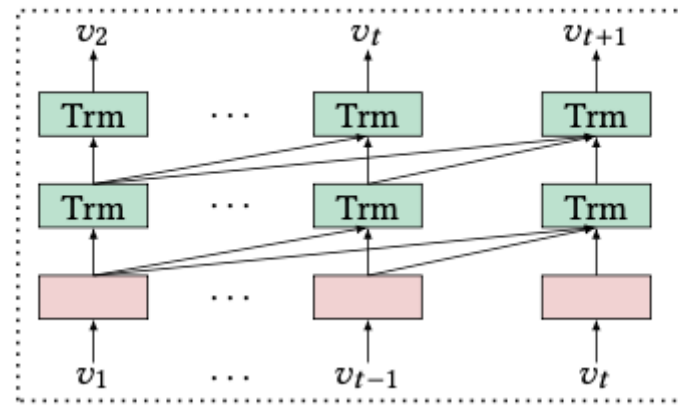


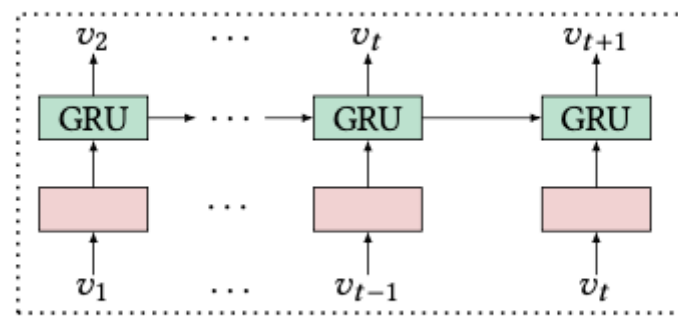
BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer

Introduction

- Sequential Recommendation System은 유저의 과거 행동 패턴을 학습하여 이후에 유저가 구매할 아이템을 추천하는 모델
- 기존의 Sequential Recommendation System (SASRec, GRU4Rec)은 단방향 모델 (left-to-right unidirectional model)로 과거에 구매한 아이템 정보 만을 학습하여 유저의 이전 행동패턴 만을 모델이 고려하기 때문에 성능에 한계가 생김
 - 표현력 한계
 - 순서가 무의미한 경우도 순서를 정함
- 본 논문은 BERT의 Masked Language Model을 활용하여 다음 아이템을 예측하는 **BERT4Rec**을 제안



(c) SASRec model architecture.



(d) RNN based sequential recommendation methods.

기존 단방향(unidirectional) 추천 시스템 모델

BERT4Rec

Problem Statement

- 유저 벡터

$$\mathcal{U}=\{u_1, u_2, \dots, u_{|\mathcal{U}|}\}$$

- 아이템 벡터

$$\mathcal{V}=\{v_1, v_2, \dots, v_{|\mathcal{V}|}\}$$

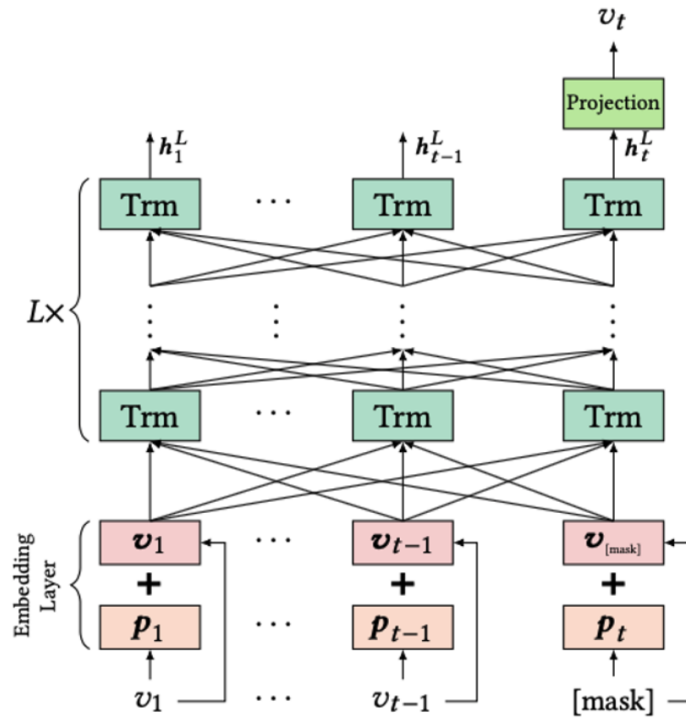
- Interaction sequence: 특정 유저 u 가 t 시점에 상호작용하는 아이템 시퀀스

$$\mathcal{S}_u = [v_1^{(u)}, \dots, v_t^{(u)}, \dots, v_{n_u}^{(u)}]$$

- 이전 n_u 시점까지의 정보를 통해 $n_u + 1$ 시점에서 사용자가 어떤 아이템과 상호작용할지 예측하는 문제

$$p(v_{n_u+1}^{(u)} = v | \mathcal{S}_u)$$

Model Architecture



- BERT4Rec은 L 개의 양방향 Transformer layers로 이루어짐
- 병렬적으로 이전 layer에 존재하는 모든 위치 정보들을 상호 교환하여 각 위치의 representation을 반복 수정함으로써 학습을 진행
- 한쪽으로만 정보를 전달하는 기존의 RNN 기반의 방법과 달리 Self-Attention을 활용하여 거리가 멀어져도 의존도를 반영할 수 있음

