SIGIR18 Papers about Recommendation

Wu, 2019/4/2

Overview

- Online recommendation(1)
- Recommendation with Social Networks(2+1)
  - Group representation, community detection, sequence-aware Rec
- Recommendation with Knowledge Base(1)
- Improve traditional methods(3)
  - APR, CMN, Bandit problem
- Some specific tasks(5)
  - Recommend email, mention, citation, Wikipedia article section
  - conversational recommender system
- User modeling: Geo-social based(1)

Some useful basics

- Recommendation
  - MF, BPR
- Sequence-aware recommendation
  - RNN, LSTM, GRU, Memory network
- Clustering
- Learn image representation
  - CNN, Inception, VGG, ResNet
- Graph embedding

Online recommendation

- In real life, new ratings, users and items come continually

- Four challenges of online recommendation
  - Online updating and advoiding overload
  - Capturing users' long-term interests
  - Capturing users' drifted interests
  - Modeling new users and items

RKMF

- **MF** based

- Assumption
  - the model build from **S**(ratings set) and the model build from **SU**$\{r_{u,i}\}$ is mostly the same from a global perspective

  - For new-user, his (local) features might change a lot from the new rating

RKMF: Online updating

- Update(retrain) the related user or item only

procedure USERUPDATE($S, W, H, r_{u,i}$)
    $S \leftarrow S \cup \{r_{u,i}\}$
    return USERRETRAIN(S, W, H, u)
end procedure

procedure USERRETRAIN(S, W, H, u^*)
    initialize u^*-th row in W
    repeat
        for $r_{u,i} \in C(u^*, \cdot)$ **do**
            for $f \leftarrow 1, \ldots, f$ **do**
                $w_{u,f} \leftarrow w_{u,f} - \alpha \frac{\partial}{\partial w_{u,f}} \text{Opt}(S, W, H)$
            end for
        end for
    until Stopping criteria met
    return (W, H)
end procedure

Figure 3: Online updates for new-user problem.

procedure ADDRATING($S, W, H, r_{u,i}$)
    $S \leftarrow S \cup \{r_{u,i}\}$
    return UPDATERATING($S, W, H, r_{u,i}$)
end procedure

procedure REMOVERATING($S, W, H, r_{u,i}$)
    $S \leftarrow S \setminus \{r_{u,i}\}$
    return UPDATERATING($S, W, H, r_{u,i}$)
end procedure

procedure UPDATERATING($S, W, H, r_{u,i}$)
    if $P_u(\text{train}|r_{u,i}) >$ RANDOM **then**
        $(W, H) \leftarrow$ USERRETRAIN(S, W, H, u)
    end if
    if $P_i(\text{train}|r_{u,i}) >$ RANDOM **then**
        $(W, H) \leftarrow$ ITEMRETRAIN(S, W, H, i)
    end if
    return (W, H)
end procedure

Figure 4: General algorithm for online-updates.

Rendle S, Schmidt-Thieme L. Online-updating regularized kernel matrix factorization models for large-scale recommender systems[C]//Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008: 251-258.

RMFX

- **BPR** based
- Use **reservoir** to capture users/ long-term interests
- **Reservoir**: a batch of ratings
- With probability |R|/t and replaces uniformly at random an instance from the reservoir, the reservoir is a random sample of the current dataset.

RMFX Framework

Input:

Reservoir representing a sample of the stream at time t: R; Regularization parameters λ_W, λ_{H^+}, and λ_{H^-}; Learning rate η_0; Learning rate schedule α; Number of iterations T_S, and T_θ; Parameter c to control how often to perform the model updates.

Output: $\theta = (\mathbf{W}, \mathbf{H})$

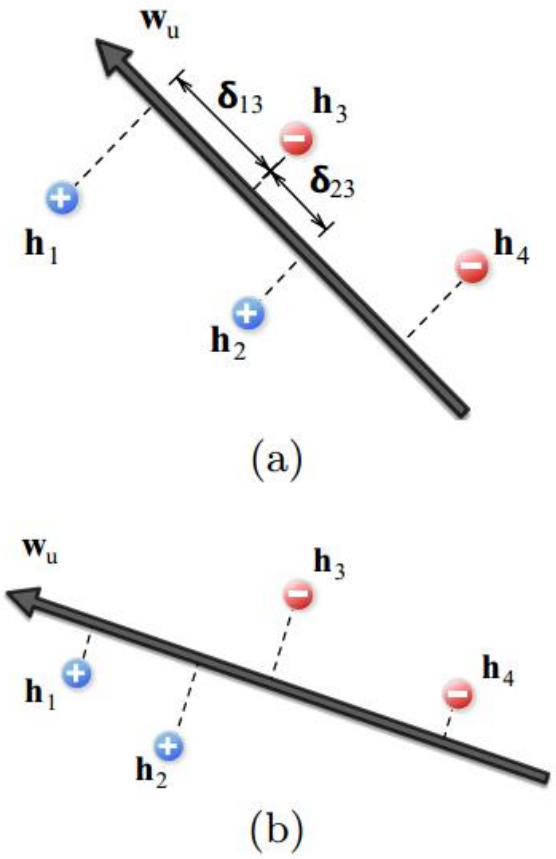
initialize $\mathbf{W}_0$ and $\mathbf{H}_0$
initialize sample stream $S' \leftarrow \emptyset$
counter $\leftarrow 0$
for $t = 1$ **to** T_S **do**
    $R \leftarrow$ **updateReservoir**(R)
    counter $\leftarrow$ counter $+ 1$
    if $c =$ counter **then**
        $\theta \leftarrow$ **updateModel**$(S_t, \lambda_W, \lambda_{H^+}, \lambda_{H^-}, \eta, \alpha, T_\theta)$
        counter $\leftarrow 0$
    end if
end for
return $\theta_T = (\mathbf{W}_T, \mathbf{H}_T)$

Diaz-Aviles E, Drumond L, Schmidt-Thieme L, et al. Real-time top-n recommendation in social streams[C]//Proceedings of the sixth ACM conference on Recommender systems. ACM, 2012: 59-66.

RMFX

- It is still far from the accuracy achieved by the offline cases if use the **reservoir** data for online updating

- How to sample the pairs needed for creating the contrasts



RMFX: online update

RMFX Model Update based on SGD for MF using active learning with small buffers

Input:

Reservoir representing a sample of the stream at time t: R; Regularization parameters λ_W, λ_{H^+}, and λ_{H^-}; Learning rate η_0; Learning rate schedule α; Number of iterations T_θ.

Output: $\theta = (\mathbf{W}, \mathbf{H})$

procedure UPDATEMODEL($S_t, \lambda_W, \lambda_{H^+}, \lambda_{H^-}, \eta_0, \alpha, T_\theta$)
  for $t = 1$ **to** T_θ **do**
    Select a user-item pair (u, i) from R uniformly at random
    Construct a small buffer for user u by sampling 59 negative items j's from R ("59 trick" [16, 6])
    Compute the distances δ_{uijb} for each pair $p_b = ((u, i), (u, j_b)) \in P$, $b = 1 \ldots 59$ in the small buffer
    Sample a pair $p^* = ((u, i), (u, j))$ from the buffer with probability proportional to its *informativeness:* $1/\delta_{uijb}$
    // Perform the model updates as follows:
    $y_{uij} \leftarrow sign(x_{ui} - x_{uj})$
    $\mathbf{w}_u \leftarrow \mathbf{w}_u + \eta\, y_{uij}\, (\mathbf{h}_i - \mathbf{h}_j) - \eta\, \lambda_W\, \mathbf{w}_u$
    $\mathbf{h}_i \leftarrow \mathbf{h}_i + \eta\, y_{uij}\, \mathbf{w}_u - \eta\, \lambda_{H^+}\, \mathbf{h}_i$
    $\mathbf{h}_j \leftarrow \mathbf{h}_j + \eta\, y_{uij}\, (-\mathbf{w}_u) - \eta\, \lambda_{H^-}\, \mathbf{h}_j$
    $\eta = \alpha \cdot \eta$
  end for
  return $\theta = (\mathbf{W}_{T_\theta}, \mathbf{H}_{T_\theta})$
end procedure

SPMF

- Binary classification with negative sampling Based(likely)

$$O(x_{ij}) = -\left(x_{ij} log\sigma(f(i, j; \Theta)) + (1 - x_{ij}) log(1 - \sigma(f(i, j; \Theta)))\right)$$

- Update model use both reservoir and new data
- Similar methods to sample train example from reservoir and new data: tend to sample the data that has **low** score

$$w_{ij} = exp(\frac{rank_{ij}}{|\mathcal{W} \cup \mathcal{R}|}) \quad (13)$$

$$P(x_{ij}) = \frac{w_{ij}}{\sum_{x_{ij} \in \mathcal{W} \cup \mathcal{R}} w_{ij}} \quad (14)$$

Wang W, Yin H, Huang Z, et al. Streaming ranking based recommender systems[C]//The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. ACM, 2018: 525-534.

