Knowledge Graph Attention for Sequential Recommendations

Tags	
Column	
Column 1	

Abstract

- SR system : historical interactions 에 기반한 user의 dynamic preferences 을 modeling. short-term과 long-term behavior를 modeling하는 것은 nontrivial 하고 challenging함.
- KATRec(Knowledge Aware aTtentive sequential Recommendations) : enhanced knowledge graph 를 사용해 해결
- knowledge graph attention network를 통해 sequence of interacted items을 modeling 하고 pre-existing side information을 leveraging 하여 short and long-term interests of users을 학습
    - entity-level에서의 item multi-relations 과 item-level에서의 users' dynamic sequences를 포함
- higher-order connections 을 고려하여 item representation learning
- next item을 추천하는 동안 user preference representation를 incorporating
- three public datasets 으로 실험 > SOTA
- high-quality recommendations achieve 을 위해 temporal 하고 side information을 modeling 하는 것이 중요

1. introduction

- 선행연구 : Markov chains, recurrent neural networks (RNN), graph convolutional neural networks (GCN), and self-attention based models (to name a few), primarily focus on various ways to model such historical data
- 그러나, 이 sequential models은 items간 relationship을 disregard 하는 경향이 있음. 특히, platforms 은 가치있는 recommendations을 위해 항상 two types of

information에 접근 가능해야 함.

1) time에 따라 진화하는 users 와 service의 interactions

2) users에 대한 side information, items, other auxiliary components.

- 논문은 knowledge graph에 two types of information 를 포함해서 build

1) Users, items, and other entities are the nodes

2) interactions and other relations are the edges of the knowledge graph

- user의 temporal data와 side information 은 primary gap이 있는데 논문에선 이를 채움
  - 선행논문에도 성능이 우수한 솔루션이 많이 있지만 두 가지 유형의 정보를 모두 효과적으로 포착하지는 못함.
  - 예) BERT4Rec, SASRec 및 GRU4Rec[9, 14, 24]과 같은 모델은 사용자 행동 시퀀스를 인코딩하여 temporal aspect에 초점.
  - 또 다른 선행논문은 item relations와 side information를 포착하기 위한 capture structure에 초점. TransE [1], TransH [31], TransR [18].
  - 논문에서 이 두 가지 선행연구를 기반으로 통합하는 새로운 방법을 제시. 추천 품질에 대한 두 가지 유형의 정보를 모두 포착하는 것의 중요성을 체계적으로 탐구함으로써 이를 추적
- 두 개의 module로 구성

1) a bidirectional transformer : 어떤 temporal distance에서도 items 간의 inter-dependencies가 고려됨으로써 sequential interests를 captures

2) a knowledge graph attention network 는 higher-order user-item와 item-item relations을 modeling.

- user-item relations는 클릭, 구매, 보기 등과 같은 항목과 사용자의 상호 작용을 기반으로 하며 collaborative information를 캡처.
- item-item relations는 shared entities(예: 배우 또는 장르가 동일한 영화)를 기반으로 item 간의 semantic relatedness을 포착. sequential recommendations를 하면서 그러한 정보를 포착하는 것의 중요성은 이전에 Ji et al에서 논의됨. [13], Ma et al. [19], Zhang et al. [34] to name a few.
- first-order relations 외에도 KATRec에 의해 생성된 sequential recommendations은 higher-order connections와 items 간의 relations 를 사용할 수 있음.

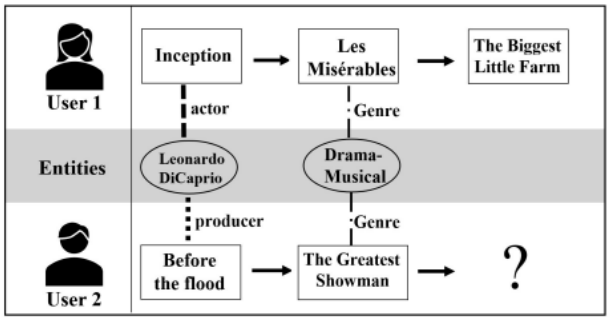


Fig. 1. Sequential interactions of two users with the systems. While users interact with different items, their collaborative signal can be detected via shared entities.

