Jointly Non-Sampling Learning for Knowledge Graph Enhanced Recommendation

Tuo Liu

August 10, 2022

1 Main Contribution

1. We highlight the importance of building KG enhanced recommendation models without negative sampling, and derive an efficient optimization method to learn from the whole knowledge graph with a controllable time complexity.
2. We propose a novel end-to-end model JNSKR, which creatively addresses the KG enhanced recommendation task from the basic but important perspective of model learning. To the best of our knowledge, this is the first non-sampling learning method for KG enhanced recommendation.
3. Extensive experiments on two public benchmarks show that JNSKR consistently and significantly outperforms the state-of-the-art models in terms of both recommendation performance and training efficiency. The source code of JNSKR and datasets used in the paper have been made available1.

2 Methodology

2.1 Efficient Non-sampling Knowledge Graph Embedding

The paper applies non-sampling strategy for knowledge graph embedding learning. The process of the non-sampling is as follows. for a batch of entities B, the squared loss in graph embedding learning is defined as

$$\begin{aligned}\mathcal{L}_{KG}(\Theta) &= \sum_{h\in\mathbf{B}}\sum_{t\in\mathbf{E}}\sum_{r\in\mathbf{R}} w_{hrt}\left(g_{hrt}-\hat{g}_{hrt}\right)^2 \\ &= \sum_{h\in\mathbf{B}}\sum_{t\in\mathbf{E}}\sum_{r\in\mathbf{R}} w_{hrt}\left(g_{hrt}^2-2g_{hrt}\hat{g}_{hrt}+\hat{g}_{hrt}^2\right)\end{aligned}$$

where w_{hrt} denotes the weight of entry g_{hrt} , $g_{hrt}=1$ if there is a relation r between h and t, and $g_{hrt}=0$ otherwise. Since $g_{hrt}\in\{0,1\}$, it can be replaced by a constant to simplify the equation. Also, the loss of non-observed data can be expressed by the residual between the loss of all data and that of positive data. The paper has the following derivation:

$$\begin{aligned}\tilde{\mathcal{L}}_{KG}(\Theta) = &-2\sum_{h\in\mathbf{B}}\sum_{t\in\mathbf{E}^+}\sum_{r\in\mathbf{R}^+} w_{hrt}^+\hat{g}_{hrt} + \sum_{h\in\mathbf{B}}\sum_{t\in\mathbf{E}}\sum_{r\in\mathbf{R}} w_{hrt}\hat{g}_{hrt}^2 \\ &\overbrace{\sum_{h\in\mathbf{B}}\sum_{t\in\mathbf{E}^+}\sum_{r\in\mathbf{R}^+}\left(\left(w_{hrt}^+-w_{hrt}^-\right)\hat{g}_{hrt}^2-2w_{hrt}^+\hat{g}_{hrt}\right)}^{\mathcal{L}_{KG}^P(\Theta)} \\ &\overbrace{\sum_{h\in\mathbf{B}}\sum_{t\in\mathbf{E}}\sum_{r\in\mathbf{R}} w_{hrt}^-\hat{g}_{hrt}^2}^{\mathcal{L}_{KG}^A(\Theta)}\end{aligned}$$

where the Θ-invariant constant value has been eliminated, $L_{KG}^P(\Theta)$ denotes the loss for positive data, and $L_{KG}^A(\Theta)$ denotes the loss for all data. The computational bottleneck lies in $L_{KG}^A(\Theta)$. As can be

seen from $Eq.(2)$, to address the inefficiency issue of $L^A_{KG}(\Theta)$, $\hat{g}^2_{hrt}$ need to be a score function that can be properly expanded. The paper employs DistMult, which defines the scoring function as:

$$\hat{g}_{hrt} = \mathbf{e}_h^T \cdot \mathrm{diag}(\mathbf{r}) \cdot \mathbf{e}_t = \sum_i^d e_{h,i} r_i e_{t,i}$$

where $diag(\mathbf{r})$ denotes a diagonal matrix whose diagonal elements equal to $\mathbf{r}$ correspondingly.Based on a decouple manipulation for the inner product operation, the summation operator and elements in $\mathbf{e}_h$, $\mathbf{e}_h$ and $\mathbf{r}$ can be rearranged:

$$\begin{aligned}\hat{g}^2_{hrt} &= \sum_i^d e_{h,i} r_i e_{t,i} \sum_j^d e_{h,j} r_j e_{t,j} \\ &= \sum_i^d \sum_j^d \left(e_{h,i} e_{h,j}\right) \left(r_i, r_j\right) \left(e_{t,i} e_{t,j}\right)\end{aligned}$$