SPMF algorithm

$\mathcal{R}' = \emptyset$;
foreach *data instance* $\{u_i, v_j\} \in \{\mathcal{R} \cup \mathcal{W}\}$ **do**
    if u_i *or* v_j *is a new user or a new item* **then**
        Generate $\boldsymbol{u}_i$ or $\boldsymbol{v}_j$ based on Gaussian prior distribution;
    end
    else
        Get $\boldsymbol{u}_i$ or $\boldsymbol{v}_j$ from Θ^t;
    end
    Compute the score $f(i, j; \Theta^t) = \boldsymbol{u}_i \cdot \boldsymbol{v}_j$;
    if *this is an instance in* $\mathcal{W}$ **then**
        Sample this instance with probability $|\mathcal{R}|/(t + i')$ where $1 \leq i' \leq |\mathcal{W}|$ to a temporary set $\mathcal{R}'$;
    end
end
Rank data instances in $\{\mathcal{R} \cup \mathcal{W}\}$ decreasingly according to $f(i, j; \Theta^t)$ to get a rank $rank(x_{ij})$ for each x_{ij};
Compute w_{ij} for each x_{ij} in $\{\mathcal{R} \cup \mathcal{W}\}$ using Equation 13;
Compute $P(x_{ij})$ for each x_{ij} in $\{\mathcal{R} \cup \mathcal{W}\}$ using Equation 14;
while *next window of data have not arrived* **do**
    Sample a data instance x_{ij} from $\mathcal{R} \cup \mathcal{W}$ according to $P(x_{ij})$;
    Sample N negative examples x_{in} from $\mathcal{D}_t^- = \mathcal{D} - \mathcal{R} - \mathcal{W}$ by fixing the user u_i;
    Update the gradients of the latent variables associated with users and items in the batch based on Equation 10 and 11;
    Update the latent variables by batch SGD;
end
Replace the data in $\mathcal{R}$ randomly with the data in $\mathcal{R}'$;

sRec

- Sequence PMF

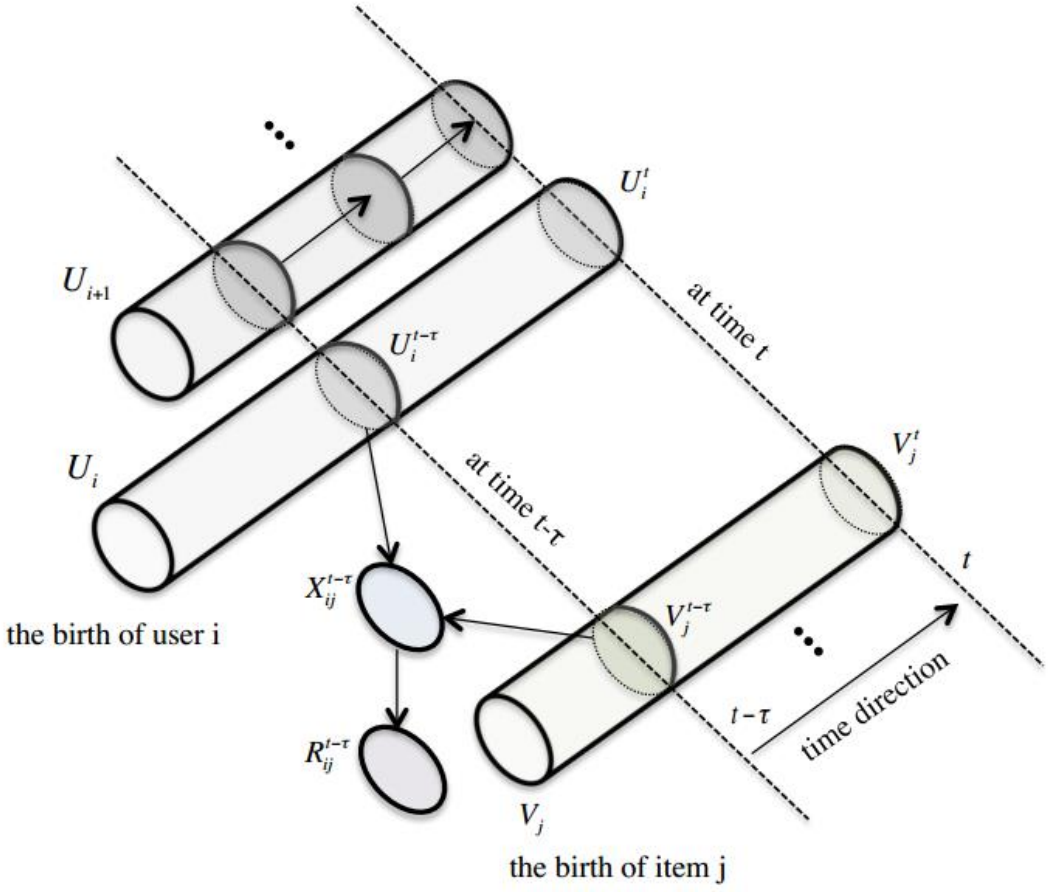
$$U_i^t | U_i^{t-\tau} \sim \mathcal{N}\left(U_i^{t-\tau}, \sigma_U^2 \tau I\right), \tau > 0.$$

- New user

$$U_i^0 \sim \mathcal{N}(0, I), \forall i, t_U(i) = 0$$

$$U_i^{t_U(i)} | \left\{R_{ij}^t : t < t_U(i)\right\}$$
$$\sim \mathcal{N}\left(\underset{k:t_U(k)<t_U(i)}{\text{mean}} \mathbb{E}\left(U_k^{t_U(i)} | \left\{R_{ij}^t : t < t_U(i)\right\}\right), \sigma_{U0}^2 I\right)$$

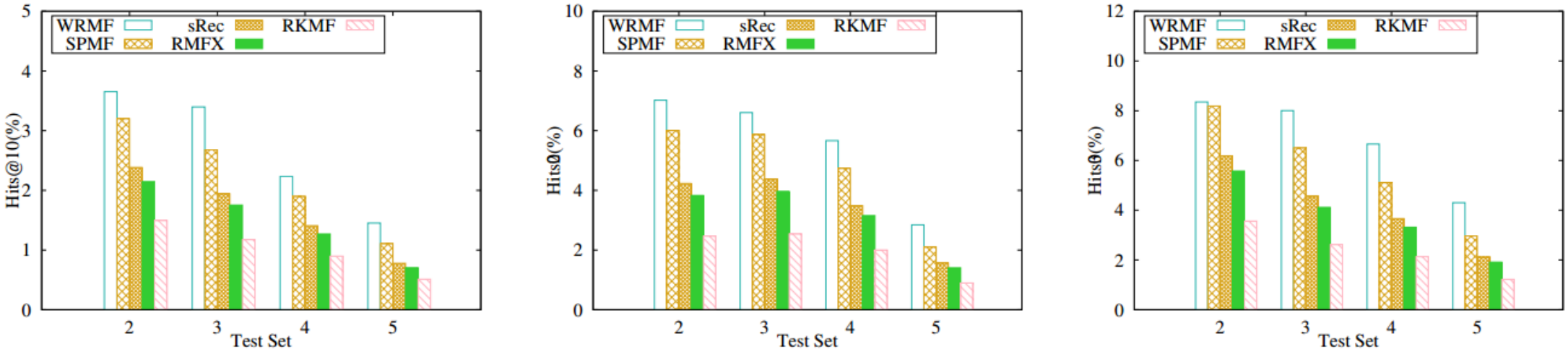
- Focus on drifted interests



Chang S, Zhang Y, Tang J, et al. Streaming recommender systems[C]//Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2017: 381-389.