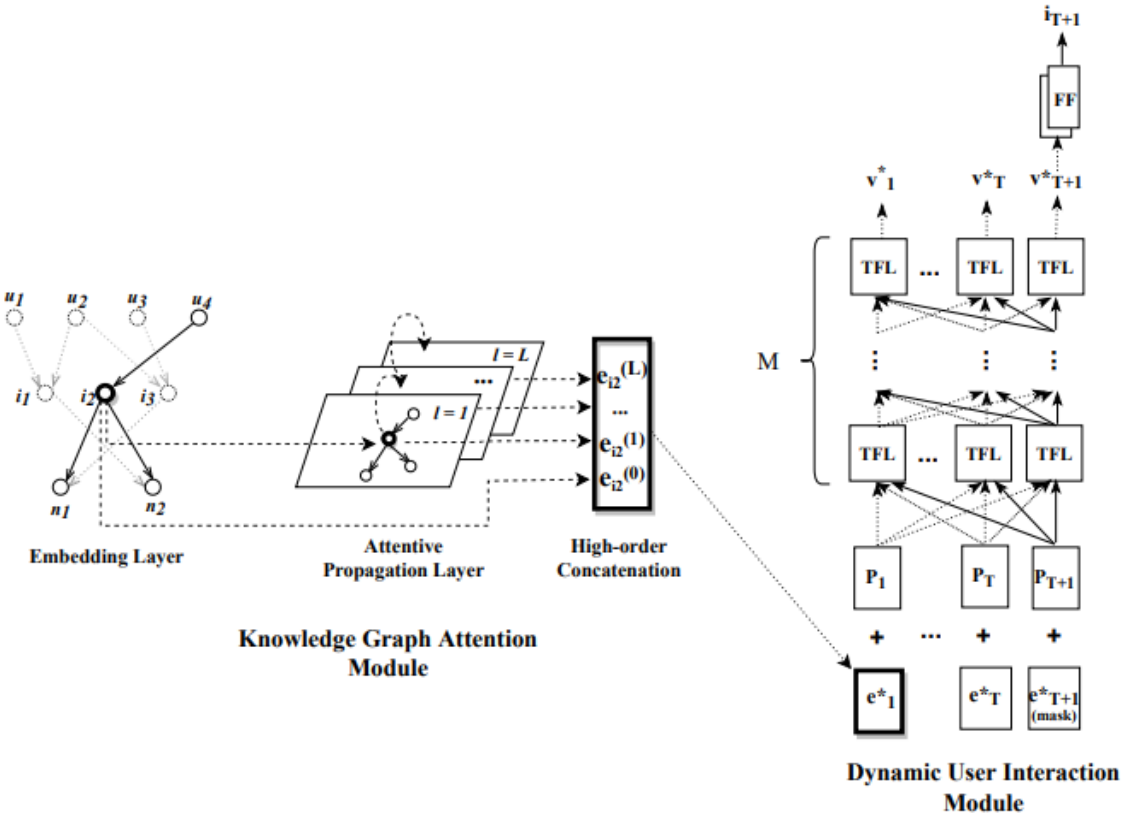
- 그림 1은 사용자 상호작용의 두 가지 순서를 보여줌. traditional sequential recommendation system은 이 두 사용자가 서로 다른 영화와 상호 작용하기 때문에 다르게 모델링하지만, 이러한 사용자는 shared entities(genre, actor, producer, etc.).가 있는 영화와 유사한 관심을 보임. 따라서 knowledge graph를 사용하여 정보를 통합하면 representations이 향상되고 결과적으로 권장 성능이 향상될 수 있음. Wang et al. [29], Zhao et al. [35].
- 이 연구는 sequential behavior of users과 side information about items를 모두 포함하는 novel deep neural network를 제안.
- 제안한 구조는 지식 그래프 주의 메커니즘을 통해 sequential attention mechanis과 spatial information를 사용하여 temporal information를 캡처.
- 주요 기여
    - KATRec builds on a **knowledge graph neural network** that captures **multi-relationships** between items by tying them together using the underlying entities. This allows for better representations of items and enhances the recommendation performance.
    - KATRec models the **short-term and long-term user preferences** by adaptively aggregating dynamic interactions and item-item multi-relations through a gating mechanism. This mechanism can significantly alleviate the sparsity in both user sequences and item relationships.
    - KATRec captures **co-occurrence information** and **collaborative signals** by leveraging attention mechanisms in the knowledge graph, which ultimately impact the attention weights in the bidirectional transformer module and the overall recommendations.
    - We conduct experiments on GPU to optimize training and evaluate the impact of different components on the performance of KATRec. Our

experiments show that our model outperforms state-of-the-art baselines on three public datasets

3. METHOD

3.1 Problem Definition and Solution Overview

- user $\mathcal{U}$ 와 item $\mathcal{I}$, item 의 sequence는 $S^u = \{S_1^u, \cdots, S_T^u\}$는 user u가 T time step동안 interacted 한 items. ($T = |\mathcal{S}^u|$)
- items과 관련있는 side information에 access (e.g. actors, directors, and genre)
- historical interaction sequence $\mathcal{S}^u$, 목표 : next time step $T+1$ 에서 user u의 관심 item을 예측
- KATRec에서는, items간 multi-relations을 캡쳐하는 동안 users의 dynamic behavior을 효율적으로 모델링하는 knowledge-aware attentive sequential recommendation system 을 구축



