RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems

Hongwei Wang[1,2], Fuzheng Zhang[3], Jialin Wang[4], Miao Zhao[4], Wenjie Li[4], Xing Xie[2], Minyi Guo[1*]
[1]Shanghai Jiao Tong University, wanghongwei55@gmail.com, guo-my@cs.sjtu.edu.cn
[2]Microsoft Research Asia, xingx@microsoft.com, [3]Meituan AI Lab, zhangfuzheng@meituan.com
[4]The Hong Kong Polytechnic University, {csjlwang, csmiaozhao, cswjli}@comp.polyu.edu.hk

CIKM 2018

PanXiang

2018.12.4

Related Work

Embedding-based methods

Signed Heterogeneous Information Network Embedding

- deep autoencoders to embed sentiment networks, social networks and profile (knowledge) networks

Deep Knowledge-aware Network (DKN)

- treats entity embeddings and word embeddings as different channels
- CNN framework to combine them together for news recommendation

Related Work

path-based methods

Personalized Entity Recommendation (PER)

Meta-Graph Based Recommendation

- heterogeneous information network (HIN)
- extract meta-path/meta-graph based latent features to represent the connectivity between users and items
- natural and intuitive
- hard to optimize

Related Work

path-based methods

Personalized Entity Recommendation (PER)

Meta-Graph Based Recommendation

- heterogeneous information network (HIN)
- extract meta-path/meta-graph based latent features to represent the connectivity between users and items
- natural and intuitive
- hard to optimize

RipNet

- end-to-end:knowledge-graph-aware recommendation
- preference propagation
  - treats his historical interests as a seed set in th KG
  - extends the user’s interests iteratively along KG links
  - discover his hierarchical potential interests with respect to a candidate item
- AUC gains of 2.0% to 40.6%, 2.5%to 17.4%, and 2.6% to 22.4%
in movie, book, and news recommendations
- a new perspective of explainability