Experiments(from SPMF)

- Evaluation metric: Hit ratio, don't consider rank before k comparing to NDCG



(a) *Hits*@10 (b) *Hits*@20 (c) *Hits*@30

Figure 2: Performance on Movielens

Overview

- Online recommendation(1+3)
  - Fast online updating
  - Capture both long-term and drifted interests
  - Modeling new users or items
- Recommendation with Social Networks(2+1)
  - Group representation, community detection, sequence-aware Rec
- Improve traditional methods(3)
  - APR, CMN, Bandit problem

Recommendation with Social Networks

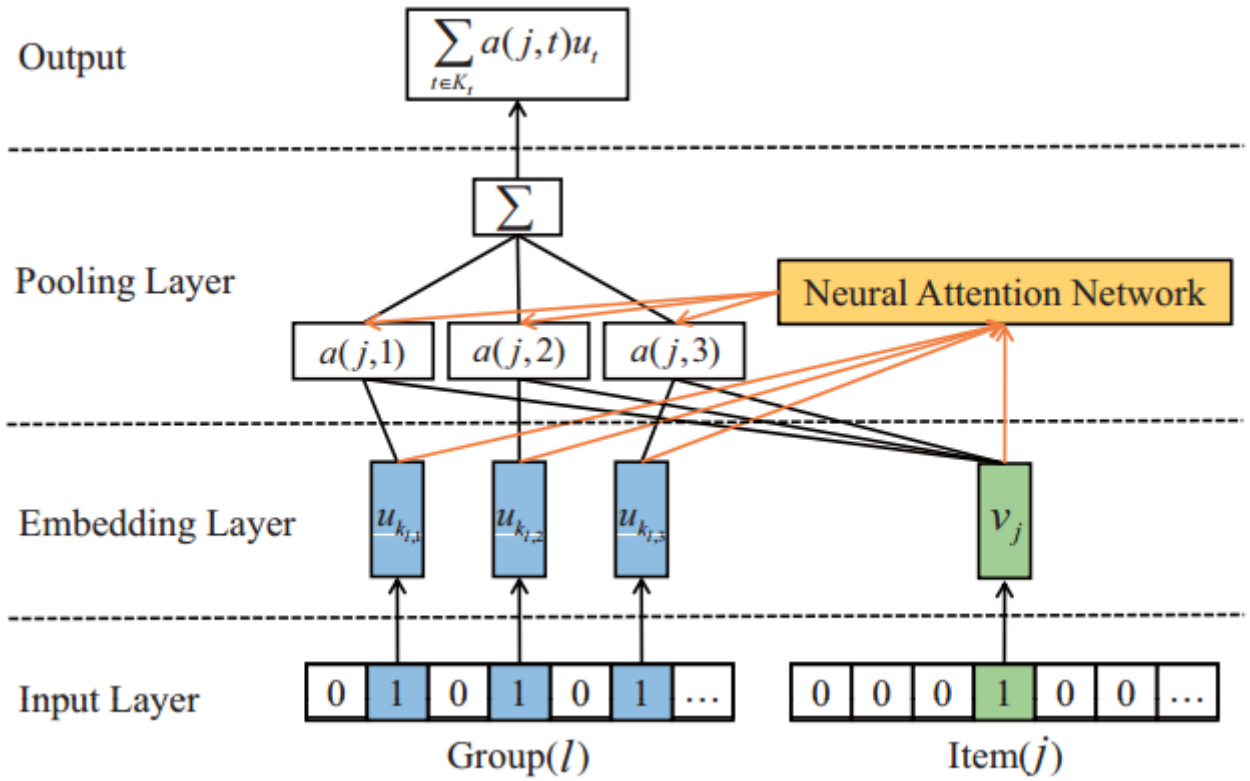
- Group representation
  - Attention based on specific user and item
- Community detection
  - Cross-domain clustering
- Sequence-aware recommendation
  - Dynamic attention over time t base on social information

Attentive Group Recommendation

- Key problem: how to aggregate the preferences of group members to infer the decision of a group
  - Naïve method: average aggregation

- Example and intuition
  - When you want to see a movie with your
    - friends ? Maybe average aggregation
    - boyfriend or girlfriend ?
    - family ?

Attentive group representation

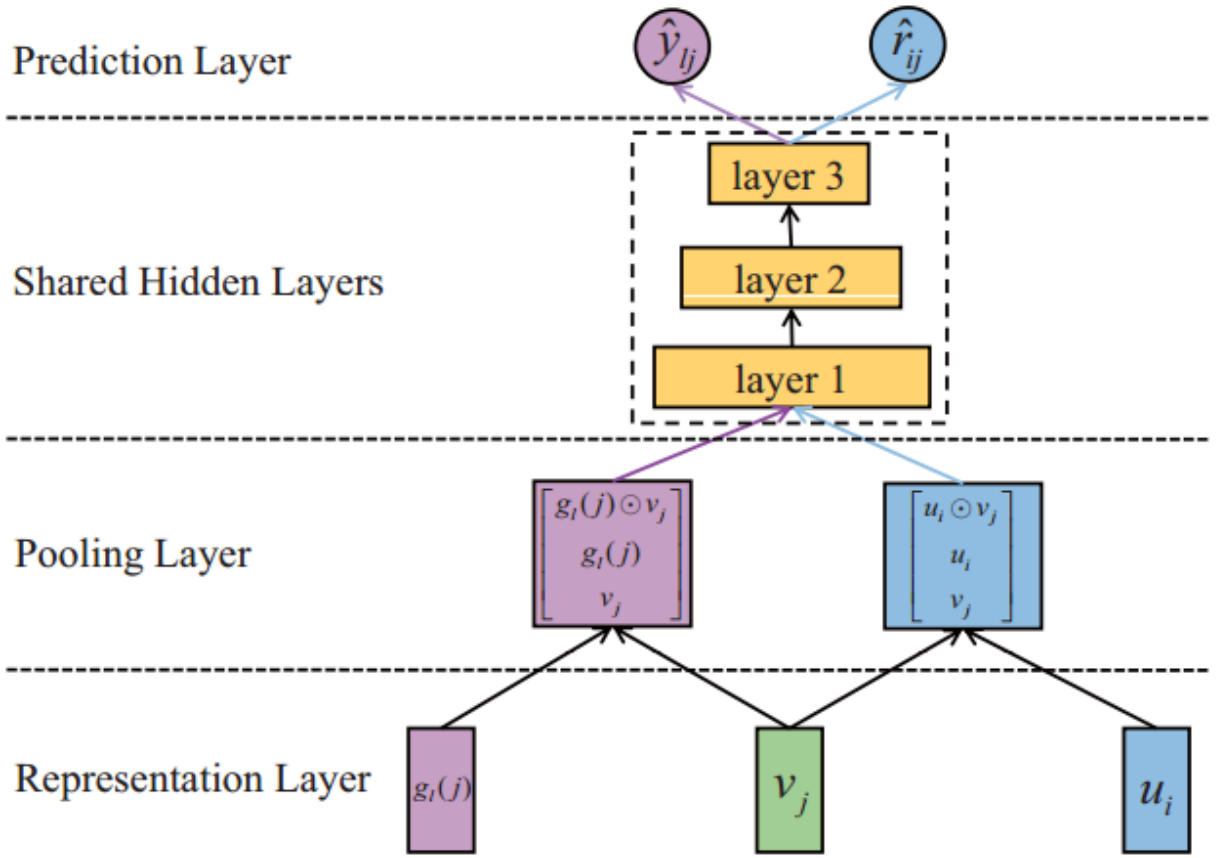


$$o(j,t) = \mathbf{h}^T \mathrm{ReLU}(\mathbf{P}_v \mathbf{v}_j + \mathbf{P}_u \mathbf{u}_t + \mathbf{b}),$$

$$\alpha(j,t) = \mathrm{softmax}(o(j,t)) = \frac{\exp o(j,t)}{\sum_{t' \in \mathcal{K}_l} \exp o(j,t')},$$

$$\mathbf{g}_l(j) = \underbrace{\sum_{t \in \mathcal{K}_l} \alpha(j,t)\mathbf{u}_t}_{\text{user embedding aggregation}} + \underbrace{\mathbf{q}_l}_{\text{group preference embedding}}.$$