- 두 modules

(a) Knowledge Graph Attention module : a graph neural network **G** 는 item level multi-relations을 captures

(b) Dynamic user Interaction Moduel : knowledge graph network 의 item embeddings 을 user가 보여주는 dynamic preferences 의 representations 에 통합하는 bidirectional transformer module

3.2 Knowledge Graph Module with Attention

- items간 items' metadata를 unified graph 로 encoding 하여 item간의 higher-order connectivity 를 활용
- graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}) where \mathcal{I} \subset \mathcal{E}$, entities를 nodes로, relationships를 edges로 통합
- 예) 영화 데이터 세트의 entities에는 item 및 그 side information(예: genres, producer, actor, etc)가 포함.
- items간의 연결을 캡처하는 building blocks으로 entities를 사용, items으로 시작하고 끝나는 paths에 초점
- items 간 K-order connectivity 는 items (i, j) 간 higher-order relationship을 captures 하는 path

$$i \xrightarrow{r_1} n_1 \xrightarrow{r_2} n_2 \xrightarrow{r_3} \cdots \xrightarrow{r_K} j, \text{ where } (i, j) \in \mathcal{I}, \text{ entity } n_k \in \mathcal{E}, \text{ and relation } r_k \in \mathcal{R} \text{ for } k \in \{1, 2, \cdots, K\}.$$

- path의 item과 entity외에도, graph의 node로 user를 추가하여 collaborative knowledge graph를 사용하여 공통적으로 관련된 item의 joint occurrence을 모델링
- 특히,item-user relations $\mathcal{R}_{\mathcal{U}}$ 를 knowledge graph에 통합하므로, resulting graph에서는 ,item-user relations도 포착. 한 쌍의 node와 그들의 relations를

$$\mathcal{G} = \{(h, r, t) | h, t \in \mathcal{E}', r \in \mathcal{R}'\}, \text{ where } \mathcal{E}' \subset \mathcal{E} \cup \mathcal{U} \text{ and } \mathcal{R}' \subset \mathcal{R} \cup \mathcal{R}_{\mathcal{U}}.$$

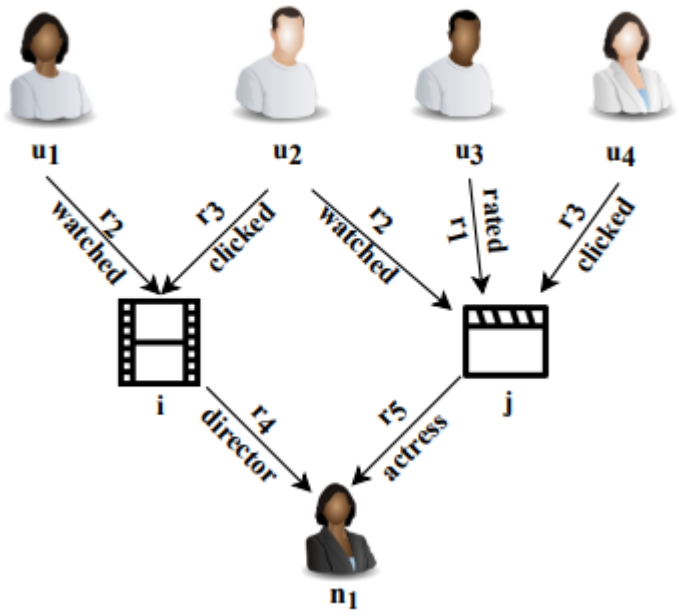


Fig3. Collaborative knowledge graph relates **items** i and j through **user** $u2$ and **entity n1** with different types of relations.

- 그림 3은 (undirected) path에서 entity n1과 관련이 있거나

$$i \xrightarrow{r_4} n_1 \xrightarrow{r_5} j$$

- path에서 user2 u_2와 관련된 second order neighbors인 movie i 와 movie j의 collaborative knowledge graph를 보여줌.

$$i \xrightarrow{r_3} u_2 \xrightarrow{r_2} j.$$

- relations을 item embeddings으로 인코딩하기 위해 TransR 방법을 사용[18]. 만약 triplet (h, r, t) 가 knowledge graph 에 존재할 경우

$$W^r e_h + e_r \approx W^r e_t \ (e_h, e_t \in \mathbb{R}^d, e_r \in \mathbb{R}^k, \text{and } W^r \in \mathbb{R}^{k \times d}),$$

- 각 triplet (h, r, t) 에 대해 **dis**similarity score 계산**(triplet loss 사용을 위해...! : dissimilarity dist 계산해야 되서...?)**

$$s(h, r, t) = \|W^r e_h + e_r - W^r e_t\|_2^2,$$

- $s(h,r,t)$ 의 score가 낮을수록, KG의 triplet이 될 가능성이 높다. [29]에 이어 node와 higher-order neighbors 사이에 attentive weights를 생성하여 각 relation's importance를 포착.
- 따라서, 각 node에 대해, initially 와 first-order relation가 있는 모든 node를 고려.

