The 27th ACM International Conference on Information and Knowledge Management (CIKM 2018)
Lingotto, Turin, Italy, 22-26 October 2018

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
[2] Microsoft Research Asia
[3] Meituan-Dianping Group
[4] The Hong Kong Polytechnic University

October 23, 2018

上海交通大學 Shanghai Jiao Tong University   Microsoft Research 微软亚洲研究院   美团 美团点评   The Hong Kong Polytechnic University 香港理工大學

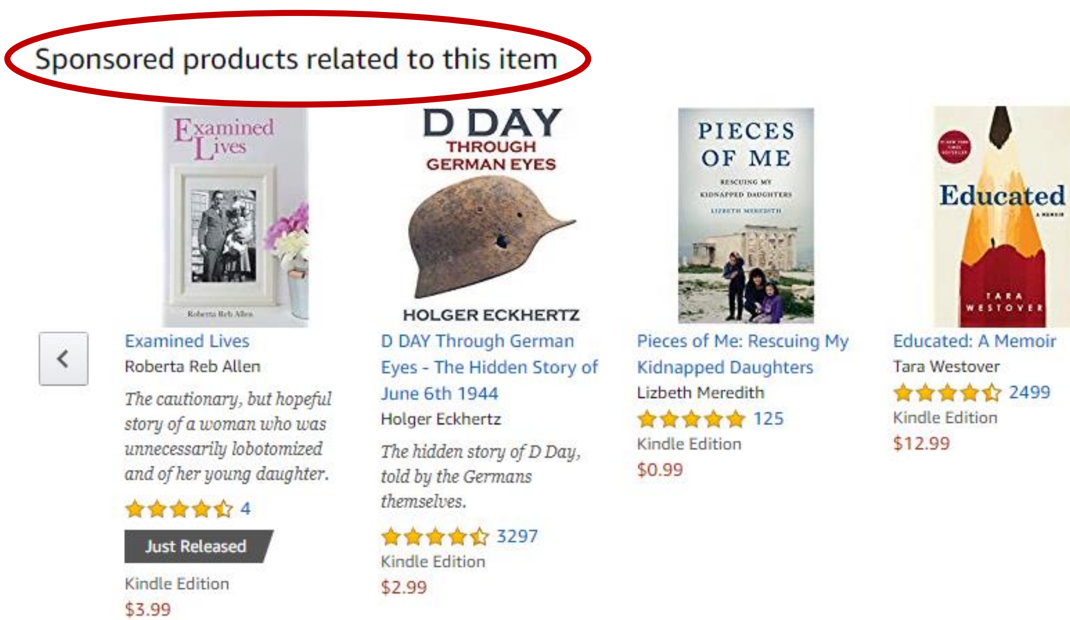
Recommender Systems

Recommender systems (RS) intend to address the information explosion by finding a small set of items for users to meet their personalized interests

Movie (IMDb)

Recommender Systems

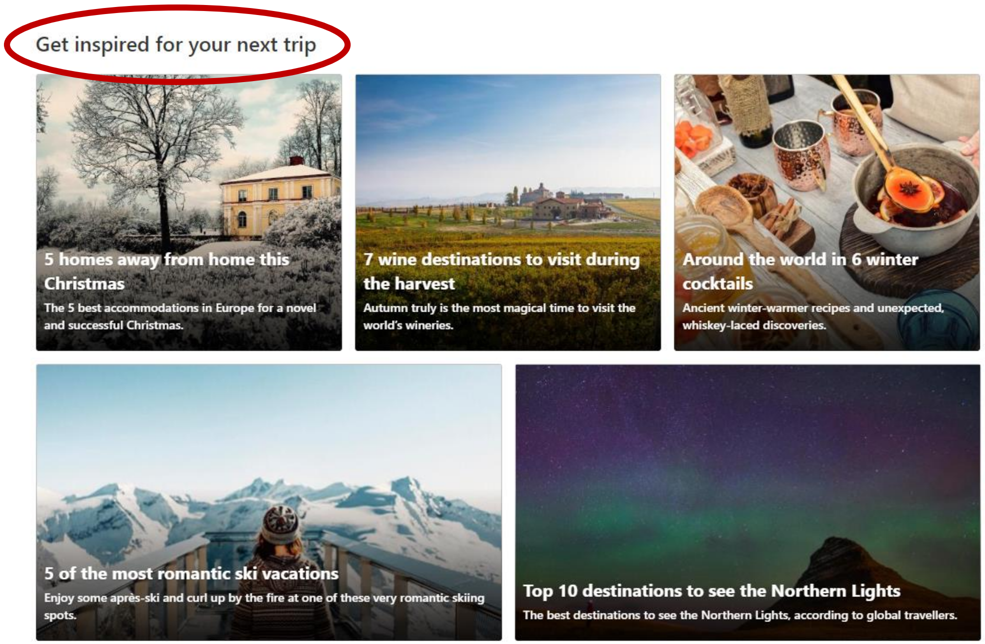
Recommender systems (RS) intend to address the information explosion by finding a small set of items for users to meet their personalized interests



Book (Amazon)

Recommender Systems

Recommender systems (RS) intend to address the information explosion by finding a small set of items for users to meet their personalized interests



Trip (Booking)

Rating/CTR Prediction

Explicit feedback

2	?	3	?
?	?	4	?
?	5	?	2
3	1	4	?

Rating prediction

Implicit feedback

1	?	0	?
?	?	1	?
?	0	?	1
0	1	0	?