Interaction learning based on NCF



$$\begin{cases} \mathbf{e}_1 = \mathrm{ReLU}(\mathbf{W}_1\mathbf{e}_0 + \mathbf{b}_1) \\ \mathbf{e}_2 = \mathrm{ReLU}(\mathbf{W}_2\mathbf{e}_1 + \mathbf{b}_2) \\ \ldots\ldots \\ \mathbf{e}_h = \mathrm{ReLU}(\mathbf{W}_h\mathbf{e}_{h-1} + \mathbf{b}_h) \end{cases},$$

$$\begin{cases} \hat{r}_{ij} = \mathbf{w}^T\mathbf{e}_h, \ if \ \mathbf{e}_0 = \varphi_{pooling}(\mathbf{u}_i, \mathbf{v}_j) \\ \hat{y}_{lj} = \mathbf{w}^T\mathbf{e}_h, \ if \ \mathbf{e}_0 = \varphi_{pooling}(\mathbf{g}_l(j), \mathbf{v}_j) \end{cases},$$

Model optimization

- BPR loss: $-\log \sigma(\hat{r}_{ij} - \hat{r}_{is})$
- To decrease the BPR loss on a **multi-layer model**, a trivial solution is to **scale up** the weights in each update. As such, it is crucial to enforce the L2 regularization on the weights to avoid this trivial solution.
- Pairwise square loss(without regularization)

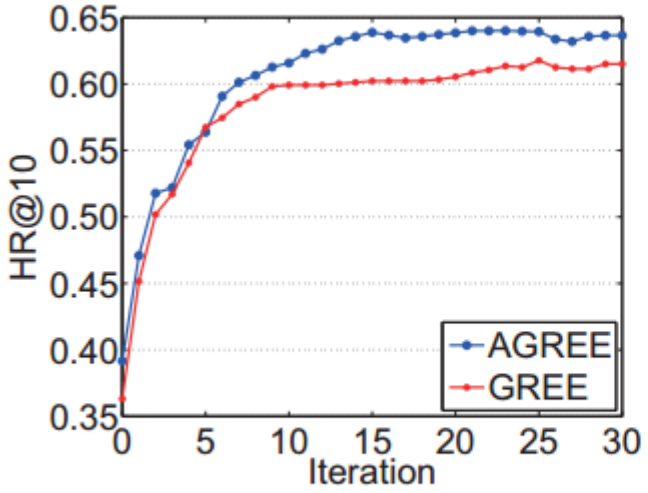
$$\mathcal{L}_{user} = \sum_{(i,j,s)\in O} (r_{ijs} - \hat{r}_{ijs})^2 = \sum_{(i,j,s)\in O} (\hat{r}_{ij} - \hat{r}_{is} - 1)^2,$$

$$\mathcal{L}_{group} = \sum_{(l,j,s)\in O'} (y_{ljs} - \hat{y}_{ljs})^2 = \sum_{(l,j,s)\in O'} (\hat{y}_{lj} - \hat{y}_{ls} - 1)^2,$$

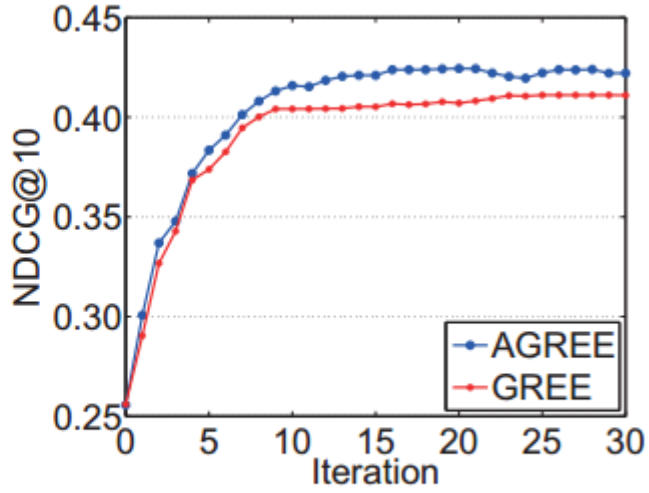
Experiments

Table 1: Case studies of a sampled group on the effect of attention (Section 3.2). The member weights and prediction scores of the group for positive venues (Venue #30, #32, #106) and negative venues (Venue #65, #121, #123) are shown (Section 3.2).

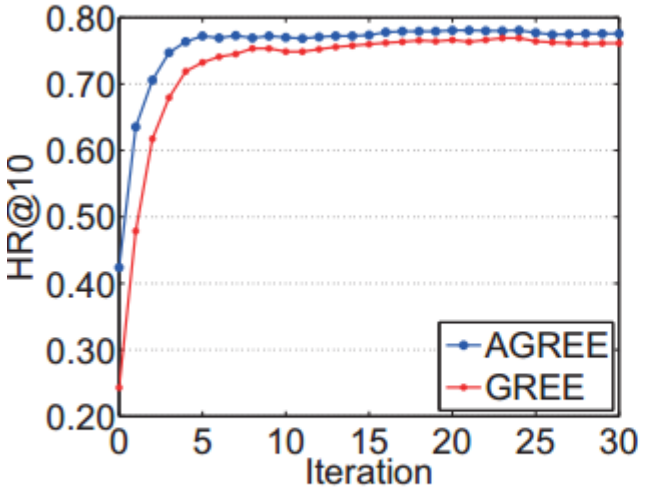
	Model	User #805	User #806	User #807	$\hat{y}$
Venue #30	GREE	0.333	0.333	0.333	0.260
	AGREE	0.286	0.302	**0.412**	**0.572**
Venue #32	GREE	0.333	0.333	0.333	0.096
	AGREE	0.222	**0.583**	0.195	**0.370**
Venue #106	GREE	0.333	0.333	0.333	0.192
	AGREE	**0.364**	0.287	0.347	**0.318**
Venue #65	GREE	0.333	0.333	0.333	0.132
	AGREE	**0.408**	0.311	0.281	**0.091**
Venue #121	GREE	0.333	0.333	0.333	0.132
	AGREE	0.335	**0.374**	0.291	**0.053**
Venue #123	GREE	0.333	0.333	0.333	0.109
	AGREE	0.288	**0.411**	0.301	**0.063**