$$\mathcal{N}_h = \{(h', r, t) | (h', r, t) \in \mathcal{G} \text{ with } h' = h\}.$$

- head node h의 first order connectivity embedding은 ego network embedding의 linear combination으로 정의.

$$e_{\mathcal{N}_h} = \sum_{(h,r,t) \in \mathcal{N}_h} \pi(h, r, t) e_t,$$

- attention factor $\pi(h,r,t)$는 specific relations를 기반으로 different tails의 정보가 head h로 얼마나 propagated 될 수 있는지를 controls하며 다음과 같이 계산

$$\pi(h, r, t) = (W^r e_t)^\top \tanh(W^r e_h + e_r).$$

- 이 attention mechanism은 relationship space에서 더 가까운 entities로부터 더 많은 정보를 propagate 할 것. softmax function을 사용하여 모든 h의 first-order relations에 걸쳐 coefficients를 normalize.

$$\pi'(h, r, t) = \frac{\exp(\pi(h, r, t))}{\sum_{(h,r',t') \in \mathcal{N}_h} \exp(\pi(h, r', t'))}.$$

- 다음 관계를 사용하여 node의 representation과 ego network/connectivity representation을 집계하여 node의 representation을 업데이트 : $e_h^{(1)} = f_{\mathcal{G}}^{(1)}(e_h, e_{\mathcal{N}_h})$, aggregator : $f_{\mathcal{G}}^{(1)}$ 정의

$$f_{\mathcal{G}}^{(1)} = \sigma\left(W_1^{(1)}(e_h + e_{\mathcal{N}_h})\right) + \sigma\left(W_2^{(1)}(e_h \odot e_{\mathcal{N}_h})\right).$$

- σ : **Leaky ReLU** activation function, $W_1^{(1)}, W_2^{(1)} \in \mathbb{R}^{d(1)} \times \mathbb{R}^d$: trainable weight matrices. 두representations e_h와 $e_{\mathcal{N}_h}$의 합을 통해 information를 aggregating하여 **residual connections**에 대해 similar intuition을 따르고, identity transformation을 사용하여 copy of input 을 유지.
- $\odot$: e_h와 $e_{\mathcal{N}_h}$간의 feature interaction capture를 위한 element-wise product. similar nodes로부터 information 의 propagation 은 richer 됨을 ensures
- higher-order propagation을 위해, higher-order neighbors로부터 information를 cascade하기 위해 더 많은 propagation layers를 쌓음. I번째 단계 node representation 공식

$$e_h^{(l)} = f_{\mathcal{G}}^{(l)}(e_h^{(l-1)}, e_{\mathcal{N}_h}^{(l-1)}),$$

- $l-1$ th ego network에서 cascaded되는 information의 정의

$$e_{\mathcal{N}_h}^{(l-1)} = \sum_{(h,r,t)\in\mathcal{N}_h} \pi'(h,r,t) e_t^{(l-1)}.$$

- 이 embedding propagation mechanism을 사용하여 L 레이어를 쌓으면, node representation에서 higher-order connectivities을 포착 가능.
- node의 final representation을 얻기 위해 이러한 representations을 하나의 vector로 concatenate.

$$e_h^* = [e_h^{(0)} \,\|\, \cdots \,\|\, e_h^{(L)}] \in \mathbb{R}^q,$$

- $e_h^{(0)} := e_h$, $q = d + d^{(1)} + \cdots + d^{(L)}$
- 위에서 설명한 knowledge graph attention module의 different layers은 그림 2(왼쪽)에 나와 있습니다.

3.3 Dynamic User Interaction Module

- user가 상호 작용한 successive items 사이의 sequential patterns을 capture하기 위해, BERT(Transformers) 아키텍처[2, 26]를 사용. context에서 BERT는 사용자 u에 해당하는 historical item sequence S_u를 사용하고 user u가 다음 time 단계 $T+1$

에 관심 있는 item을 예측하는 것을 목표로 함. **BERT**는 M bidirectional transformer layers를 모델링하고 each layer에서 각 item's representation을 수정.

- previous layer의 all positions에서 information를 교환. 주요 기여는 3.2절에서 논의된 item relations의 higher-order connectivity을 BERT 모듈에 embed하여 보다 informative context에서 user preferences를 포착하는 것.

3.3.1 Embedding Layer

Positional Embedding: self-attention mechanism은 recurrent 또는 convolutional blocks을 포함하지 않기 때문에 item의 Positional Embedding을 인식할 수 없음. 따라서, pre-determined positional embedding $P \in \mathbb{R}^{T \times q}$을 input embedding에 통합.

$$v_i^{(0)} = e_i^* + p_i,$$

- $v_i^{(0)}$ 은 each position index i = 1, $\cdots$,T .에서 items 의 input representation을 computes.
- user sequence에서T items의 Concatenating embedding, $v_i^{(0)} \in \mathbb{R}^q$, 결과는 $V^{(0)} \in \mathbb{R}^{T \times q}$
- 이 positional embedding matrix 가 first transformer layer의 input 으로 들어감.