Structure

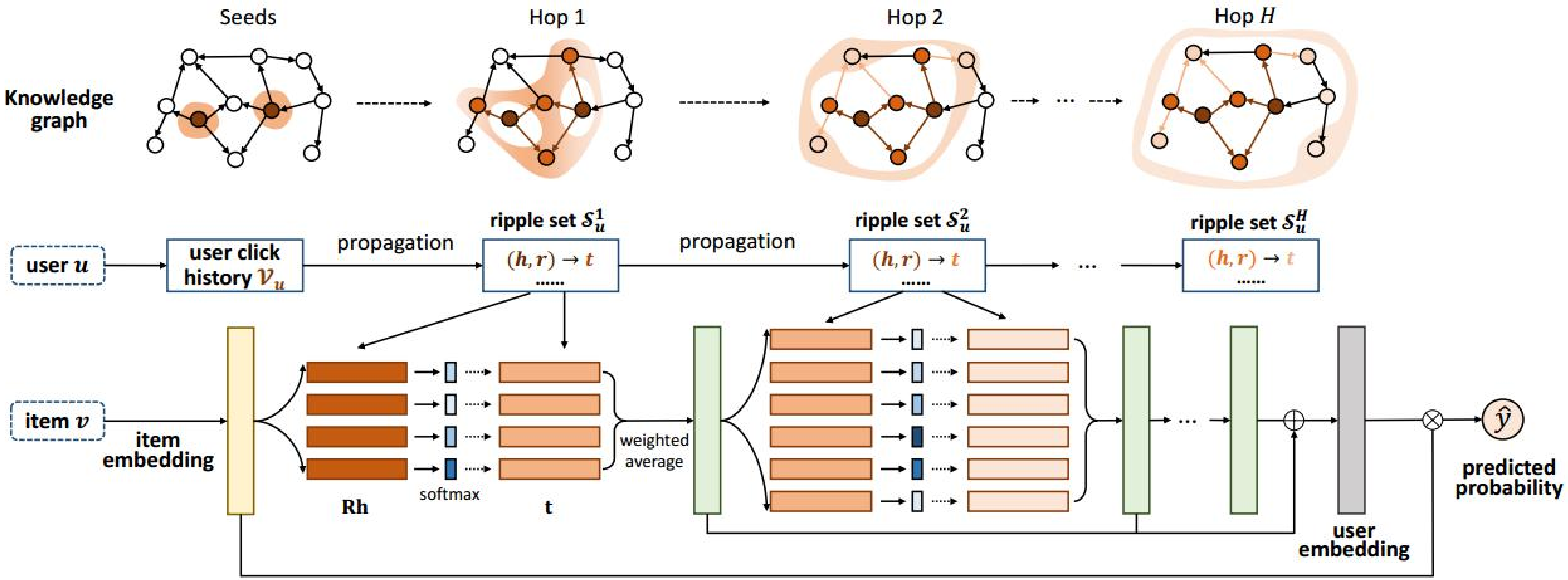

Figure 2: The overall framework of the RippleNet. It takes one user and one item as input, and outputs the predicted probability that the user will click the item. The KGs in the upper part illustrate the corresponding ripple sets activated by the user's click history.

Framework

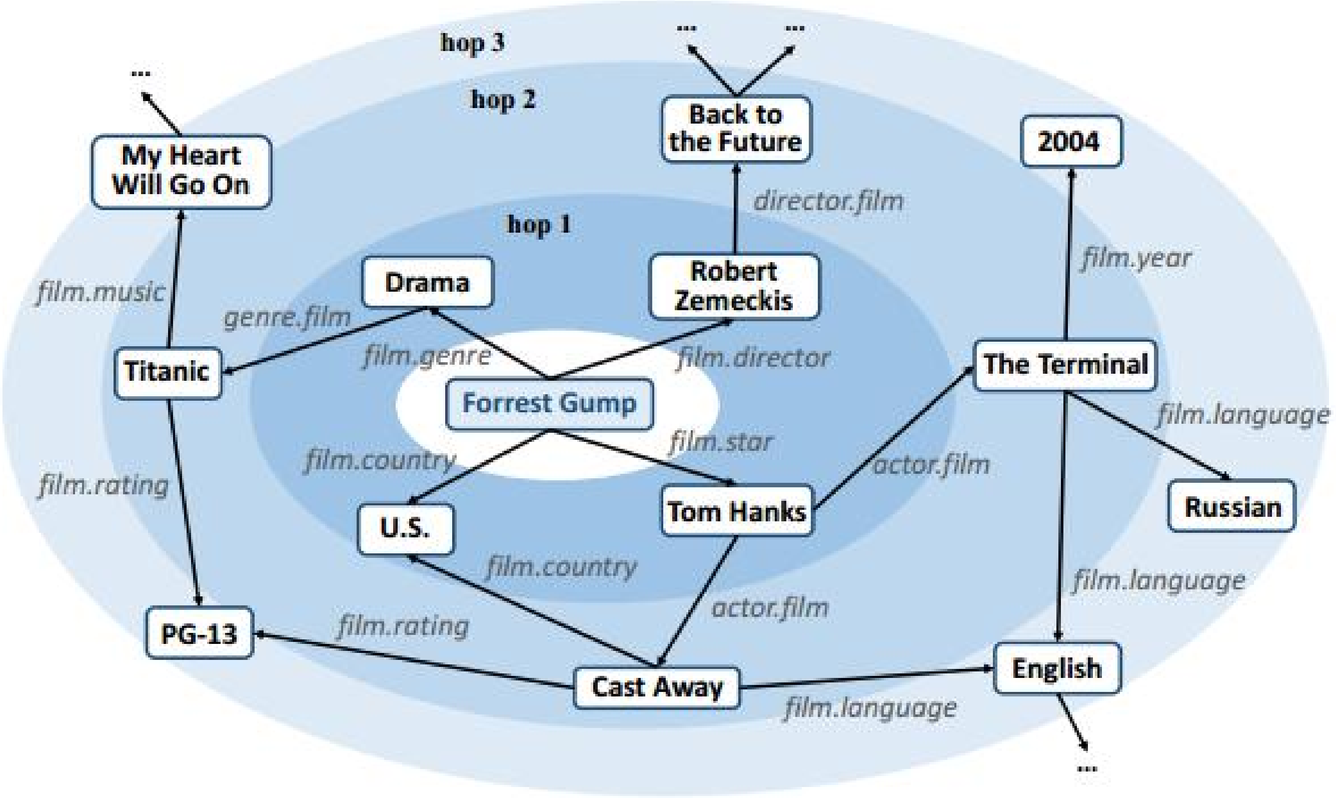

Figure 3: Illustration of ripple sets of "Forrest Gump" in KG of movies. The concentric circles denotes the ripple sets with different hops. The fading blue indicates decreasing relatedness between the center and surrounding entities.Note that the ripple sets of different hops are not necessarily disjoint in practice.