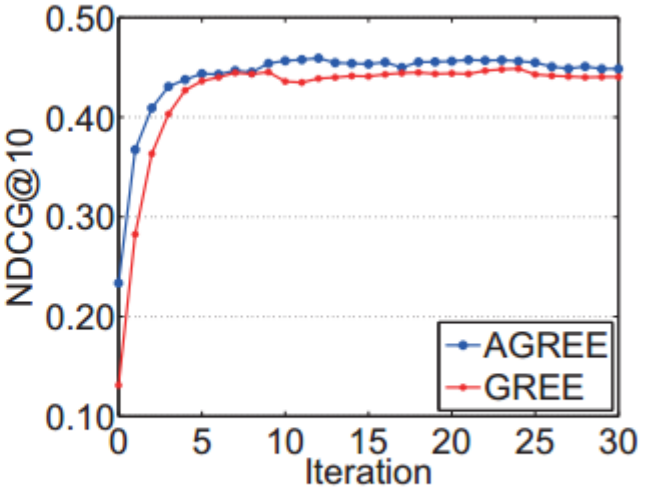
(a) Mafengwo — HR@10



(b) Mafengwo — NDCG@10



(c) CAMRa2011 — HR@10



(d) CAMRa2011 — NDCG@10

Attentive Recurrent Social Recommendation

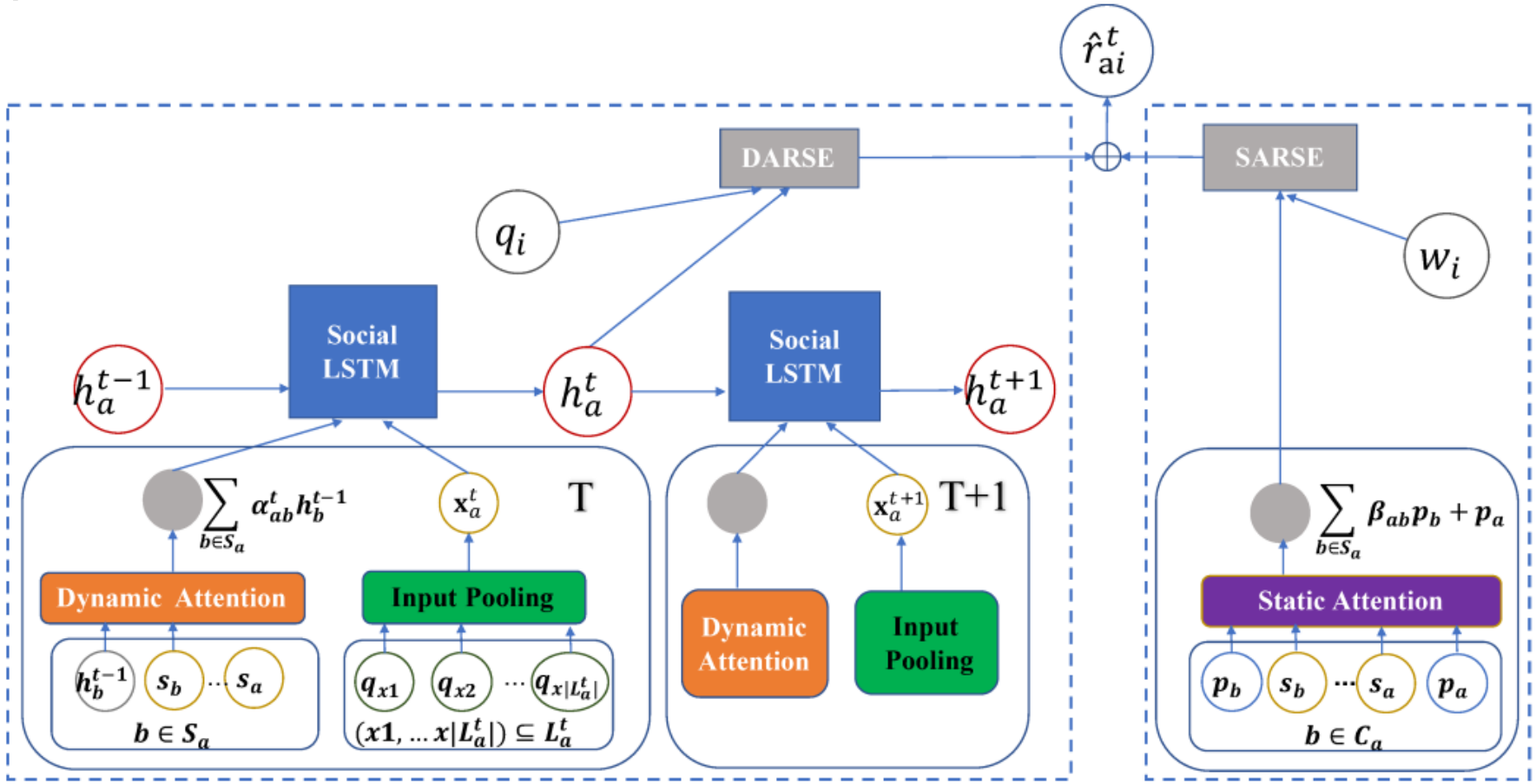
- Key problem
  - The complex interplay between users' internal interests and the social influence from the social network drives the evolution of users' preferences over time
  - Traditional approaches either neglected the social network structure for temporal recommendation or **assumed a static social influence strength** for static social recommendation

Sun P, Wu L, Wang M. Attentive recurrent social recommendation[C]//The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. ACM, 2018: 185-194.

Notations

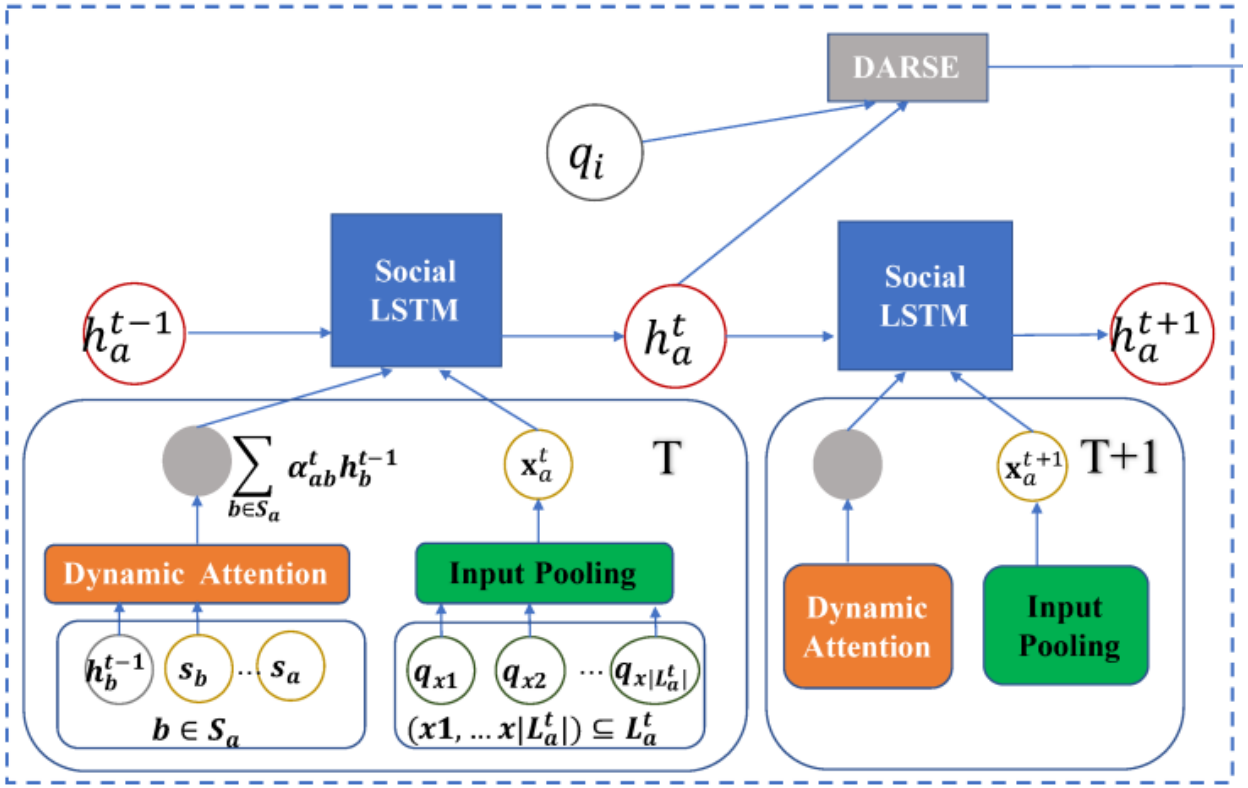
Notations	Description
U	Userset, $\|U\| = M$
V	Itemset, $\|V\| = N$
a,b,c,u	User
i,j,k,v	Item
$R^t \in \mathbb{R}^{M \times N}$	Rating matrix at time t
$S \in \mathbb{R}^{M \times M}$	Social network matrix, with s_{ba} denotes whether a follows b
$L_a^t \in V$	The item list that a likes at time t, $L_a^t = [i : \|r_{ai}^t = 1]$
$Q \in \mathbb{R}^{D \times N}$	Item latent matrix in the dynamic latent space
$W \in \mathbb{R}^{D \times N}$	Item latent matrix in the static latent space
$P \in \mathbb{R}^{D \times N}$	User base latent matrix in the static latent space
q_i	The dynamic embedding of item i in the dynamic latent space
w_i	The static embedding of item i in the static latent space
p_a	The static embedding of user a in the static latent space
$\mathbf{x}_a^t$	The input vector of user a at time t
h_a^t	The dynamic latent vector of a at time t
α_{ab}^t	The dynamic influence strength of b to a at time t
β_{ab}	The static influence strength of b to a