3.3.2 Transformer Layer.

- transformer layer에는 두 개의 sublayers, multi-head self-attention sublayer 및 position-wise feed-forward network가 포함.

1) **Multi-Head Self-Attention**

- attention mechanism은 모델이 input sequence의 any distance에서도 여러 subspace representations에 걸쳐 각 item pair 간의 dependencies를 동시에 캡처하는 데 도움.
- 다양한 representation subspaces에서 information을 학습하기 위해 multi-head attention 사용[2].
- 먼저 $V^{(0)}$를 different learnable projections을 사용하여 k subspaces에 linearly project한 다음 각각에 attention function를 parallel하게 적용하여 다음과 같이 k개 heads를 $H_1, \cdots, H_k$ 를 만듬.

$$H_j = \text{Attention}(V^{(0)}W_j^{\mathcal{Q}}, V^{(0)}W_j^{\mathcal{K}}, V^{(0)}W_j^{\mathcal{V}}),$$

$W_j^{\mathcal{Q}}, W_j^{\mathcal{K}}, W_j^{\mathcal{V}}$ 는 모두 head index에 해당하는 $j = 1, \cdots, k$, $\mathbb{R}^{q \times q/k}$ learnable projection matrices

attention function은 Attention($\mathcal{Q}, \mathcal{K}, \mathcal{V}$) = $softmax(\frac{\mathcal{Q}\mathcal{K}^T}{\sqrt{q/k}})\mathcal{V}$로 정의된 scaled dot-product. multi-head attention output을 계산하기 위해 이러한 k head를 연결한 다음 다음과 같이 투영.

$$\text{MH}(V^{(0)}) = [H_1 \parallel H_2 \parallel \cdots \parallel H_k]W^O,$$

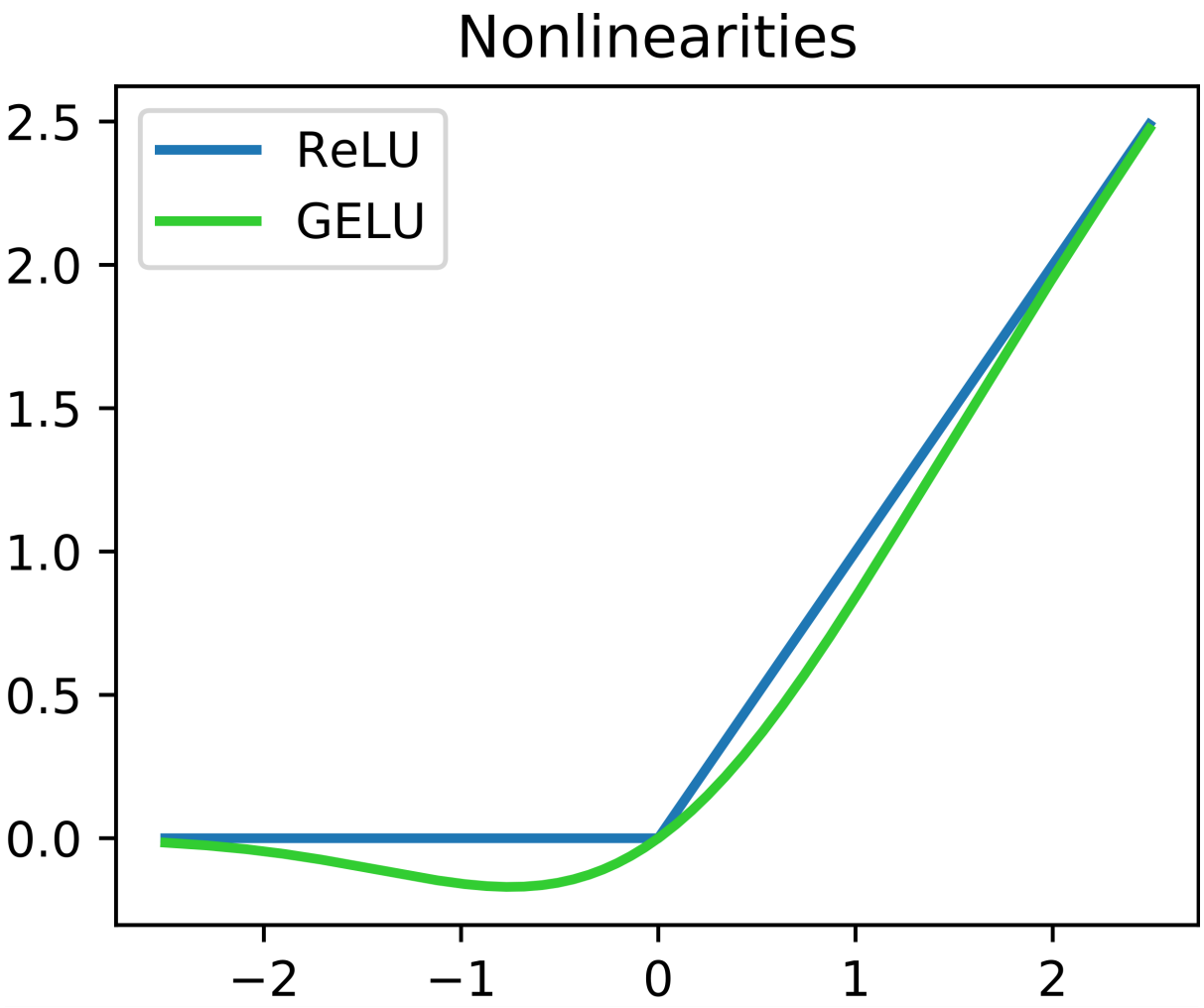
learnable projection matrix $W^O \in \mathbb{R}^{p \times q}$. 아래에 설명된 feed-forward sublayer에 input MH($V^{(0)}$) 입력합니다.