Transformer Layer

- Transformer layer는 multi-head self-attention과 position-wise feed-forward network로 구성
- **Multi-Head Self Attention**

$$\text{MH}(H^l) = [\text{head}_1; \text{head}_2; \dots; \text{head}_h] W^O$$

$$\text{head}_i = \text{Attention}(H^l W_i^Q, H^l W_i^K, H^l W_i^V)$$

- Scaled Dot-Product Attention 수식

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d/h}}\right)V$$

- **Position-Wise Feed Forward Network**

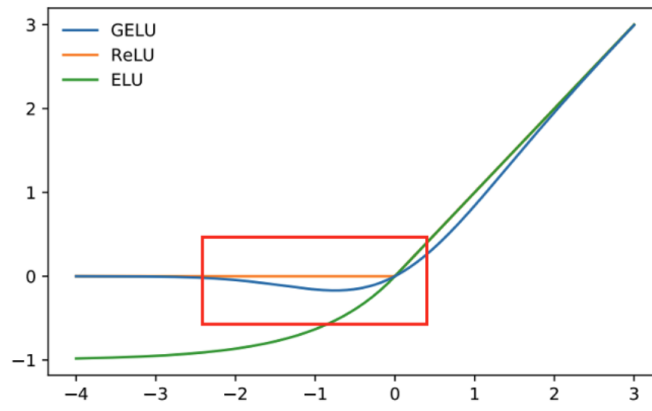
$$\text{PFFN}(H^l) = [\text{FFN}(h_1^l)^\top; \dots; \text{FFN}(h_t^l)^\top]^\top$$

$$\text{FFN}(x) = \text{GELU}(xW^{(1)} + b^{(1)})W^{(2)} + b^{(2)}$$

$$\text{GELU}(x) = x\Phi(x)$$

- Multi-Head Self-Attention을 통해 나온 결과 값은 linear projection 이기 때문에 FC Layer에 GELU (Gaussian Error Linear Unit) 함수를 활성화 함수로 사용하여 비선형성을 줌

- $\phi(x)$ 는 표준 정규 분포의 누적 분포 함수이며 학습가능한 파라미터로 모든 위치에서 사용됨
- GELU는 BERT 모델에 사용되는 활성화 함수로 음수에서 미분이 가능함



- 여러 Transformer layer를 쌓아 너무 깊어지면 학습하기 어려우므로 residual connection과 layer normalization, dropout을 함께 사용

$$\begin{aligned}
 H^l &= \text{Trm}(H^{l-1}), \quad \forall i \in [1, \dots, L] \\
 \text{Trm}(H^{l-1}) &= \text{LN}(A^{l-1} + \text{Dropout}(\text{PFFN}(A^{l-1}))) \\
 A^{l-1} &= \text{LN}(H^{l-1} + \text{Dropout}(\text{MH}(H^{l-1})))
 \end{aligned}$$

Embedding Layer

- Transformer에는 위치 정보가 포함되지 않기 때문에 Positional Embedding Layer를 추가 함

$$h_i^0 = v_i + p_i$$

- 모델이 처리할 수 있는 최대 길이를 제한하여 길이 N을 넘는 sequence는 마지막 N개에 대해서만 학습을 진행

Output Layer

- L개의 layer를 통과하고 나면 다음과 같은 결과물을 얻을 수 있음

$$P(v) = \text{softmax}(\text{GELU}(h_t^L W^P + b^P) E^T + b^O)$$

Model Learning

- 모델을 학습할 때는 BERT처럼 [MASK]를 씌우고 이를 맞히게 하는 Cloze Task (Masked Language Model)를 진행하며, [MASK]가 여러 개인 경우에는 순차적으로 진행함

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

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

- Negative Log Likelihood를 이용하여 label과 예측값 간의 loss를 정의

$$\mathcal{L} = \frac{1}{|S_u^m|} \sum_{v_m \in S_u^m} -\log P(v_m = v_m^* | S_u')$$

- S'_u 는 유저 S_u 에 masked를 한 것이고 v_m^* 은 masked된 item으로 v_m 의 target item 임
- Cloze task의 또 하나의 장점은 하나의 sequence에서 여러 학습 샘플을 뽑을 수 있음
- 실제로 테스트 단계에서는 마지막 토큰만 [MASK]처리해서 추론함
 - 학습 단계와 테스트 단계에서의 Task의 목적이 다름

Experiments

- 4개의 데이터 셋 (Beauty, Steam, ML-1m, ML-20m)으로 실험을 진행

Table 1: Statistics of datasets.

Datasets	#users	#items	#actions	Avg. length	Density
Beauty	40,226	54,542	0.35m	8.8	0.02%
Steam	281,428	13,044	3.5m	12.4	0.10%
ML-1m	6040	3416	1.0m	163.5	4.79%
ML-20m	138,493	26,744	20m	144.4	0.54%

데이터 통계량

- BERT4Rec이 모든 데이터셋에서 좋은 성능을 보이고 있음
- sequential recommendation 방법(FPMC, GRU4Rec)은 non-sequential(BPR-MF와 NCF)를 능가함

Table 2: Performance comparison of different methods on next-item prediction. Bold scores are the best in each row, while underlined scores are the second best. Improvements over baselines are statistically significant with $p < 0.01$.

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

모델 성능 비교