Proposed model



$$\hat{r}_{ai}^{t} = \hat{r}_{D,ai}^{t} + \hat{r}_{S,ai} = q_i' \times h_a^t + w_i' \times \tilde{p}_a,$$

Dynamic part



$$\mathbf{x}_a^t = Pooling(Q(:, L_a^t))$$

$$m^t(a, b) = ReLU(A_5 \times ReLU(A_1 \times h_a^{t-1} + A_2 \times h_b^{t-1} + A_3 \times e_a + A_4 \times e_b))$$

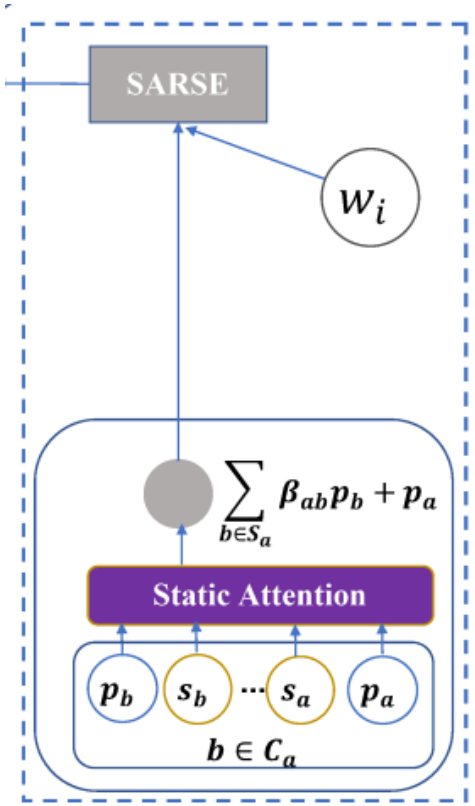
$$\alpha_{ab}^t = \frac{exp(m^t(a, b))}{\sum_{c \in S_a} exp(m^t(a, c))}.$$

$$\widetilde{h}_a^t = \sum_{b \in S_a} \alpha_{ab}^t \times h_b^t$$

$$h_a^t = f_{LSTM}([\mathbf{x}_a^t, h_a^{t-1}, \widetilde{h}_a^{t-1}]),$$

$$\hat{r}_{D,ai}^t = q_i' \times h_a^t.$$

Static part



$$n(a, b) = ReLU(B_5 \times ReLU(B_1 \times p_a + B_2 \times p_b + B_3 \times e_a + B_4 \times e_b)).$$

$$\beta_{ab} = \frac{exp(n(a, b))}{\sum_{c \in S_a} exp(n(a, c))}.$$

$$\hat{r}^t_{S,ai} = w'_i \times \tilde{p}_a.$$

Model optimization

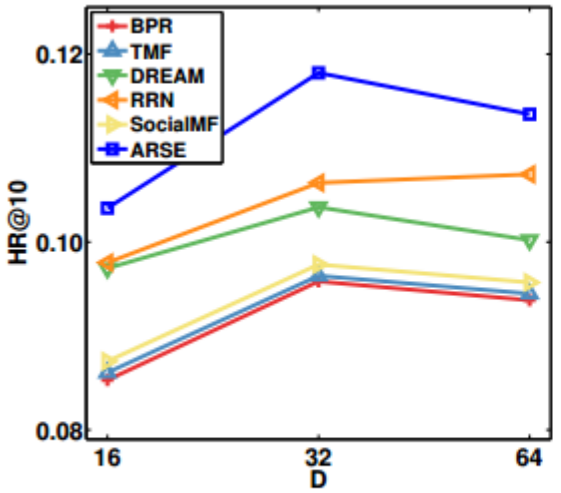
- Focus on the implicit feedback of users, we adopt the widely used log loss function

$$L_{\Theta}(\mathbf{R}, \hat{\mathbf{R}}) = -\sum_{t=1}^{T}\sum_{a=1}^{M}\sum_{i=1}^{N}[r_{ai}^{t} log(\hat{r}_{ai}^{t}) + (1 - r_{ai}^{t}) log(1 - \hat{r}_{ai}^{t})].$$

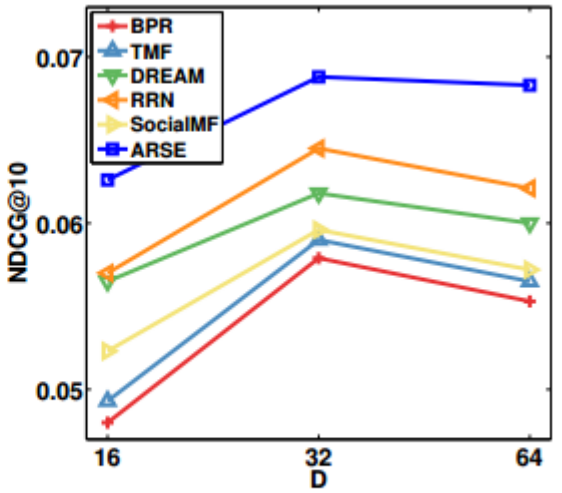
Experiments

Table 2: The statistics of the two datasets.

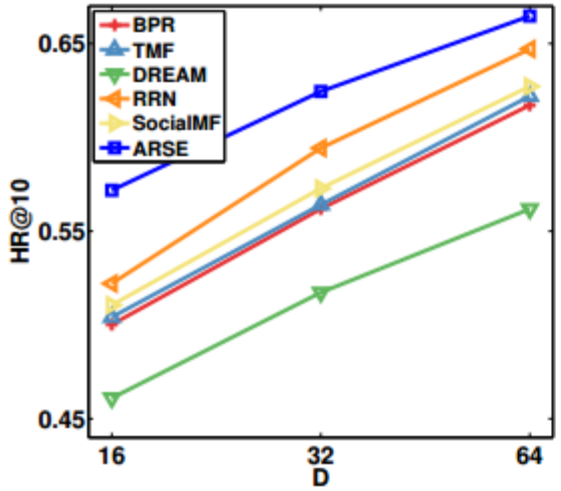
Dataset	Epinions	Gowalla
Users	4,630	21,755
Items	26,991	71,139
Time Windows	12	4
Total Links	78,356	257,550
Training Ratings	62,872	278,154
Test Ratings	2,811	52,448
Link Density	0.35%	0.053%
Rating Density	0.050%	0.018%



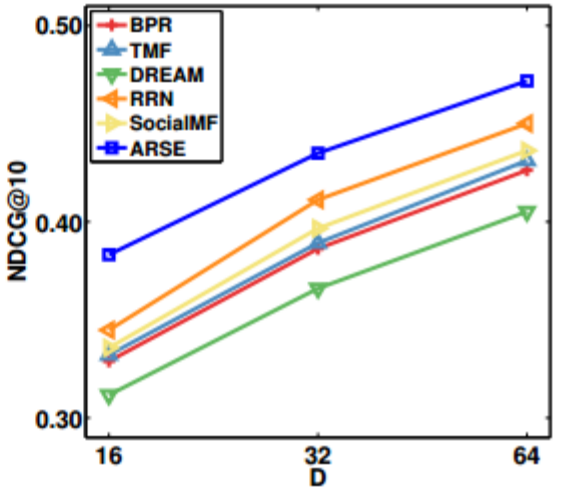
(a) Epinions HR@10



(b) Epinions NDCG@10



(c) Gowalla HR@10



(d) Gowalla NDCG@10

Cross-Domain Recommendation via Clustering on Multi-Layer Graphs

- Key problem
  - social connections between users (i.e. follower/followee relationship) that are often hidden behind the privacy settings
  - So, to avoid privacy concerns, we propose to use user generated data on multiple platforms to **detect the similarities between users**.

Farseev A, Samborskii I, Filchenkov A, et al. Cross-domain recommendation via clustering on multi-layer graphs[C]//Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2017: 195-204.