2) **Position-wise Feed-Forward Network**

- self-attention sub-layer는 주로 linear transformations을 반복하기 때문에 non-linearity을 구현하고 different dimensions 간의 interactions도 포착할 수 있는 또 다른 sub-layer가 필요. 각 position $i = 1, \cdots, T$에서 multi-head self-attention sublayer의 출력에 position-wise feed-forward network를 사용.
- **(concat의 의미?)**

$$\text{PFF}(V^{(0)}) = [FF(v_1^{(0)})^\top \parallel \cdots \parallel FF(v_T^{(0)})^\top]$$

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



- GELU non-linearity, $W^1, W^{2^T} \in \mathbb{R}^{q \times 4q}, b^1 \in \mathbb{R}^{4q}, b^2 \in \mathbb{R}^q$는 different positions에서 learnable 하고 **shared**.
- 또한 multi-head self-attention 및 position-wise feed-forward sub-layers의 output에 **dropout**을 적용. architecture depth 때문에 item-item interactions을 더 잘 포착하기 위해 residual connections도 추가.
- 훈련을 stabilize하고 accelerate하기위해 **layer normalization**(LN)를 적용. 이러한 연산은 $LN(x + Dropout(sublayer(x)))$ 으로 요약가능.
- sequence의 item을 보다 복잡하게 표현하기 위해 그림 2에 설명된 것처럼 M **Transformer layers**를 쌓음. 다시 한 번, 위의 선택들이 언어 모델링을 위한 원래의 BERT 아키텍처에 의해 결정.

3.3.3 Output Layer.

1) Item Co-occurrence Modeling

- pairwise item relations를 포착하는 것은 recommendation systems의 effectiveness에 중요한 역할을 하며 어느 정도의 **해석 가능성**도 허용. item embeddings에 item-item relations를 포함하기 위해 sequential module과 knowledge graph에 의해 학습된 item embeddings을 concatenate.

$$\hat{E} = \sigma((V^* \parallel E^*)\hat{W} + \hat{b}),$$

- $\hat{E} \in \mathbb{R}^{|\mathcal{I}| \times q}$: the set of final KATRec embeddings item set $\mathcal{I}$와 $\hat{W} \in \mathbb{R}^{2q \times q}$, $\hat{b} \in \mathbb{R}^{q}$: learnable parameters. V^* : sequential module에 의해 학습되는 items의 embedding matrix. E^* : knowledge graph에 의해 학습되는 items embedding table.
- (Time T+1) next item 예측

$$P(\mathcal{I}) = \text{softmax}(\text{GELU}(v_{T+1}^{(M)} W^P + b^P)\hat{E}^\top + b^O),$$

- W^P, b^P, b^O : learnable parameters, $\hat{E} \in \mathbb{R}^{|\mathcal{I}| \times q}$: item set $\mathcal{I}$ 에 대한 embedding matrix, $P(\cdot)$: KATRec model's **predicted distribution** over the target items. $v_{T+1}^{(M)}$: M transformer layers 이후의 position $T+1$ 의 hidden state. v_{T+1}^*

2) Explicit User Modeling

기존 접근방식
(1) users' previous actions을 기반으로 사용자를 explicitly 하게[16, 22, 25] 포함

(2) 사용자가 visited한 items sequence 의 embeddings을 기반으로 implicitly[8, 14, 24] 모델링하여 personalized recommendation을 제공.

hidden state embedding $v_{T+1}^{(M)}$ 을 고려하여 timestep T +1 단계에서 다음 item을 예측하기 때문에 KATRec은 후자의 범주에 속함.

