ADKGN: An Attentive Dynamic Knowledge Graph Network for Sequential Recommendation

Mengqiu Yao Liqiang Song Ye Bi
Wei Wang Kun Deng Jianming Wang Jing Xiao
Ping An Technology Shenzhen Co., Ltd
Xiaoyun Lin Zhaojun Gui
Ping An Finserve Shenzhen Co., Ltd

Speaker: **Ye Bi**

About the Speaker

- Senior Algorithm Engineer in Ping An Technology Shenzhen Co., Ltd
- **Major Research Direction:** Deep Learning, Recommendation System, Knowledge Graph

Information overload brings difficulties to users to find appropriate items

Recommendation system (RS) overcomes the challenge and suggests a small set of items to meet users' personalized interests.As we know, sequential recommendation system's goal is to predict users'next actions based on their historical behavior sequences. More recent items have a larger impact than the previous ones. Meanwhile, modeling users' current interests are challenging. Knowledge graph (KG) contains a vast of information, which can help us capture users'interests by propagating their interactions.

Contributions

Our contributions in this paper are as follows:

- Design a novel recommendation approach, which aims to embed users' current interactions into a KG and incorporate entities and relations from KG to get user representations.
- Propose a dynamic knowledge graph embedding method to combine the strengths of different kinds of KGE methods by dynamically learning a weight for each pretrained KGE method.
- Design parallelbased aggregation layer to effectively aggregate useful information from multi-hop neighbors.
- Conduct experiments on public dataset, the results prove the efficacy of ADKGN over several baselines.

Framework

In our paper, we propose an Attentive Dynamic Knowledge Graph Networks (ADKGN) for sequential recommendation, which includes an embedding module, an attentive dynamic knowledge graph module, a sequential process module and a prediction module.

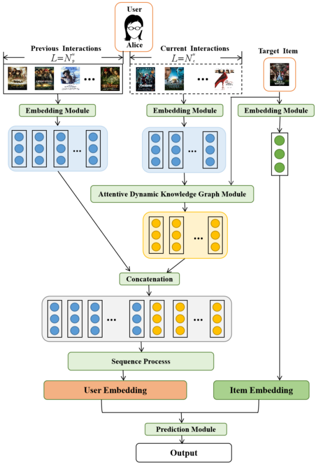


Figure 1: The Framework of ADKGN

Embedding Module

The first stage of our method is item embedding. For the item in users' historical interaction sequences H, we combine the item's latent feature and its sequential attributes.Specifically, we employ word2vec over users' historical interaction sequences to capture item i's sequential attribute s_i. And then, we represent item embedding by combining the item' s latent feature and its sequential attribute:

$$\theta_i = P \cdot concat(i, s_i)$$

where$P \in \mathbb{R}^{d \times 2d}$is a projection matrix, i is latent features vector, which is initialized randomly.

Attentive Dynamic Knowledge Graph Module

Three main layers to model the users’current interests:

- dynamic knowledge graph embedding layer,
- top-k layer
- parallel-based aggregation layer.

Dynamic Knowledge Graph Embedding Layer

We pretrain KG with different KGE methods,the inputs of all methods are the same randomly generated vectors. And then we predict a weight for each KGE method,optionally depending on the RS task. For DKG method, we combine the KGE methods by taking the weighted sum, take the tail as an example

$$e_t^D = \sum_{j=1}^{n} \beta_j e_t^j$$

where $beta_j$ are scalar weights from self-attention mechanism$\beta_j = \phi(a \cdot e_t^j + b)$, $a \in \mathbb{R}^d$ and $b \in \mathbb{R}$ are learned parameters and ϕ is a softmax.

Top-k Layer

Each item $i \in \mathcal{I}^u$ is treated as a seed in KG, then extended along links. As defined above, $\mathcal{N}_l(i)$ is the set of entities l hops away from seed i. KG usually contains fruitful connections, even in $\mathcal{N}_1(i)$, there may be thousands of entities. Some entities bring more noises than useful signals,so we rank entities in $\mathcal{N}_l(i)$ according to rating information,and then choose top k entities, which will form a new entity set $top_k(\mathcal{N}_l(i))$.

parallel-based aggregation layer

Given the target item v's DKG vector e_v^D and top k l-hop neighbors $top_k(\mathcal{N}_l(i))$, each entity e_j in $top_k(\mathcal{N}_l(i))$ is assigned an attention score by comparing it to the target item v:

$$\alpha_j^i = \frac{exp(\langle e_v^D, e_j^D \rangle)}{\sum_{e \in top_k(\mathcal{N}_l(i))} exp(\langle e_v^D, e^D \rangle)}$$