Cross domain community detection

- Similarity graph construction
- Community detection on one domain
  - Ncut problem and Spectral clustering
- Cross domain community detection
  - Spectral clustering on multi-layer graph
- Incorporating inter-layer relationship

Notations

Table 1: Notations summary

Symb.	Description
vec_u	Distribution of the user u among items (venue categories) in u's past
C_u	Community of the user u
γ	Parameter that controls personal aspect of rec-n
θ	Parameter that controls group aspect of rec-n
N	Number of users
M	Number of data sources (graph layers)
L_i	Laplacian matrix of the i-th layer
U_i	Eigendecomposition matrix of the i-th layer
$\hat{L}_i$	Inter-layer relationship regularized Laplacian of the i-th layer
$\hat{U}_i$	Inter-layer relationship regularized eigendecomposition matrix of the i-th layer
$\hat{L}_{mod}$	Sub-space regularized Laplacian matrix
W_R	Adjacency matrix of inter-layer similarity graph
k	Parameter that controls the number of clusters
α	Parameter that controls sub-space regularization
β_i	Parameter that controls inter-layer regularization for the layer i

Similarity graph construction

- For every graph node pair (i, j) from the m-th graph layer, the corresponding distance is:

$$d_{m_{i,j}} = e^{-\frac{||x_{m_i} - x_{m_j}||^2}{\sigma}},$$

- A graph layer is built from the user generated data from the same platform

Community detection on one domain(a graph layer)

- NCut problem: divide the nodes to communities that are formed by users that are most similar to each problem
- NCut problem is NP-hard.
- Approximation by spectral clustering

$$\min_{U \in R^{n \times k}} \operatorname{tr}(U^{\top} L_{sym} U), \; s.t. \; U^{\top} U = I.$$

  - the solution of the problem is given by the first *k* eigenvectors of the normalized graph Laplacian $L_{sym} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$
  - W is adjacency matrix, and D is degree matrix
- Clustering over the feature space U

Spectral clustering on multi-layer graph

- The final data representation (latent representation) must be consistent with all graph layers
- Measure closeness of two latent space

$$d^2_{Proj}(S_1, S_2) = \frac{1}{2}||S_1 S_1^\intercal - S_2 S_2^\intercal||^2_F,$$

- Measure closeness between target space S and other domain specific space $\{S_i\}$

$$d^2_{Proj}(S, \{S_i\}_{i=1}^M) = kM - \sum_{i=1}^{M} \mathrm{tr}(SS^\intercal S_i S_i^\intercal).$$

Spectral clustering on multi-layer graph

- Optimization target

$$\min_{U \in \mathbb{R}^{n \times k}} \sum_{i=1}^{M} \mathrm{tr}(U^\top L_i U) + \alpha(kM - \sum_{i=1}^{M} \mathrm{tr}(U U^\top U_i U_i^\top))$$

$$= \min_{U \in \mathbb{R}^{n \times k}} \mathrm{tr}(U^\top \sum_{i=1}^{M} (L_i - \alpha U_i U_i^\top) U),$$

- The same as one graph layer, the solution of the problem is given

by the first k eigenvectors of $\sum_{i=1}^{M} (L_i - \alpha U_i U_i^\top)$

Incorporating inter-layer relationship

- Real world problems often require a consideration of inter-layer relationship (similarity)
- New optimization target for the i-th layer

$$\min_{\hat{U}_i \in \mathbb{R}^{n \times k}} \operatorname{tr}(\hat{U}_i^\top L_i \hat{U}_i) + \beta_i (kM - \sum_{j=1, j \neq i}^{M} w_{i,j} \operatorname{tr}(\hat{U}_i \hat{U}_i^\top U_j U_j^\top))$$

$$= \min_{\hat{U}_i \in \mathbb{R}^{n \times k}} \operatorname{tr}(\hat{U}_i^\top (L_i - \beta_i \sum_{j=1, j \neq i}^{M} w_{i,j} U_j U_j^\top) \hat{U}_i),$$

- New regularized Laplacian Matrix

$$\hat{L}_i := L_i - \beta_i \sum_{j=1, j \neq i}^{M} w_{i,j} U_j U_j^\top .$$

Incorporating inter-layer relationship

- Final optimization target

$$\min_{U \in \mathbb{R}^{n \times k}} \sum_{i=1}^{M} \mathrm{tr}(U^\top \hat{L}_i U) + \alpha(kM - \sum_{i=1}^{M} \mathrm{tr}(UU^\top \hat{U}_i \hat{U}_i^\top))$$

$$= \min_{U \in \mathbb{R}^{n \times k}} \mathrm{tr}(U^\top \sum_{i=1}^{M} (\hat{L}_i - \alpha \hat{U}_i \hat{U}_i^\top) U),$$

$$s.t.\ U^\top U = I.$$

Computing Inter-Layer Relationship

- Define N × N k-clustering co-occurrence matrices $M_{q,k}$, $M_{w,k}$, in which each value $m_{i,j}$ is equal to 1 if user i is assigned to the same cluster as user j in both layers w and q, and 0 otherwise

- Similarity between layer q and w

$$sim(w, q) = \left(\sum_{k=2}^{K} \left(1 - \frac{||M_{w,k} - M_{q,k}||}{\sqrt{N(N-1)}} \right) \right) / (K-1).$$

Algorithm

Algorithm 1 C^3R clustering

function CLUSTER($\{W_i\}_{i=1}^M$, W_R, k, α, $\{\beta_i\}_{i=1}^M$)
▷ $\{W_i\}_{i=1}^M$ are weighted adjacency matrices of layers $\{G_i\}_{i=1}^M$,
W_R is adjacency matrix of inter-layer similarity graph,
k is target number of clusters,
α and $\{\beta_i\}_{i=1}^M$ are regularization parameters

for $i \leftarrow [0; M-1]$ **do**
Compute L_i and U_i for G_i [50]
▷ L_i is the normalized Laplacian matrix of the layer i,
U_i is subspace representation of the layer i,
G_i is ith layer graph
Compute $\hat{L}_i \leftarrow L_i - \beta_i \sum_{j=1, j\neq i}^{M} w_{i,j} U_j U_j^\intercal$
▷ $\hat{L}_i$ is the regularized Laplacian matrix of ith layer
Compute $\hat{U}_i \in \mathbb{R}^{N\times k}$
▷ $\hat{U}_i$ is the the matrix of first k eigenvectors of $\hat{L}_i$[28]
end for
Compute $\hat{L}_{mod} \leftarrow \sum_{i=1}^{M}(\hat{L}_i - \alpha \hat{U}_i \hat{U}_i^\intercal)$
▷ $\hat{L}_{mod}$ is the modified Laplacian matrix [12]
Compute $U \in \mathbb{R}^{N\times k}$
▷ U is the matrix of first k eigenvectors [28] of $\hat{L}_{mod}$
Normalize rows of U to get U_{norm}
$\{C\}_{i=1}^k \leftarrow$ FINALCLUSTERING(U_{norm})
▷ FINALCLUSTERING() is k-means or x-means clustering

return $\{C\}_{i=1}^k$
▷ $C_1, \ldots, C_k$ are cluster assignment

end function

Overview

- Online recommendation(1+3)
  - Fast online updating
  - Capture both long-term and drifted interests
  - Modeling new users or items
- Recommendation with Social Networks(2+1)
  - Group representation
  - Sequence-aware Recommendation
  - Community detection avoiding privacy concerns
- Improve traditional methods(3)
  - APR, CMN, Bandit problem

Adversarial Personalized Ranking for Recommendation

- Optimizing MF with BPR leads to a recommender model that is **not robust**
- The resultant model is highly vulnerable to adversarial perturbations on its model parameters, which **implies the possibly large error in generalization**
- To enhance the **robustness** of a recommender model and thus improve its **generalization performance**

He X, He Z, Du X, et al. Adversarial personalized ranking for recommendation[C]//The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. ACM, 2018: 355-364.