Click-through rate
(CTR) prediction

Collaborative Filtering

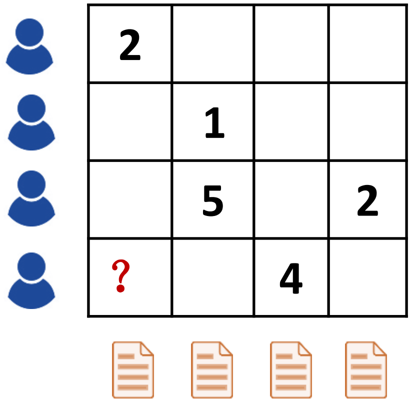
- Collaborative filtering (CF) assumes that similar uses (with respect to their historical records) have similar preferences
  - Matrix factorization

$$R_{pq} = \mathbf{p}^\top \mathbf{q}$$

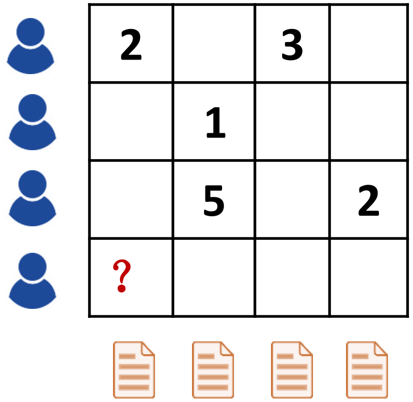
$$L = \| \boldsymbol{R} - \boldsymbol{P}^{\mathrm{T}} \boldsymbol{Q} \|_2^2 + \| \boldsymbol{P} \|_2^2 + \| \boldsymbol{Q} \|_2^2$$

CF Fails To Address …

- Sparsity of user-item interactions
- Cold start problem

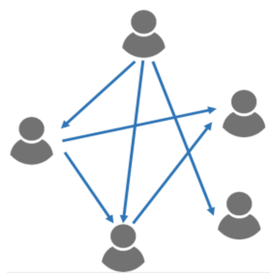


Sparsity



Cold start

CF + Side Information



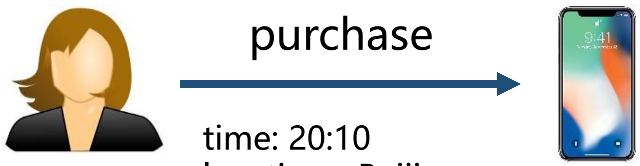
Social network

Alice
Female
California
…

iPhone X
2017
5.8 inch
$999
…

User/item attributes

Multimedia (image, text, video, audio …)



time: 20:10
location: Beijing
What else in carts: …

Context

Knowledge Graph

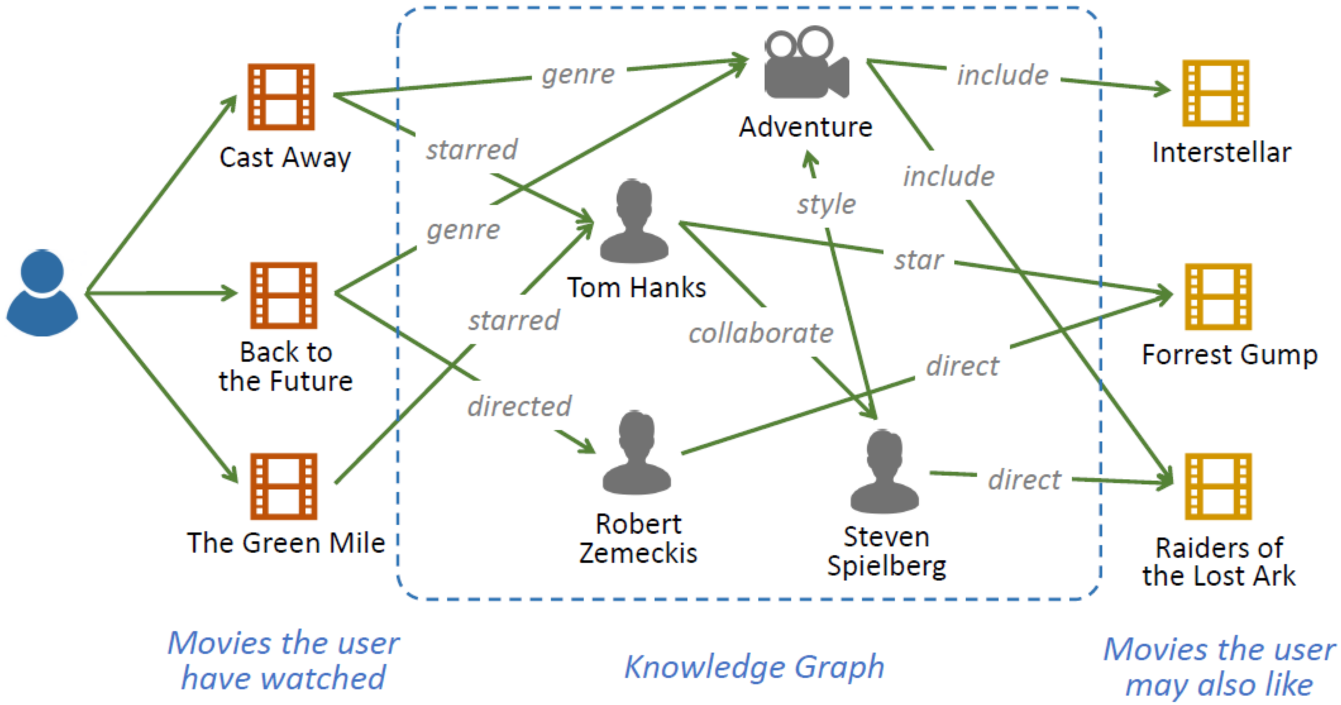
- A **knowledge graph** (KG) is a type of directed heterogeneous graph in which nodes correspond to **entities** and edges correspond to **relations**
- A KG usually consists of massive triples **(head, relation, tail)**

Forrest Gump —*film.film.star*→ Tom Hanks

Tom Hanks —*people.people.date_of_brith*→ July 7, 1956

Knowledge Graph

A snippet of knowledge graph in movie recommendation



KG-aware RS

- **Embedding-based methods** pre-process a KG with knowledge graph embedding (KGE) algorithms, then incorporate the learned entity embeddings into a recommendation framework
  - Collaborative Knowledge base Embedding (CKE) [KDD 16]
  - Deep Knowledge-aware Network (DKN) [WWW 18]
  - Signed Heterogeneous Information Network Embedding (SHINE) [WSDM 18]
  - Knowledge-enhanced Sequential Recommender (KSR) [SIGIR 18]

KG-aware RS

- **Path-based methods** explore the various patterns of connections among items in KG to provide additional guidance for recommendations
  - Personalized Entity Recommendation (PER) [WSDM 14]
  - Factorization Machine with Group lasso (FMG) [KDD 17]

Pros and Cons

- **Embedding-based methods**
  - have high flexibility, but …
  - the adopted KGE algorithms focus more on modeling rigorous semantic relatedness
- **Path-based methods**
  - make use of KG in a more natural and intuitive way, but …
  - they rely heavily on manually designed meta-paths/meta-graphs