Then we take the weighted sum of entities in $top_k(\mathcal{N}_l(i))$, and the vector o_i^l is returned:

$$o_i^l = \sum_{e \in top_k(\mathcal{N}_l(i))} \alpha_j^i e^D$$

Sequential Process Module

We first concat a user history and processed current interactions

$$O = \left[\theta_{i_1}, ..., \theta_{i_{N_p^u}}, \omega_{i_{N_p^u+1}}, ..., \omega_{i_{|\mathcal{I}_r^u}}\right]$$

In this paper, we employ a GRU and self-attention networks to deal with O, and take the hidden layer of these models as the user representation, denoted by u.

Prediction Module

User representation u and target item representation v^D are combined to predict the clicking probability: $\hat{y}_u v = \sigma(f(u, v_D))$, where $\sigma(\,\cdot\,)$ is the sigmoid function, and f is a ranking function. The loss function L is:

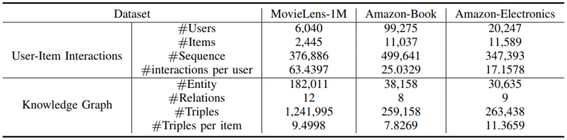
$$L = \sum_{(u,v)\in Y} -(y_{uv} log \hat{y}_{uv} + (1 - y_{uv}) log(1 - \hat{y}_{uv}))$$

Experiment

We evaluate our proposed ADKGN on recommendation tasks using two public datasets (MovieLens-1M and Amazon Dataset) to answer the following questions:
(i) Do ADKGN with several variants outperform the state-of-the-art baselines?
(ii) How ADKGN is affected by each component?

Datasets and Preprocessing

We randomly split the user-item interactions of each dataset into training set (80%) to learn parameters,validation set (10%) to tune hyper-parameters, and testing set (10%) for performance comparison. Table II shows basic statistics of the datasets.



Dataset		MovieLens-1M	Amazon-Book	Amazon-Electronics
User-Item Interactions	#Users	6,040	99,275	20,247
	#Items	2,445	11,037	11,589
	#Sequence	376,886	499,641	347,393
	#interactions per user	63.4397	25.0329	17.1578
Knowledge Graph	#Entity	182,011	38,158	30,635
	#Relations	12	8	9
	#Triples	1,241,995	259,158	263,438
	#Triples per item	9.4998	7.8269	11.3659

Figure 2: STATISTICS OF THE PREPROCESSED DATASETS

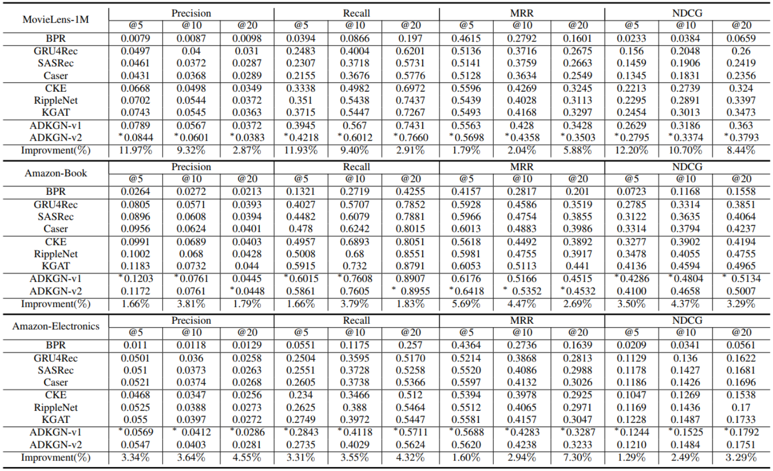
Baseline and Metrics

Baseline:

- The first group includes classical recommendation methods only considering user disorder feedback, BPR.
- The second group contains state-of-the-art sequential recommendation methods considering the sequence of user actions, GRU4Rec, SASRec and Caser.
- The third group contains state-of-the-art KG-based recommendation methods considering KG information, CKE, RippleNet and KGAT.

Metrics: Recall@N and NDCG@N (N=5, 10, 20).