Framework

Definition 1 (relevant entity). *Given interaction matrix* $\mathbf{Y}$ *and knowledge graph* $\mathcal{G}$, *the set of* k*-hop relevant entities for user* u *is defined as*

$$\mathcal{E}_u^k = \{t \mid (h, r, t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{k-1}\}, \quad k = 1, 2, ..., H, \tag{2}$$

Definition 2 (ripple set). *The* k*-hop ripple set of user* u *is defined as the set of knowledge triples starting from* $\mathcal{E}_u^{k-1}$:

$$\mathcal{S}_u^k = \{(h, r, t) \mid (h, r, t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{k-1}\}, \quad k = 1, 2, ..., H. \tag{3}$$

Preference Propagation

$$p_i = \mathrm{softmax}\left(\mathbf{v}^{\mathrm{T}}\mathbf{R}_i\mathbf{h}_i\right) = \frac{\exp\left(\mathbf{v}^{\mathrm{T}}\mathbf{R}_i\mathbf{h}_i\right)}{\sum_{(h,r,t)\in\mathcal{S}_u^1}\exp\left(\mathbf{v}^{\mathrm{T}}\mathbf{R}\mathbf{h}\right)}, \tag{4}$$

where $\mathbf{R}_i \in \mathbb{R}^{d\times d}$ and $\mathbf{h}_i \in \mathbb{R}^d$ are the embeddings of relation r_i and head h_i, respectively. The relevance probability p_i can be regarded as the similarity of item $\mathbf{v}$ and the entity $\mathbf{h}_i$ measured in the space of relation $\mathbf{R}_i$. Note that it is necessary to take the embedding matrix

Preference Propagation

and the vector $\mathbf{o}_u$ is returned.

$$\mathbf{o}_u^1 = \sum_{(h_i, r_i, t_i) \in \mathcal{S}_u^1} p_i \mathbf{t}_i, \tag{5}$$

where $\mathbf{t}_i \in \mathbb{R}^d$ is the embedding of tail t_i. Vector $\mathbf{o}_u^1$ can be seen as

$\mathbf{o}_u^1, \mathbf{o}_u^2, ..., \mathbf{o}_u^H$. The embedding of user u with respect to item v is calculated by combining the responses of all orders:

$$\mathbf{u} = \mathbf{o}_u^1 + \mathbf{o}_u^2 + ... + \mathbf{o}_u^H, \tag{6}$$

Preference Propagation

and the vector $\mathbf{o}_u$ is returned:

$$\mathbf{o}_u^1 = \sum_{(h_i, r_i, t_i) \in \mathcal{S}_u^1} p_i \mathbf{t}_i, \tag{5}$$

where $\mathbf{t}_i \in \mathbb{R}^d$ is the embedding of tail t_i. Vector $\mathbf{o}_u^1$ can be seen as

$\mathbf{o}_u^1, \mathbf{o}_u^2, ..., \mathbf{o}_u^H$. The embedding of user u with respect to item v is calculated by combining the responses of all orders:

$$\mathbf{u} = \mathbf{o}_u^1 + \mathbf{o}_u^2 + ... + \mathbf{o}_u^H, \tag{6}$$

Learning Algorithm

$$\max p(\Theta|\mathcal{G}, \mathrm{Y}),$$

$$p(\Theta|\mathcal{G}, \mathrm{Y}) = \frac{p(\Theta, \mathcal{G}, \mathrm{Y})}{p(\mathcal{G}, \mathrm{Y})} \propto p(\Theta) \cdot p(\mathcal{G}|\Theta) \cdot p(\mathrm{Y}|\Theta, \mathcal{G})$$