Our Work

- We propose **RippleNet**, an end-to-end framework for KG-aware recommendation
- The key idea of RippleNet is **preference propagation**
- RippleNet combines the advantages of the two types of methods:
  - RippleNet incorporates the KGE methods into RS by preference propagation
  - RippleNet can automatically discover possible meta-paths

Problem Formulation

- Users: $\mathcal{U} = \{u_1, u_2, \dots\}$, items: $\mathcal{V} = \{v_1, v_2, \dots\}$
- User-item interaction (implicit feedback):

$$\mathrm{Y} = \{y_{uv} \in \{0,1\} \mid u \in \mathcal{U}, v \in \mathcal{V}\}$$

- Knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R})$:
  - $\mathcal{G} = \{(h, r, t) \mid h \in \mathcal{E},\ r \in \mathcal{R},\ t \in \mathcal{E}\}$
  - $\mathcal{V} \subseteq \mathcal{E}$ (each item in $\mathcal{V}$ associates with an entity in $\mathcal{E}$)

- Goal: learning a prediction function

$$\hat{y}_{uv} = \mathcal{F}(u, v;\ \Theta) \in [0,1]$$

Relevant Entity

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

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

  - $\mathcal{E}_u^0 = \mathcal{V}_u = \{v \mid y_{uv} = 1\}$ is the set of the user's clicked items in the past

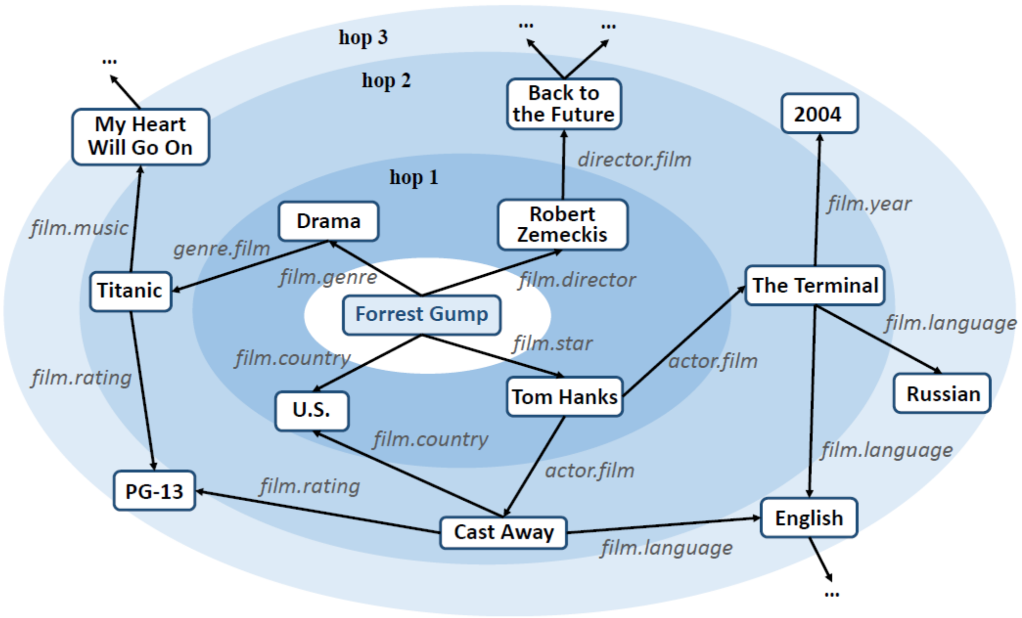
Ripple Set

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

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

Ripple Set

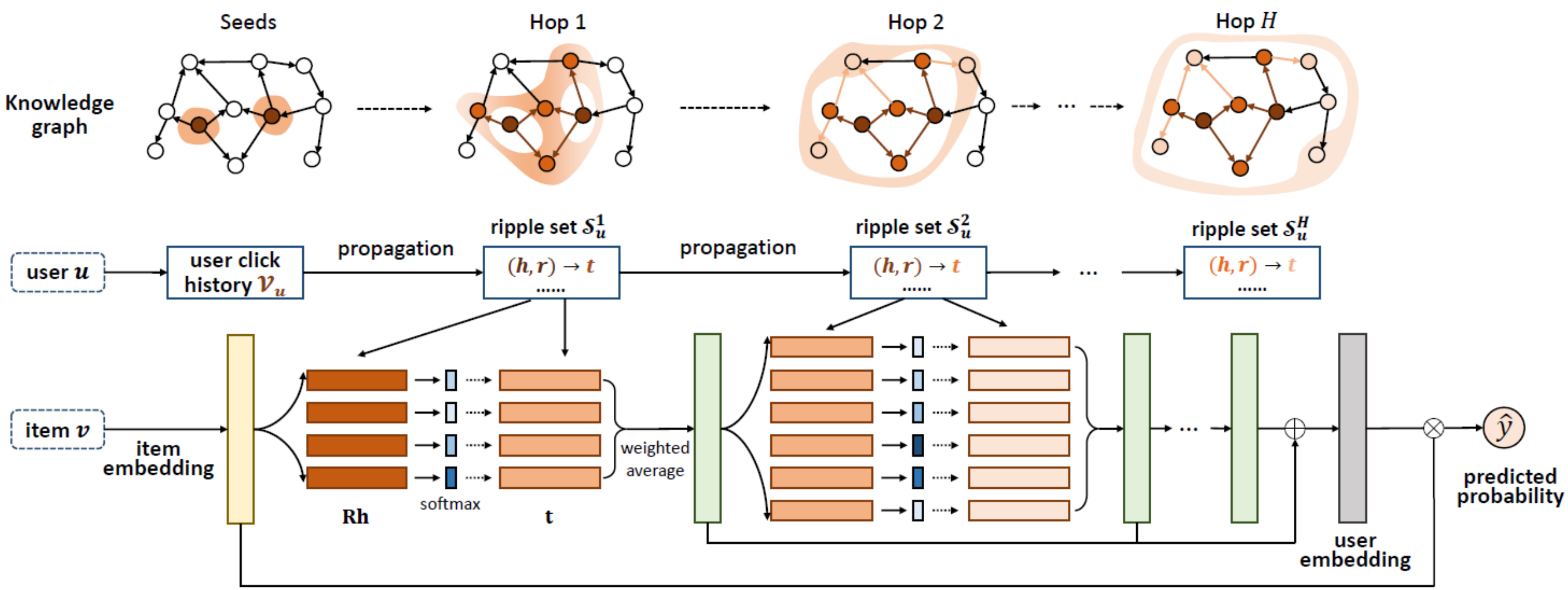
$$\mathcal{S}_u^k = \{(h, r, t) \mid (h, r, t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{k-1}\}, \quad k = 1, 2, \ldots, H.$$