Overall Performance



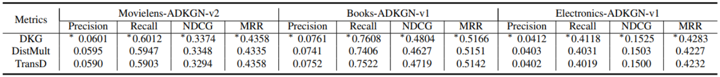
MovieLens-1M	Precision			Recall			MRR			NDCG		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
BPR	0.0079	0.0087	0.0098	0.0394	0.0866	0.197	0.4615	0.2792	0.1601	0.0233	0.0384	0.0659
GRU4Rec	0.0497	0.04	0.031	0.2483	0.4004	0.6201	0.5136	0.3716	0.2675	0.156	0.2048	0.26
SASRec	0.0461	0.0372	0.0287	0.2307	0.3718	0.5731	0.5141	0.3759	0.2663	0.1459	0.1906	0.2419
Caser	0.0431	0.0368	0.0289	0.2155	0.3676	0.5776	0.5128	0.3634	0.2549	0.1345	0.1831	0.2356
CKE	0.0668	0.0498	0.0349	0.3338	0.4982	0.6972	0.5596	0.4269	0.3245	0.2213	0.2739	0.324
RippleNet	0.0702	0.0544	0.0372	0.351	0.5438	0.7437	0.5439	0.4028	0.3113	0.2295	0.2891	0.3397
KGAT	0.0743	0.0545	0.0363	0.3715	0.5447	0.7267	0.5493	0.4168	0.3297	0.2454	0.3013	0.3473
ADKGN-v1	0.0789	0.0567	0.0372	0.3945	0.567	0.7431	0.5563	0.428	0.3428	0.2629	0.3186	0.363
ADKGN-v2	*0.0844	*0.0601	*0.0383	*0.4218	*0.6012	*0.7660	*0.5698	*0.4358	*0.3503	*0.2795	*0.3374	*0.3793
Improvment(%)	11.97%	9.32%	2.87%	11.93%	9.40%	2.91%	1.79%	2.04%	5.88%	12.20%	10.70%	8.44%

Amazon-Book	Precision			Recall			MRR			NDCG		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
BPR	0.0264	0.0272	0.0213	0.1321	0.2719	0.4255	0.4157	0.2817	0.201	0.0723	0.1168	0.1558
GRU4Rec	0.0805	0.0571	0.0393	0.4027	0.5707	0.7852	0.5928	0.4586	0.3519	0.2785	0.3314	0.3851
SASRec	0.0896	0.0608	0.0394	0.4482	0.6079	0.7881	0.5966	0.4754	0.3855	0.3122	0.3635	0.4064
Caser	0.0956	0.0624	0.0401	0.478	0.6242	0.8015	0.6013	0.4883	0.3986	0.3314	0.3794	0.4237
CKE	0.0991	0.0689	0.0403	0.4957	0.6893	0.8051	0.5618	0.4492	0.3892	0.3277	0.3902	0.4194
RippleNet	0.1002	0.068	0.0428	0.5008	0.68	0.8551	0.5981	0.4755	0.3917	0.3478	0.4055	0.4755
KGAT	0.1183	0.0732	0.044	0.5915	0.732	0.8791	0.6053	0.5113	0.441	0.4136	0.4594	0.4965
ADKGN-v1	*0.1203	*0.0761	0.0445	*0.6015	*0.7608	0.8907	0.6176	0.5166	0.4515	*0.4286	*0.4804	* 0.5134
ADKGN-v2	0.1172	0.0761	*0.0448	0.5861	0.7605	* 0.8955	*0.6418	* 0.5352	*0.4532	0.4100	0.4658	0.5007
Improvment(%)	1.66%	3.81%	1.79%	1.66%	3.79%	1.83%	5.69%	4.47%	2.69%	3.50%	4.37%	3.29%