By substituting $Eq.(4)$ in $L^A_{KG}(\Theta)$, the paper gets the final efficient non-sampling loss for KG embedding learning:

$$\begin{aligned}&\tilde{\mathcal{L}}_{KG}(\Theta) = \mathcal{L}^P_{KG}(\Theta) \\ &+ \sum_{i=1}^d \sum_{j=1}^d \left(\left(\sum_{r \in \mathbf{R}} r_i r_j \right) \left(\sum_{h \in \mathbf{B}} w_h^- e_{h,i} e_{h,j} \right) \left(\sum_{t \in \mathbf{E}} e_{t,i} e_{t,j} \right) \right)\end{aligned}$$

2.2 User-Item Preference Modeling

The preference prediction framework the paper adopts is the neural form of MF, which is:

$$\hat{y}_{uv} = \mathbf{h}^T \left(\mathbf{p}_u \odot \mathbf{q}_v\right)$$

where $\mathbf{h} \in Rd$ is the prediction vector, $\odot$ denotes the element-wise product of vectors. $\mathbf{p}_u$ and $\mathbf{q}_v$ are the representations of user u and item v, respectively.

The paper uses $Nv = \{(v, r, t) | g_{vrt} = 1\}$ to denote the neighbored knowledge triplets of v. To characterize item v, the paper defines:

$$\begin{aligned}\mathbf{q}_v &= \mathbf{e}_v + \mathbf{e}_{\mathcal{N}_v} \\ &= \mathbf{e}_v + \sum_{(v,r,t) \in \mathcal{N}_v} \alpha_{(r,t)} \mathbf{e}_t\end{aligned}$$

where $\alpha_{(r,t)}$ is the attention weight, indicating how much information being propagated from t to v conditioned to relation r. $\mathbf{e}_v$ is item's basic feature vector and $\mathbf{e}_{N_v}$ represents the information of v's knowledge triplets. More precisely, the paper defines $\alpha_{(r,t)}$ as:

$$\begin{aligned}\alpha^*_{(r,t)} &= \mathbf{h}^T_\alpha \sigma \left(\mathbf{W}_1 \mathbf{e}_t + \mathbf{W}_2 \mathbf{r} + \mathbf{b}\right) \\ \alpha_{(r,t)} &= \frac{\exp\left(\alpha^*_{(r,t)}\right)}{\sum \left(v, r', t'\right) \in \mathcal{N}_v \exp\left(\alpha^*_{(r',t')}\right)}\end{aligned}$$

Note that the prediction part satisfies the requirements, thus for a batch of items $\mathbf{B}$, we have the following efficient non-sampling loss function:

$$\begin{aligned}\tilde{\mathcal{L}}_{CF}(\Theta) = &\sum_{u \in \mathbf{U}^+} \sum_{v \in \mathbf{B}} \left(\left(c_v^+ - c_v^-\right) \hat{y}^2_{uv} - 2 c_v^+ \hat{y}_{uv} \right) \\ &+ \sum_{i=1}^d \sum_{j=1}^d \left(\left(h_i h_j\right) \left(\sum_{u \in \mathbf{U}} p_{u,i} p_{u,j} \right) \left(\sum_{v \in \mathbf{B}} c_v^- q_{v,i} q_{v,j} \right) \right)\end{aligned}$$

2.3 Jointly Multi-task Learning

To effectively learn parameters for recommendation, as well as preserve the well-structured information of knowledge graph, the paper integrates the recommendation part (i.e., $\tilde{\mathcal{L}}_{CF}(\Theta)$ and the knowledge graph embedding part (i.e., $\tilde{\mathcal{L}}_{CF}(\Theta)$ in an end-to-end fashion through a jointly multi-task learning framework:

$$\mathcal{L}(\Theta) = \tilde{\mathcal{L}}_{CF}(\Theta) + \mu\tilde{\mathcal{L}}_{KG}(\Theta) + \lambda\|\Theta\|_2^2$$

where $\tilde{\mathcal{L}}_{CF}(\Theta)$ is the recommendation loss from $Eq.(9)$, $\tilde{\mathcal{L}}_{KG}(\Theta)$ is the KG embedding loss from $Eq.(5)$, and μ is the parameter to adjust the weight proportion of each term. L_2 regularization parameterized by λ on Θ is conducted to prevent overfitting.