Ripple set of "Forrest Gump" in the knowledge graph

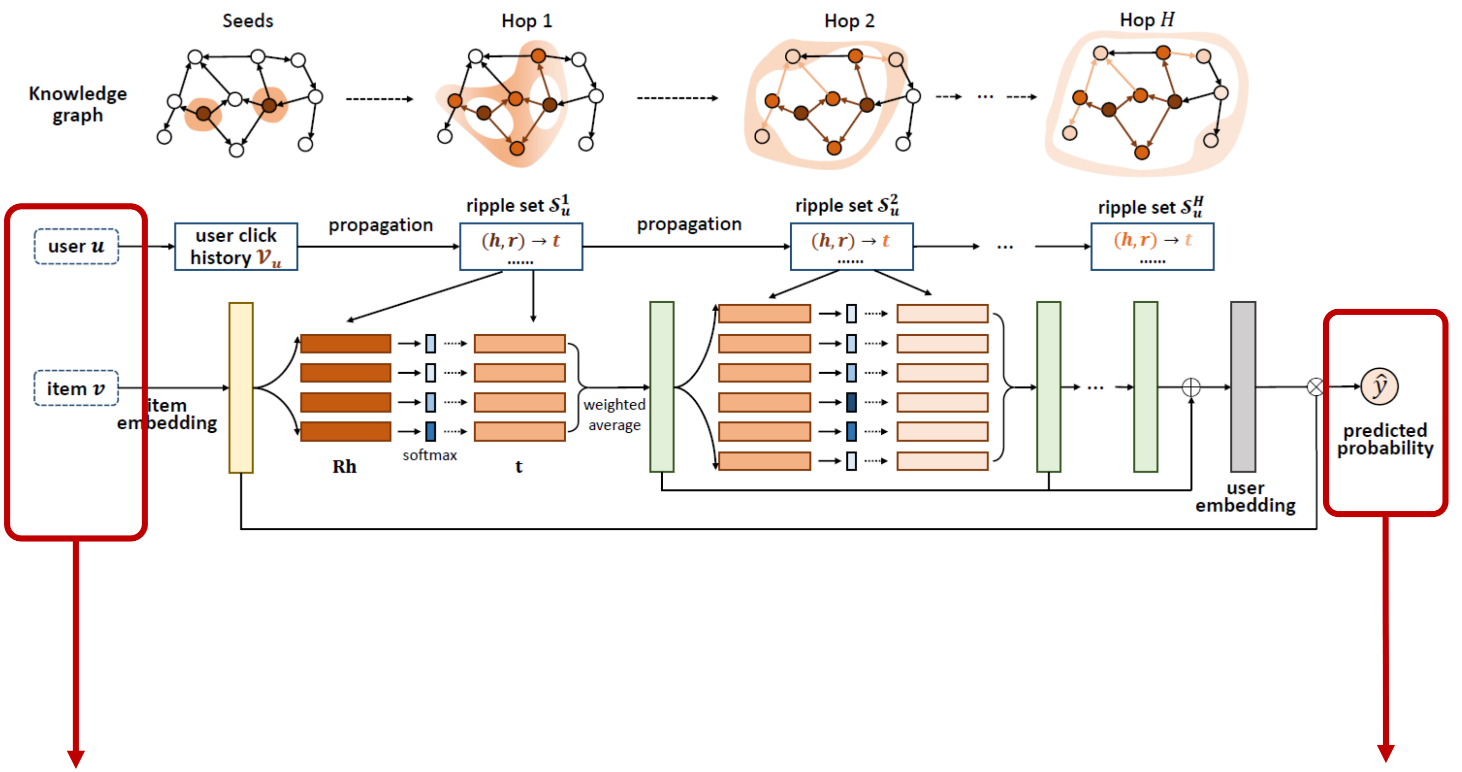
Ripples created by water droplet

Framework



Framework of RippleNet

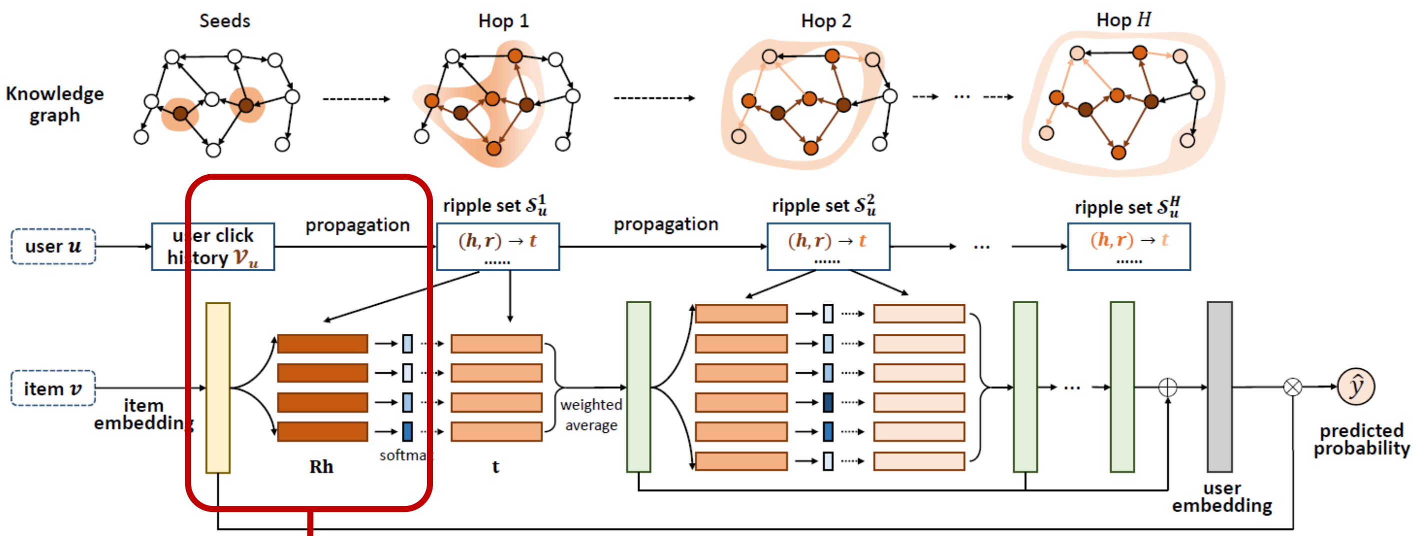
Preference Propagation



input

output

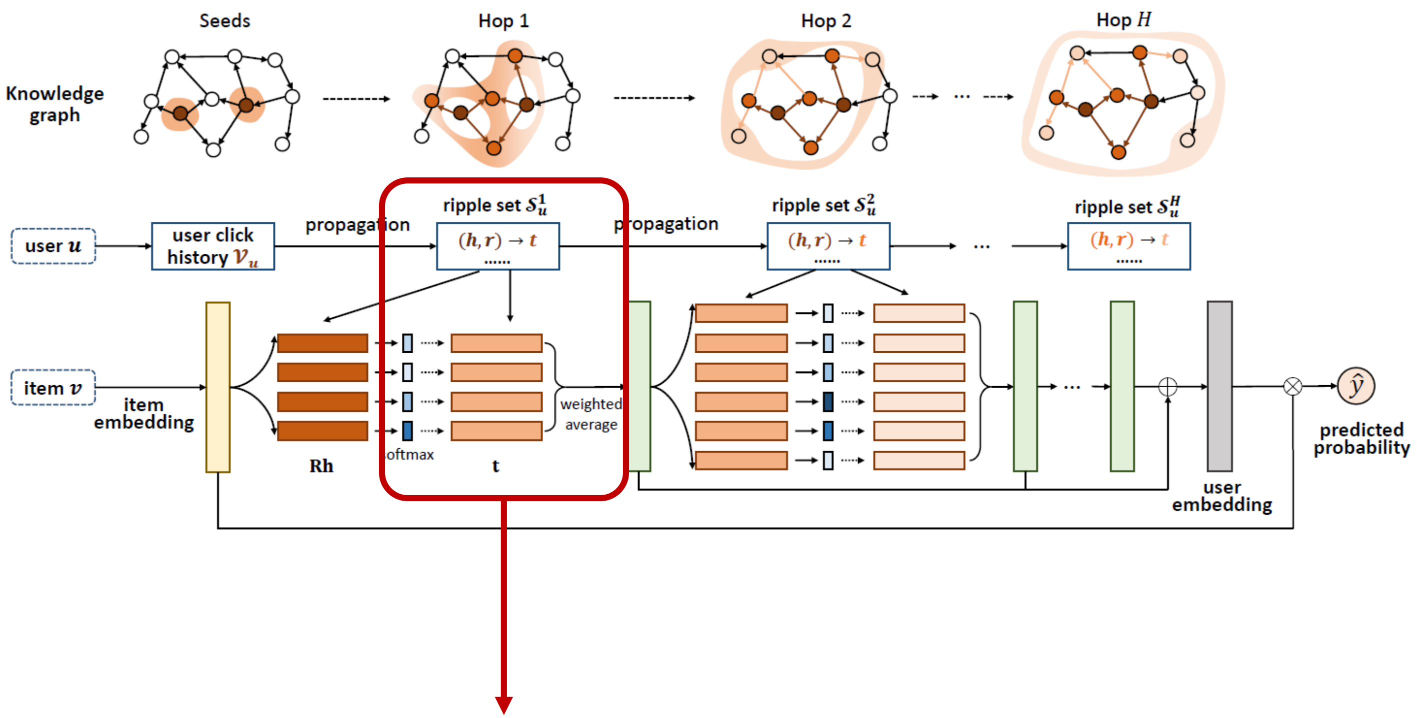
Preference Propagation



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

Relevance probability

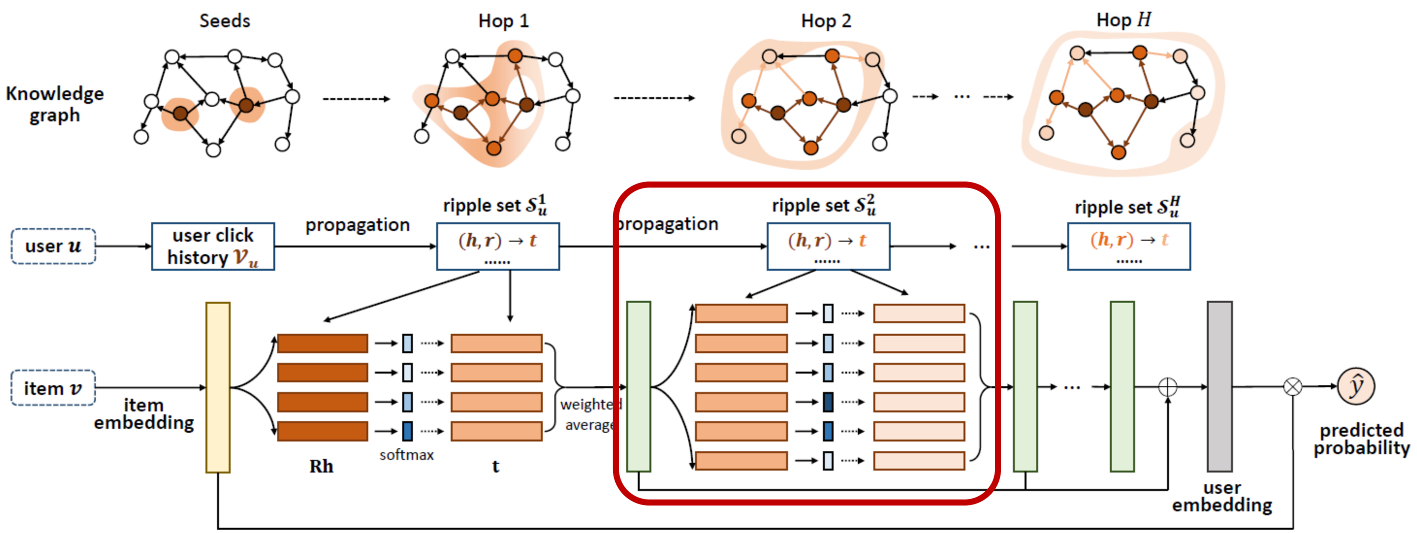
Preference Propagation



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

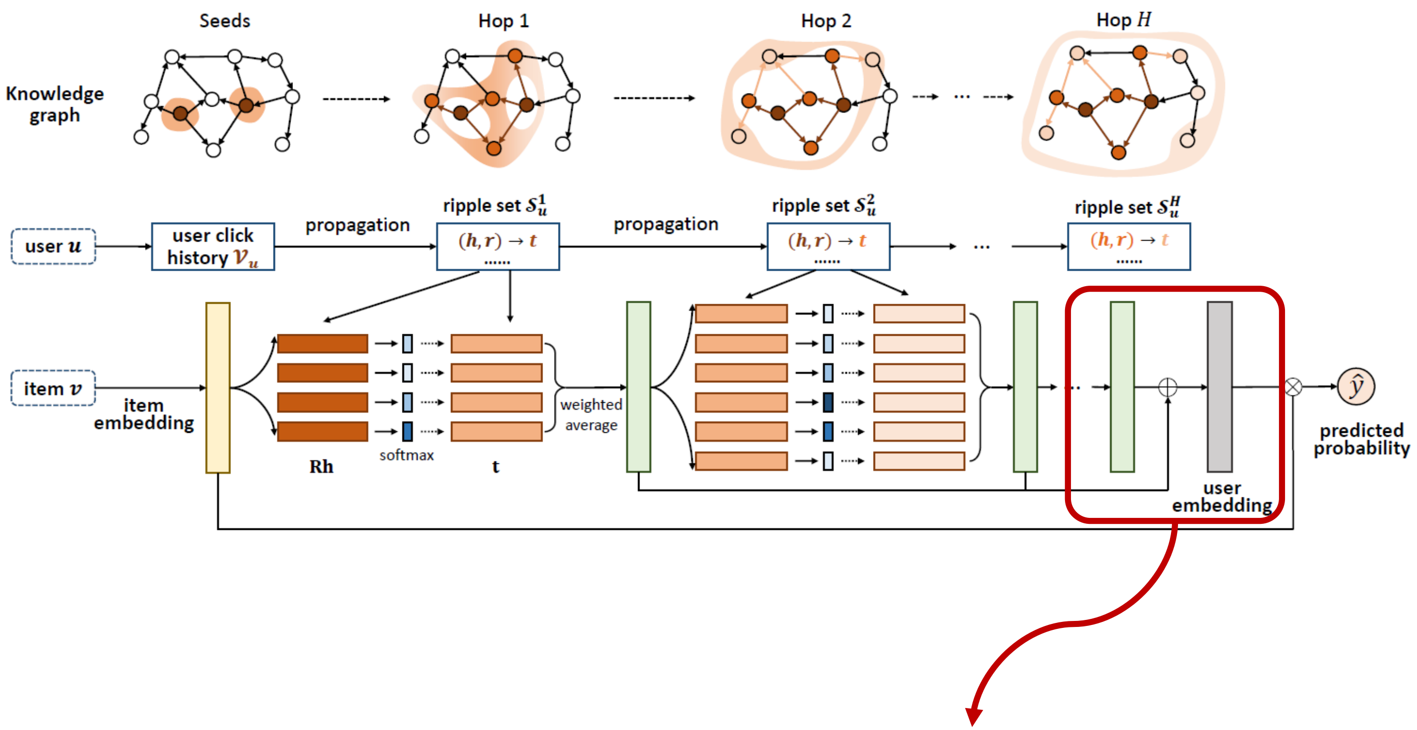
User's 1-order response

Preference Propagation



Repeat the propagation …

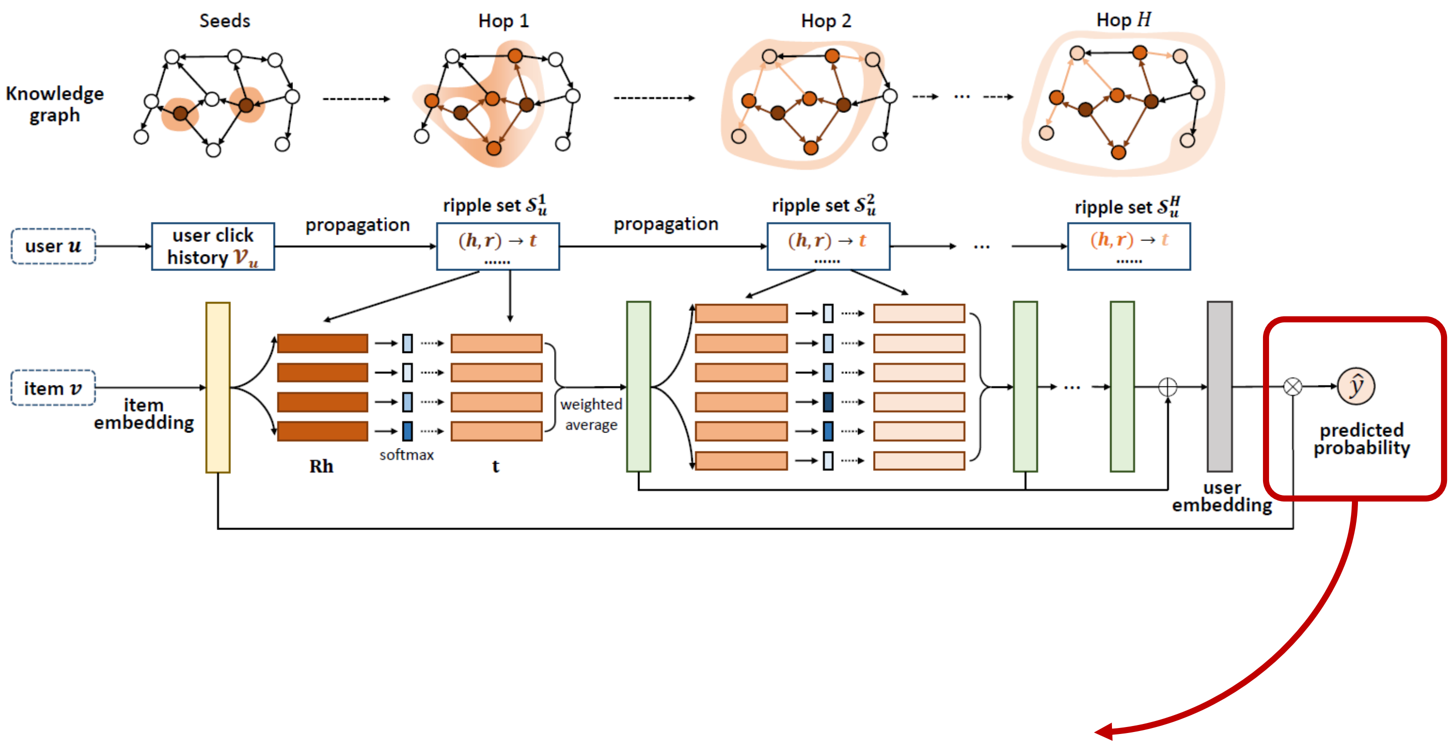
Preference Propagation



$$\mathbf{u} = \mathbf{o}_u^1 + \mathbf{o}_u^2 + \ldots + \mathbf{o}_u^H$$

User embedding

Preference Propagation