Amazon-Electronics	Precision			Recall			MRR			NDCG		
	@5	@10	@20	@5	@10	@20	@5	@10	@20	@5	@10	@20
BPR	0.011	0.0118	0.0129	0.0551	0.1175	0.257	0.4364	0.2736	0.1639	0.0209	0.0341	0.0561
GRU4Rec	0.0501	0.036	0.0258	0.2504	0.3595	0.5170	0.5214	0.3868	0.2813	0.1129	0.136	0.1622
SASRec	0.051	0.0373	0.0263	0.2551	0.3728	0.5258	0.5520	0.4086	0.2988	0.1178	0.1427	0.1681
Caser	0.0521	0.0374	0.0268	0.2605	0.3738	0.5366	0.5597	0.4132	0.3026	0.1186	0.1426	0.1696
CKE	0.0468	0.0347	0.0256	0.234	0.3466	0.512	0.5394	0.3978	0.2925	0.1047	0.1269	0.1538
RippleNet	0.0525	0.0388	0.0273	0.2625	0.388	0.5464	0.5512	0.4065	0.2971	0.1169	0.1436	0.17
KGAT	0.055	0.0397	0.0272	0.2749	0.3972	0.5447	0.5581	0.4157	0.3047	0.1228	0.1487	0.1733
ADKGN-v1	*0.0569	* 0.0412	*0.0286	*0.2843	*0.4118	*0.5711	*0.5688	*0.4283	*0.3287	*0.1244	*0.1525	*0.1792
ADKGN-v2	0.0547	0.0403	0.0281	0.2735	0.4029	0.5624	0.5620	0.4238	0.3233	0.1210	0.1484	0.1751
Improvment(%)	3.34%	3.64%	4.55%	3.31%	3.55%	4.32%	1.60%	2.94%	7.30%	1.29%	2.49%	3.29%

Figure 3: RECOMMENDATION PERFORMANCE

Model Analysis



Metrics	Movielens-ADKGN-v2				Books-ADKGN-v1				Electronics-ADKGN-v1			
	Precision	Recall	NDCG	MRR	Precision	Recall	NDCG	MRR	Precision	Recall	NDCG	MRR
DKG	* 0.0601	*0.6012	*0.3374	*0.4358	* 0.0761	*0.7608	*0.4804	*0.5166	* 0.0412	*0.4118	*0.1525	*0.4283
DistMult	0.0595	0.5947	0.3348	0.4335	0.0741	0.7406	0.4627	0.5151	0.0403	0.4031	0.1503	0.4227
TransD	0.0590	0.5903	0.3294	0.4358	0.0752	0.7522	0.4719	0.5142	0.0402	0.4019	0.1500	0.4232

Figure 4: THE RESULTS OF METRICS @10 W.R.T. DIFFERENT KG EMBEDDING OPTION

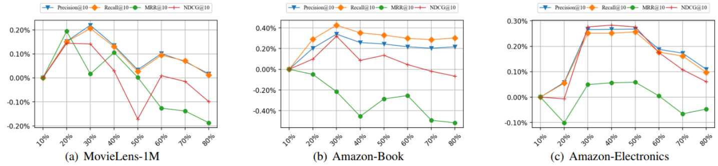
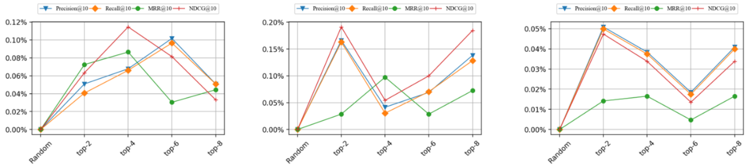


Figure 5: The results of metrics @10 w.r.t. different current interactions ratio

Model Analysis

Metrics	Movielens-ADKGN-v2				Books-ADKGN-v1				Electronics-ADKGN-v1			
	Precision	Recall	NDCG	MRR	Precision	Recall	NDCG	MRR	Precision	Recall	NDCG	MRR
DKG	* 0.0601	*0.6012	*0.3374	*0.4358	* 0.0761	*0.7608	*0.4804	*0.5166	* 0.0412	*0.4118	*0.1525	*0.4283
DistMult	0.0595	0.5947	0.3348	0.4335	0.0741	0.7406	0.4627	0.5151	0.0403	0.4031	0.1503	0.4227
TransD	0.0590	0.5903	0.3294	0.4358	0.0752	0.7522	0.4719	0.5142	0.0402	0.4019	0.1500	0.4232

Figure 6: THE RESULTS OF METRICS @10 W.R.T. DIFFERENT KG EMBEDDING OPTION



(a) MovieLens-1M (b) Amazon-Book (c) Amazon-Electronics

Figure 7: The results of metrics @10 w.r.t. different top-k sampling number

Conclusion

- In this paper, we propose a new structure of deep network, namely Attentive Dynamic Knowledge Graph Network(ADKGN) for sequential recommendation, which aims to embed users'recent items into a KG and incorporate entities and relations from KG to help get user representations.
- The experiment results on public datasets demonstrate the efficiency of our ADKGN.

Q&A

Thank you for your attention!

If you have any questions, please email to magicyebi@163.com