Learning Algorithm

set $p(\Theta)$ as Gaussian distribution with zero mean and a diagonal covariance matrix:

$$p(\Theta) = \mathcal{N}(\mathbf{0},\ \lambda_1^{-1}\mathbf{I}). \tag{10}$$

3.6.3). In RippleNet, we use a three-way tensor factorization method to define the likelihood function for KGE:

$$\begin{aligned} p(\mathcal{G}|\Theta) &= \prod_{(h,r,t)\in\mathcal{E}\times\mathcal{R}\times\mathcal{E}} p\big((h,r,t)|\Theta\big) \\ &= \prod_{(h,r,t)\in\mathcal{E}\times\mathcal{R}\times\mathcal{E}} \mathcal{N}\big(I_{h,r,t} - \mathbf{h}^{\mathrm{T}}\mathbf{R}\mathbf{t},\ \lambda_2^{-1}\big), \end{aligned} \tag{11}$$

Learning Algorithm

term in Eq. (9) is the likelihood function of the observed implicit feedback given Θ and the KG, which is defined as the product of Bernouli distributions:

$$p(\mathrm{Y}|\Theta, \mathcal{G}) = \prod_{(u,v)\in \mathrm{Y}} \sigma(\mathbf{u}^{\mathrm{T}}\mathbf{v})^{y_{uv}} \cdot \left(1 - \sigma(\mathbf{u}^{\mathrm{T}}\mathbf{v})\right)^{1-y_{uv}} \tag{12}$$

$$\begin{aligned}
\min \mathcal{L} &= -\log\left(p(\mathrm{Y}|\Theta, \mathcal{G}) \cdot p(\mathcal{G}|\Theta) \cdot p(\Theta)\right) \\
&= \sum_{(u,v)\in \mathrm{Y}} -\Big(y_{uv} \log \sigma(\mathbf{u}^{\mathrm{T}}\mathbf{v}) + (1 - y_{uv}) \log\left(1 - \sigma(\mathbf{u}^{\mathrm{T}}\mathbf{v})\right)\Big) \\
&\quad + \frac{\lambda_2}{2} \sum_{r\in\mathcal{R}} \|\mathbf{I}_r - \mathbf{E}^{\mathrm{T}}\mathbf{R}\mathbf{E}\|_2^2 + \frac{\lambda_1}{2}\Big(\|\mathbf{V}\|_2^2 + \|\mathbf{E}\|_2^2 + \sum_{r\in\mathcal{R}} \|\mathbf{R}\|_2^2\Big)
\end{aligned}$$

Links to Existing Work

Attention Mechanism
Memory Networks
Knowledge Graph Embedding(KGE)

Experiment

Table 1: Basic statistics of the three datasets.

	MovieLens-1M	Book-Crossing	Bing-News
# users	6,036	17,860	141,487
# items	2,445	14,967	535,145
# interactions	753,772	139,746	1,025,192
# 1-hop triples	20,782	19,876	503,112
# 2-hop triples	178,049	65,360	1,748,562
# 3-hop triples	318,266	84,299	3,997,736
# 4-hop triples	923,718	71,628	6,322,548

Experiment

Table 3: The results of *AUC* and *Accuracy* in CTR prediction.

Model	MovieLens-1M		Book-Crossing		Bing-News	
	AUC	*ACC*	*AUC*	*ACC*	*AUC*	*ACC*
RippleNet*	**0.921**	**0.844**	**0.729**	**0.662**	**0.678**	**0.632**
CKE	0.796	0.739	0.674	0.635	0.560	0.517
SHINE	0.778	0.732	0.668	0.631	0.554	0.537
DKN	0.655	0.589	0.621	0.598	0.661	0.604
PER	0.712	0.667	0.623	0.588	-	-
LibFM	0.892	0.812	0.685	0.639	0.644	0.588
Wide&Deep	0.903	0.822	0.711	0.623	0.654	0.595

* Statistically significant improvements by unpaired two-sample t-test with $p = 0.1$.

Training Method

依次训练:Deep Knowledge-aware Network(DKN)
联合训练:Ripple Network
交替训练:Multi-task Learning for KG enhanced Recommendation