$$\hat{y}_{uv} = \sigma(\mathbf{u}^{\mathrm{T}}\mathbf{v})$$

Predicted probability

Learning Algorithm

- Maximizing the posterior probability

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

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

- The priori probability of model parameters Θ:

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

Learning Algorithm

- Maximizing the posterior probability

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

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

- The likelihood function of the observed knowledge graph $\mathcal{G}$ given Θ (knowledge graph embedding):

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

Learning Algorithm

- Maximizing the posterior probability

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

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

- The likelihood function of the observed implicit feedback given Θ and $\mathcal{G}$ (Bernouli distributions)

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

Learning Algorithm

- Maximizing the posterior probability

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

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

- Loss function

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

Datasets

- MovieLens-1M (movie recommendation)
- Book-Crossing (book recommendation)
- Bing-News (news recommendation)

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 Setup

- Explicit feedback -> implicit feedback
- Source of the knowledge graph: **Microsoft Satori**
- Entity linking
- Constructing knowledge sub-graph for each dataset

Table 2: Hyper-parameter settings for the three datasets.

MovieLens-1M	$d = 16, H = 2, \lambda_1 = 10^{-7}, \lambda_2 = 0.01, \eta = 0.02$
Book-Crossing	$d = 4, H = 3, \lambda_1 = 10^{-5}, \lambda_2 = 0.01, \eta = 0.001$
Bing-News	$d = 32, H = 3, \lambda_1 = 10^{-5}, \lambda_2 = 0.05, \eta = 0.005$

Baselines

KG-aware methods

- **CKE** [KDD 16]: Collaborative filtering + TransR
- **SHINE** [WSDM 18]: Autoencoder
- **DKN** [WWW 18]: Convolutional neural networks
- **PER** [WSDM 14]: Meta-path

Generic methods

- **LibFM**: Factorization machines
- **Wide&Deep**: deep neural network for RS

Results —— CTR Prediction

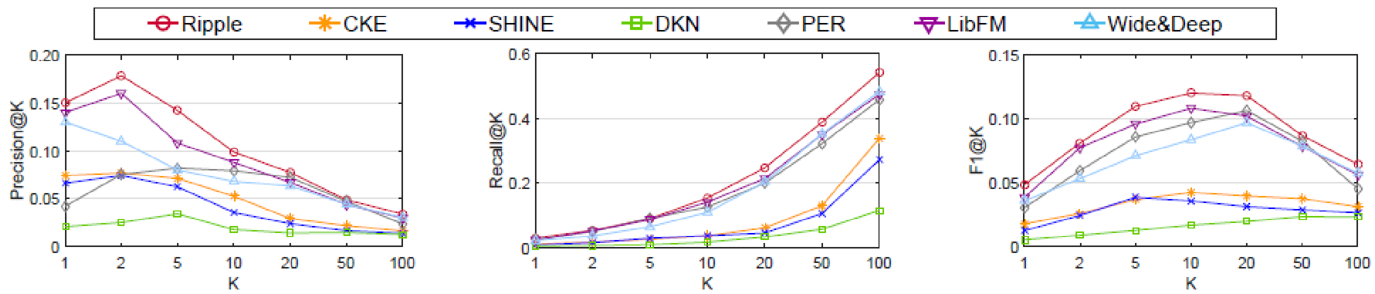
Table 3: The results of *AUC* and *Accuracy* in CTR prediction.

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 $p = 0.1$.

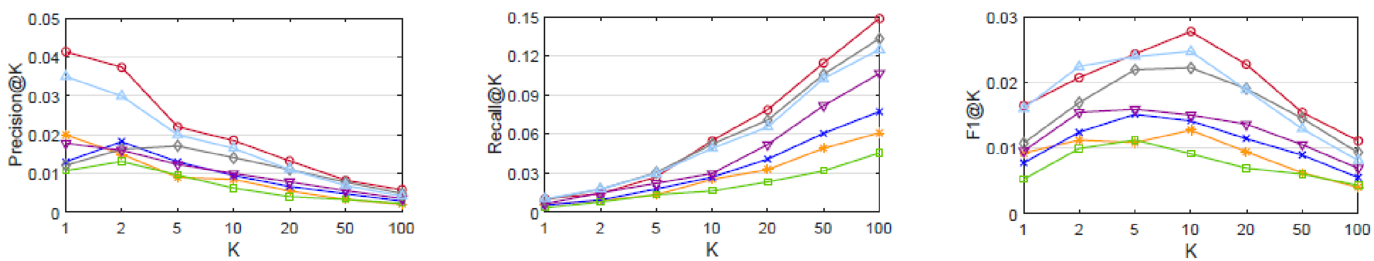
Improvements on AUC	2.0% ~ 40.6%	2.5% ~ 17.4%	2.6% ~ 22.4%