To optimize the objective function, the paper uses mini-batch Ada-grad as the optimizer. Dropout is an effective solution to prevent neural networks from overfitting, which randomly drops part of neurons during training. In this work, we employ the node dropout technique to randomly drop ρ percent of $\mathbf{q}_v$, where ρ is the dropout ratio.

3 Experiments

The paper experiments with two benchmark datasets: **Amazon-book**[2] and **Yelp2018**[3] . The baselines include **NCF**, **ENCF**, **NFM**, **CKE**, **CFKG**, **RippleNet** and **KGAT**. The results are as follows:

Models	*Amazon-book*						
	Recall@10	**Recall@20**	**Recall@40**	**NDCG@10**	**NDCG@20**	**NDCG@40**	**RI**
NCF	0.0874	0.1319	0.1924	0.0724	0.0895	0.1111	+17.03%
ENMF	0.1002	0.1472	0.2085	0.0797	0.0998	0.1215	+5.49%
NFM	0.0891	0.1366	0.1975	0.0723	0.0913	0.1152	+14.44%
CKE	0.0875	0.1343	0.1946	0.0705	0.0885	0.1114	+17.14%
CFKG	0.0769	0.1142	0.1901	0.0603	0.077	0.0985	+32.62%
RippleNet	0.0883	0.1336	0.2008	0.0747	0.0910	0.1164	+13.99%
KGAT	0.1017	0.1489	0.2094	0.0814	0.1006	0.1225	+4.31%
JNSKR	**0.1056****	**0.1558****	**0.2178****	**0.0842****	**0.1068****	**0.1271****	–
Models	*Yelp2018*						
	Recall@10	**Recall@20**	**Recall@40**	**NDCG@10**	**NDCG@20**	**NDCG@40**	**RI**
NCF	0.0389	0.0653	0.1060	0.0603	0.0802	0.1087	+14.28%
ENMF	0.0403	0.0711	0.1109	0.0611	0.0877	0.1097	+9.15%
NFM	0.0396	0.0660	0.1082	0.0603	0.0810	0.1094	+13.03%
CKE	0.0399	0.0657	0.1074	0.0608	0.0805	0.1091	+13.13%
CFKG	0.0288	0.0522	0.0904	0.0450	0.0644	0.0897	+44.27%
RippleNet	0.0402	0.0664	0.1088	0.0613	0.0822	0.1097	+11.90%
KGAT	0.0418	0.0712	0.1128	0.0630	0.0867	0.1129	+7.26%
JNSKR	**0.0456****	**0.0749****	**0.1209****	**0.0687****	**0.0917****	**0.1211****	–

Figure 1: Performance

1. JNSKR achieves the best performance on the two datasets, significantly outperforming all the state-of-the-art baseline methods with p-values smaller than 0.01.

2. Non-sampling methods (ENMF and our JNSKR) generally perform better than sampling-based methods.

3. The paper observes that recent studies on KG enhanced recommendation have largely focused on advanced neural network structures. Although they do achieve better performance than conventional CF methods when adopting the same sampling-based learning strategy (e.g., KG baselines vs NCF), they are still limited by the inherent weakness of negative sampling.

4 Future Work

In the future, we will explore JNSKR on other related tasks like knowledge graph representation and network embedding. Also, we are interested in integrating more structural information such as social networks and heterogeneous information networks, to further improve our model.

5 My Idea

This paper proposes a novel non-sampling model JNSKR. Most of the time, we use negative sampling in the study of knowledge graph. However, this paper shows us that non-sampling method performs better than negative sampling and they do find way to reduce the computation. In future study, we can try using their method instead of negative sampling as sampling method, and see if the performance be better.