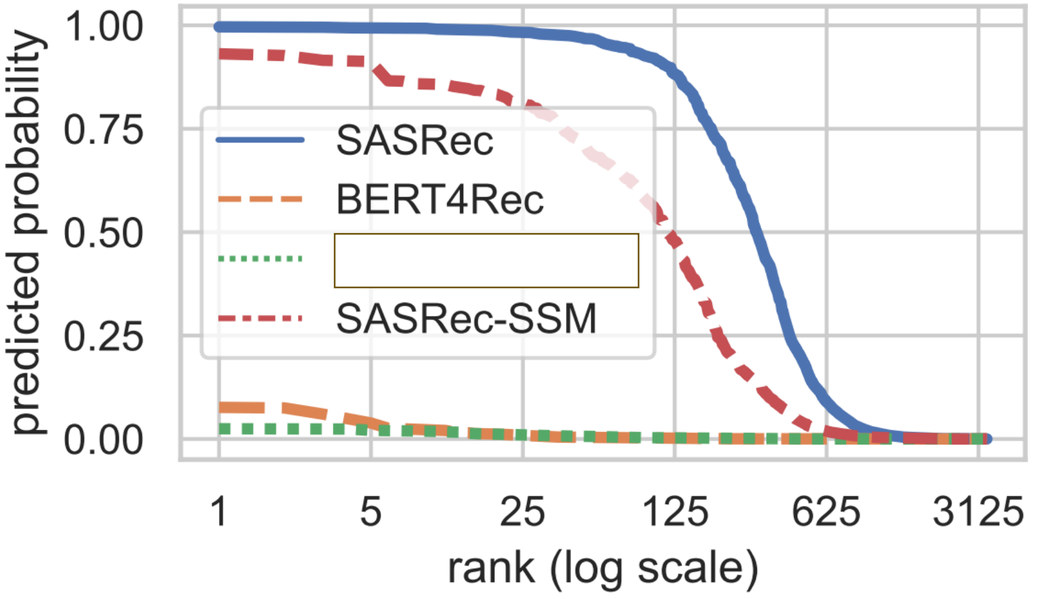


Figure 1: Predicted probability at different ranks for user 963 in MovieLens-1M. SASRec-SSM is a SASRec model trained with Sampled Softmax loss with 16 negatives.

Petrov AV, Macdonald C. gsasrec: Reducing overconfidence in sequential recommendation trained with negative sampling.

SASRec: Обобщение

gSASRec: Reducing Overconfidence in Sequential Recommendation Trained with Negative Sampling

Aleksandr Petrov
University of Glasgow
United Kingdom
a.petrov.1@research.gla.ac.uk

Craig Macdonald
University of Glasgow
United Kingdom
craig.macdonald@glasgow.ac.uk

$$\mathcal{L}^{\beta}_{\text{gBCE}} = -\frac{1}{|I_k^-| + 1}\left(\log(\sigma^{\beta}(s_{i^+})) + \sum_{i \in I_k^-} \log(1 - \sigma(s_i))\right)$$

$$\alpha = \frac{1}{|I| - 1}$$

$$\beta = \alpha\left(t\left(1 - \frac{1}{\alpha}\right) + \frac{1}{\alpha}\right)$$

Petrov AV, Macdonald C. gsasrec: Reducing overconfidence in sequential recommendation trained with negative sampling.

SASRec: Обобщение

Dataset	Negative sampling and loss function→ Architecture↓	1 negative per positive; BCE Loss (as SASRec)	No negative sampling; Softmax Loss (as BERT4Rec)	**Negative sampling and loss effect**
ML-1M	SASRec	0.131	0.169	**+29.0%***
	BERT4Rec	0.123	0.161	**+30.8%***
	Architecture effect	**-6.1%**	**-4.7%**	
Steam	SASRec	0.0581	0.0721	**+24.1%***
	BERT4Rec	0.0513	0.0746	**+45.4%***
	Architecture effect	**-11.7%***	**+3.4%***	

Petrov AV, Macdonald C. gsasrec: Reducing overconfidence in sequential recommendation trained with negative sampling.

SASRec: Обобщение

6.1.2 Metrics. Until recently, a somewhat common approach in evaluating recommender systems on sampled metrics using only small number of items, but it has been shown that this leads to incorrect evaluation results in general [3, 18], and specifically for sequential recommender systems [7, 25]. Hence, we always evaluate all item scores at the inference stage. Following [25], we evaluate our models using the popular Recall and NDCG metrics measured at cutoff 10. We also calculate Recall at cutoff 1, because according to Equation (22), we expect SASRec to be more overconfident on the highest-ranked metrics, and mitigating overconfidence should have a bigger effect on metrics measured at the highest cutoff.

Model				Datasets											
				MovieLens-1M				Steam				Gowalla			
Category	Model	Negative Sampling	Loss	Recall @1	Recall @10	NDCG @10	Time (min)	Recall @1	Recall @10	NDCG @10	Time (min)	Recall @1	Recall @10	NDCG @10	Time (min)
Baselines	Popularity	N/A	N/A	0.005*	0.036*	0.017*	0.0	0.0077*	0.0529*	0.0268*	0.0	0.0011*	0.0081*	0.0041*	0.1
	MF-BPR	Yes	BPR	0.010*	0.075*	0.037*	0.1	0.0071*	0.0393*	0.0206*	0.4	0.0083*	0.0282*	0.0170*	2.1
Unsampled	BERT4Rec	No	Softmax	0.058*	0.294	0.161*	86	0.0281	**0.1379**	**0.0746**	642	Infeasible (requires >100GB of GPU memory, see Section 1)			
	SASRec-Softmax	No	Softmax	0.073	0.293	0.169	9	0.0280	0.1323*	0.0721*	80				
Sampled	SASRec	Yes	BCE	0.046*	0.247*	0.131*	13	0.0193*	0.1121*	0.0581*	32	0.0505*	0.1831*	0.1097*	145
	gSASRec	Yes	gBCE	**0.082**	**0.300**	**0.176**	23	**0.0283**	0.1355	0.0735	58	**0.0782**	**0.2590**	**0.1616**	191

Petrov AV, Macdonald C. gsasrec: Reducing overconfidence in sequential recommendation trained with negative sampling.

SASRec: Обобщение

- Способ обучение важнее модели

- Измеряйте метрики правильно

Что почитать?

- Объяснение трансформера:
  https://jalammar.github.io/illustrated-transformer/

- Подробное объяснение трансформера:
  https://nlp.seas.harvard.edu/2018/04/03/attention.html

Спасибо за внимание