Results —— Top-K Recommendation



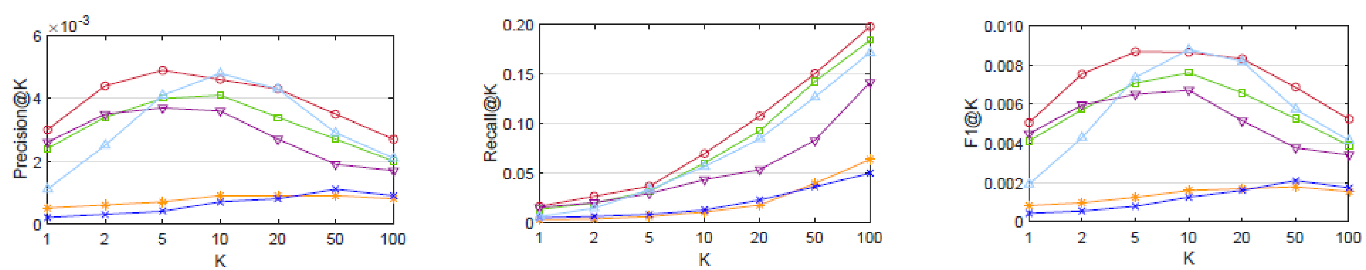
(a) *Precision@K* (b) *Recall@K* (c) *F1@K*

Figure 5: *Precision@K*, *Recall@K*, and *F1@K* in top-*K* recommendation for MovieLens-1M.



(a) *Precision@K* (b) *Recall@K* (c) *F1@K*

Figure 6: *Precision@K*, *Recall@K*, and *F1@K* in top-*K* recommendation for Book-Crossing.



(a) *Precision@K* (b) *Recall@K* (c) *F1@K*

Figure 7: *Precision@K*, *Recall@K*, and *F1@K* in top-*K* recommendation for Bing-News.

Results —— Parameter Sensitivity

Size of ripple set

Table 4: The results of *AUC* w.r.t. different sizes of a user's ripple set.

Size of ripple set	2	4	8	16	32	64
MovieLens-1M	0.903	0.908	0.911	0.918	**0.920**	0.919
Book-Crossing	0.694	0.696	0.708	**0.726**	0.706	0.711
Bing-News	0.659	0.672	0.670	0.673	**0.678**	0.671

Hop number

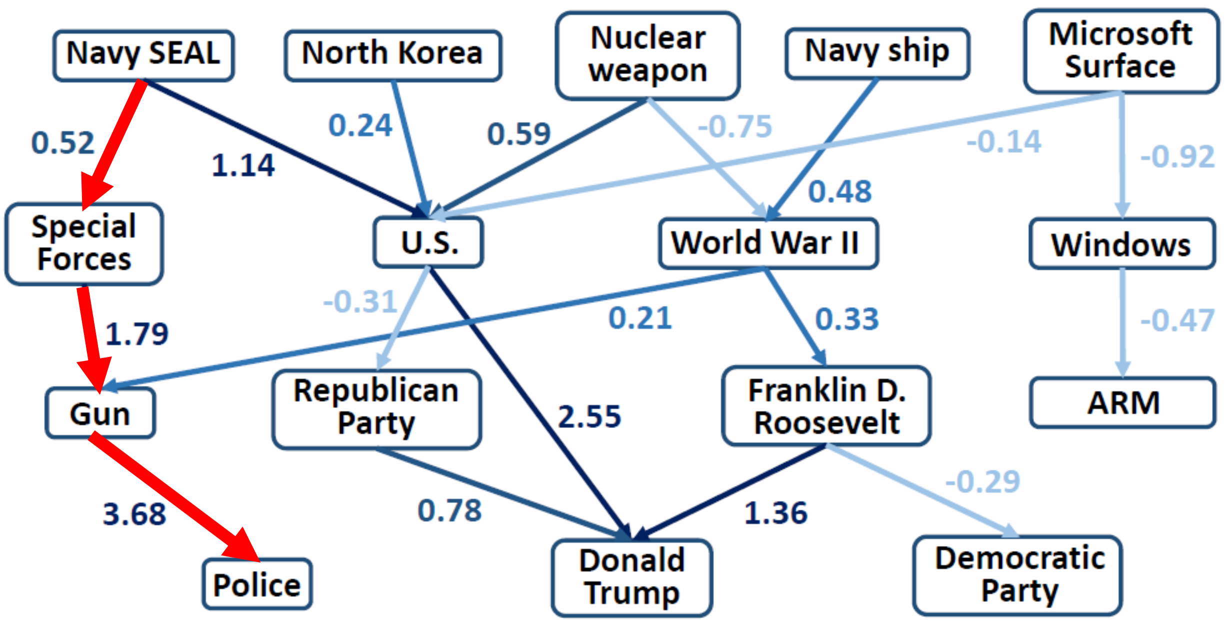
Table 5: The results of *AUC* w.r.t. different hop numbers.

Hop number H	1	2	3	4
MovieLens-1M	0.916	**0.919**	0.915	0.918
Book-Crossing	0.727	0.722	**0.730**	0.702
Bing-News	0.662	0.676	**0.679**	0.674

Results —— Explainability

Click history:

1. *Family of **Navy SEAL** Trainee Who Died During Pool Exercise Plans to Take Legal Action*
2. ***North Korea** Vows to Strengthen **Nuclear Weapons***
3. ***North Korea** Threatens 'Toughest Counteraction' After **U.S.** Moves **Navy Ships***
4. *Consumer Reports Pulls Recommendation for **Microsoft Surface** Laptops*



Candidate news: ***Trump** Announces Gunman Dead, Credits 'Heroic Actions' of **Police***

Summary

- We propose **RippleNet**, an end-to-end framework for KG-aware recommendation
- The key idea of RippleNet is **preference propagation**, which automatically propagates users’ preferences and explores their hierarchical interests in the KG
- RippleNet **combines the advantages** of embedding-based and path-based methods
- Experiment results demonstrate the **efficacy** and **explainability** of RippleNet

Q & A

Thanks !

- This work was partially sponsored by National Basic Research 973 Program of China (2015CB352403), National Natural Science Foundation of China (61272291), and SIGIR Student Travel Grant.
- The code is available at https://github.com/hwwang55/RippleNet.