- 이와는 별도로 knowledge graph에서 학습한 user embeddings을 concatenation을 통해 user's hidden state에 통합함으로써 explicituser behavior을 고려 :

$$[e_u^* \parallel v_{T+1}^{(M)}],$$

- e_u^* : knowledge graph의 user u에 대한 embedding
- 이 concatenation이 promising해보이지만, empirically model's performance 향상을 관찰하지 못함. model이 interacted items의 sequence를 고려하여 users' embedding을 매우 잘 학습했기 때문일 수 있음.

3.4 Optimization

- KATRec의 loss function는 regularizer와 함께 TransR 목표를 포함. 특히, TransR을 사용하여 entity embeddings을 훈련. objective function는 collaborative knowledge graph에서 valid triplets과 invalid triplets 사이를 구별함으로써 최소화할 수 있음.

$$\mathcal{L}_{\mathcal{G}} = \sum_{(h,r,t,t')} -\ln \sigma\Big(s(h,r,t') - s(h,r,t)\Big) + \lambda \|\Theta\|_2^2, \tag{1}$$

- 여기서 합은 knowledge graph $\mathcal{G}$ 의 모든 valid $(h,r,t) \in \mathcal{G}$ 과 invalid $(h,r,t') \notin \mathcal{G}$triplets, $\sigma(\cdot)$: sigmoid function, $\lambda||\Theta||_2^2$: l2 regularization.
- [2]와 유사하게, 방정식 (1)의 pairwise ranking loss 외에도 Cloze task training approach을 구현. 이를 통해 transformer layers에서 encoders의 parameters를 학습할 수 있음. 이 접근법에서는 input sequence $\mathcal{S}_u$의 item 일부를 무작위로 마스킹하고 예측. 마스크된 각 입력 S'_u의 loss는 다음 식에 의해 주어진다.

$$\mathcal{L}_{\mathcal{S}} = \frac{1}{|\mathcal{S}_u^m|} \sum_{i_m \in \mathcal{S}_u^m} -\log P(i_m = i_m^{\star} | \mathcal{S}'_u),$$

- $\mathcal{S}_u^m$: set of randomly masked items, $i_m^{\star}$: masked item i_m 을 따르는 true item. $P(\cdot)$: target item에 대한 predicted probability mass function.
- These two losses are **jointly minimized** over their respective parameters using **standard first-order approaches.**

4. Experiments

4.1 Dataset

Table 1. Statistics of datasets

Datsets	Users	Items	Interactions	Entities	Relations	Triplets	Density
Amazon-book	70679	24915	846434	88572	39	2557746	0.048%
LastFM	23566	48123	8057269	58266	9	464567	0.7105%
Yelp2018	45919	45538	1185068	90961	42	1853704	0.057%

4.2.5 Ablation Study

Table 2. KATRec versus baselines over three datasets. The improvement percentage compares KATRec versus the next-best alternative.

Datasets	Metrics	GRU	*GRU*$^{++}$	SASRec	BERT	KATRec	Improv.
Amazon	NDCG@1	0.3485	0.3464	0.3749	0.4344	**0.4706**	8.33%
	NDCG@5	0.4404	0.4358	0.5267	0.5715	**0.6110**	6.91%
	NDCG@10	0.4598	0.4574	0.5600	0.6022	**0.6401**	6.2%
	Hit@5	0.5202	0.5148	0.6594	0.6910	**0.7321**	5.94%
	Hit@10	0.58	0.5814	0.7621	0.7856	**0.8217**	4.6%
	MAP	0.42	0.4259	0.5065	0.5539	**0.5907**	6.64%
LastFM	NDCG@1	0.3646	0.3523	0.6771	0.6339	**0.6931**	2.36%
	NDCG@5	0.4648	0.4448	**0.7765**	0.7606	0.7725	-0.51%
	NDCG@10	0.4881	0.4674	**0.7930**	0.7786	0.7911	-0.24%
	Hit@5	0.5531	0.5263	**0.8600**	0.8281	0.8426	-2.06%
	Hit@10	0.6249	0.5958	**0.9105**	0.8836	0.9001	-1.15%
	MAP	0.4577	0.4357	0.7598	0.7509	**0.7618**	0.26%
Yelp2018	NDCG@1	0.3946	0.4148	0.3723	0.4149	**0.4405**	6.17%
	NDCG@5	0.5041	0.5143	0.5703	0.6039	**0.629**	4.15%
	NDCG@10	0.5278	0.5395	0.6068	0.6400	**0.663**	3.6%
	Hit@5	0.5991	0.6021	0.7434	0.7690	**0.7927**	3.08%
	Hit@10	0.6721	0.68	0.8551	0.8796	**0.899**	2.2%
	MAP	0.49	0.515	0.5351	0.5706	**0.5946**	4.21%

- LastFM : **relations** 이 적고, denser한 데이터라 잘 안된다

Table 3. Ablation study of design choices in KATRec using three datasets.

Datasets	Metrics	KATRec	NoAtten.	Level-1	Connect.	NoPretrain	Concat.
Amazon-book	NDCG@10	0.6401	0.6371	0.6386	0.621	0.6306	0.6318
	Hit@10	0.8217	0.8178	0.8195	0.801	0.8092	0.8142
LastFM	NDCG@10	0.7911	0.7836	0.7853	0.763	0.7587	0.7855
	HIT@10	0.9001	0.8967	0.8957	0.8796	0.8752	0.8908
Yelp2018	NDCG@10	0.663	0.6567	0.6546	0.6458	0.6359	0.6515
	HIT@10	0.899	0.8954	0.8929	0.885	0.8696	0.8881

- (1) No Attention: knowledge graph에서 attention mechanism 제거 & each entity's neighbors에 equal weights 할당

- (2) Level-1: neighbors에서 node로 propagate할 수 있는 level of information을 줄이고, node embeddings을 개선하기 위해 immediate neighbors만 사용할 때의 영향을 연구
- (3) Connection: 모델의 two encoder modules 사이에 connection이 존재하지만, 두 모듈이 학습 가능한 임베딩을 사용하여 독립적으로 학습하는 설정을 고려**(embedding freeze)**
- (4) No Pretraining: knowledge graph에 pre-trained embeddings entities를 포함시키는 것을 포기
- (5) Concat: item co-occurrence을 다루는 모델의 일부를 제거하고 sequential module에서 나오는 item embedding vector만 고려

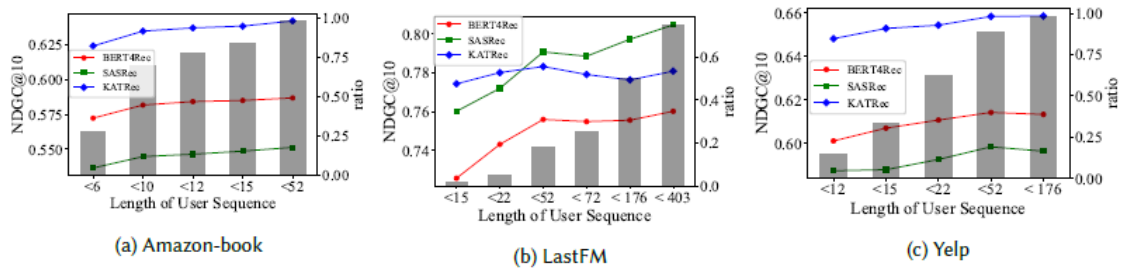


Fig. 4. Performance comparison of models across users with different sequence lengths on the Amazon-book, LastFM and Yelp2018 datasets.

4.2.7 Attention Weight Case Study.

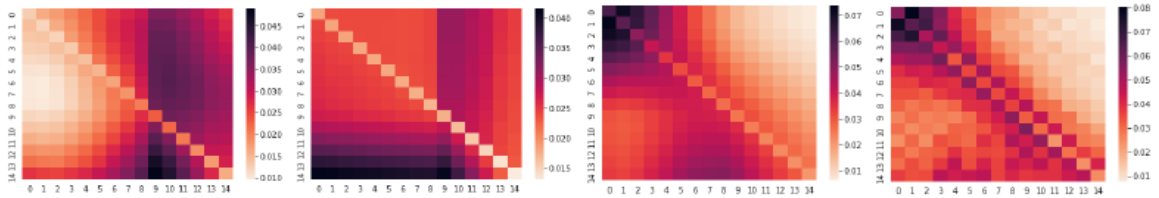
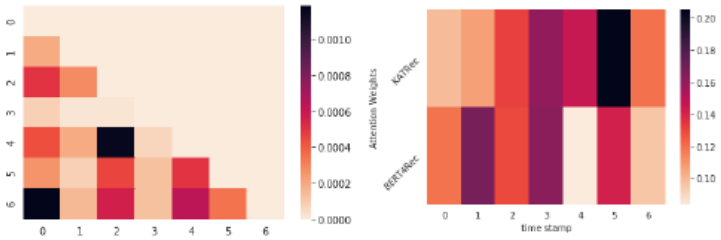


Fig. 5. Average attention weights on positions (x-axis) at different time (y-axis) on Yelp



(a) Co-occurence information (b) Attention weights

Fig. 6. Attention heatmap comparison of a random user in Yelp2018 dataset. 6a shows the co-occurrence ratio, which is the co-occurrence frequency of each pair of six items among all users' sequences. 6b compares these items' attention weight at the last position in KATRec and BERT4Rec.

- user의 item seq의 순서간 시간차에 따라 문제 발생?