Adversarial Noises

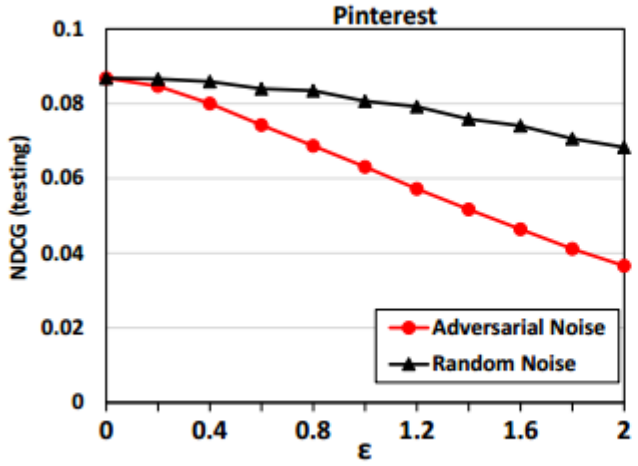
- Defined as the perturbations that aim to maximize the objective function of BPR

$$\Delta_{adv} = \arg \max_{\Delta, ||\Delta|| \leq \epsilon} L_{BPR}(\mathcal{D}|\hat{\Theta} + \Delta),$$

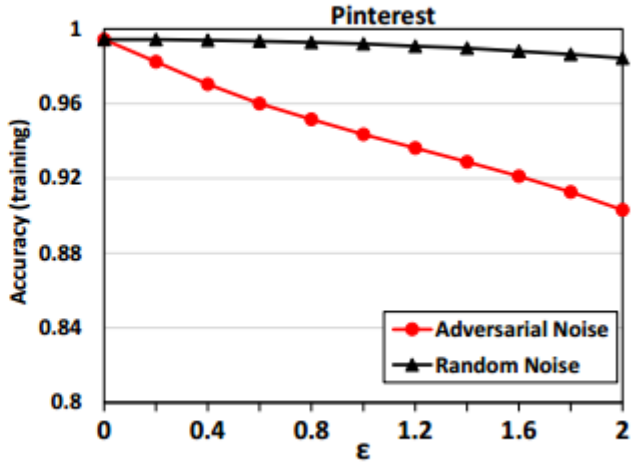
- Approximation

$$\Delta_{adv} = \epsilon \frac{\Gamma}{||\Gamma||} \quad \text{where} \quad \Gamma = \frac{\partial L_{BPR}(\mathcal{D}|\hat{\Theta} + \Delta)}{\partial \Delta}.$$

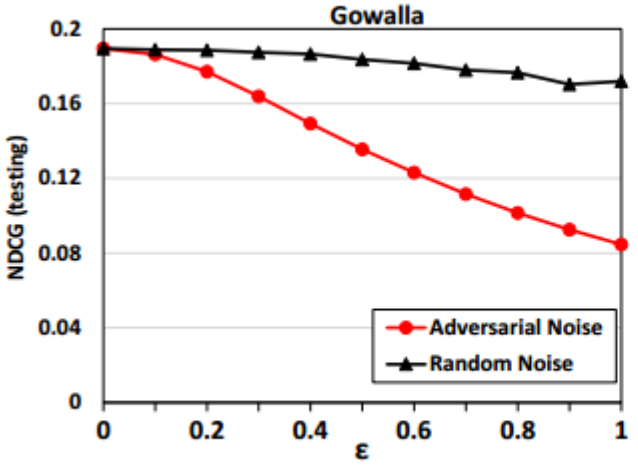
BPR-MF is Vulnerable



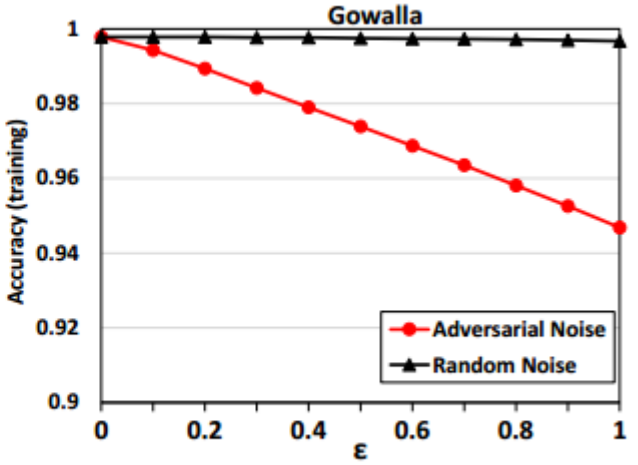
(a) Testing NDCG vs. ϵ



(b) Training Accuracy vs. ϵ

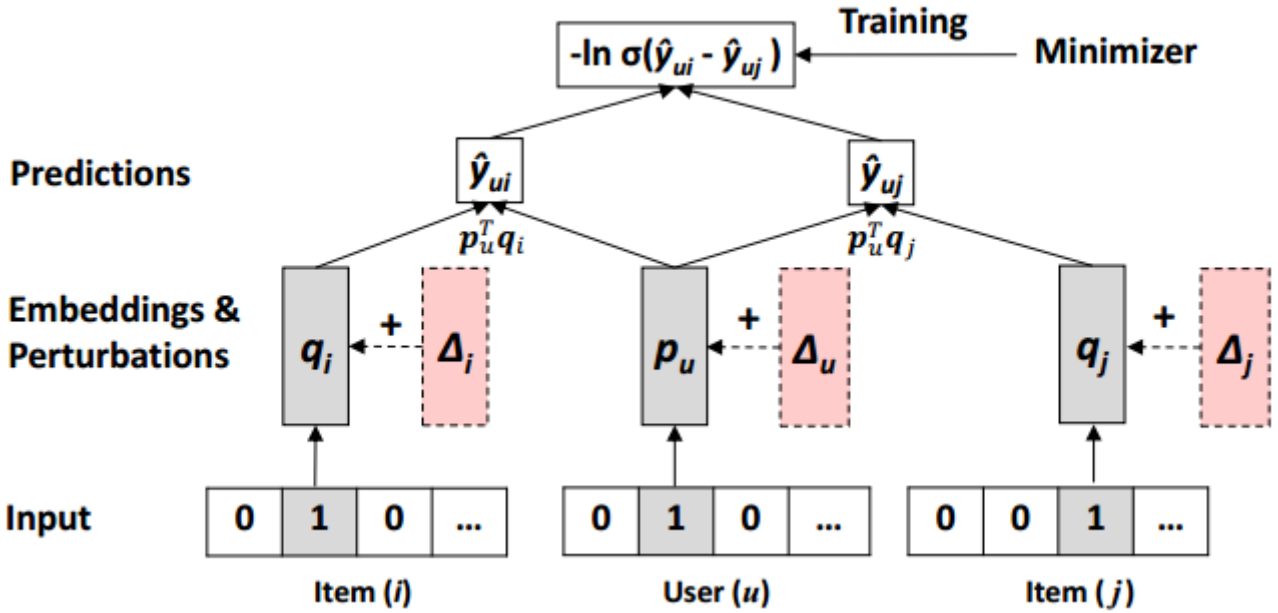


(c) Testing NDCG vs. ϵ



(d) Training Accuracy vs. ϵ

Adversarial Personalized Ranking



$$L_{APR}(\mathcal{D}|\Theta) = L_{BPR}(\mathcal{D}|\Theta) + \lambda L_{BPR}(\mathcal{D}|\Theta + \Delta_{adv}),$$

$$\text{where} \quad \Delta_{adv} = \arg \max_{\Delta, ||\Delta|| \leq \epsilon} L_{BPR}(\mathcal{D}|\hat{\Theta} + \Delta),$$

$$\Theta^*, \Delta^* = \arg \min_{\Theta} \max_{\Delta, ||\Delta|| \leq \epsilon} L_{BPR}(\mathcal{D}|\Theta) + \lambda L_{BPR}(\mathcal{D}|\Theta + \Delta),$$

SGD learning algorithm for APR

Algorithm 1: SGD learning algorithm for APR.

Input: Training data $\mathcal{D}$, adversarial noise level ϵ, adversarial regularizer λ, L_2 regularizer λ_Θ, learning rate η;
Output: Model parameters Θ;

```
Initialize Θ from BPR ;
while Stopping criteria is not met do
    Randomly draw (u, i, j) from D ;
    // Constructing adversarial perturbations
    Δ_adv ← Equation (8) ;
    // Updating model parameters
    Θ ← Equation (11) ;
end
return Θ
```

$$l_{adv}((u,i,j)|\Delta) = -\lambda \ln \sigma(\hat{y}_{ui}(\hat{\Theta}+\Delta) - \hat{y}_{uj}(\hat{\Theta}+\Delta)).$$

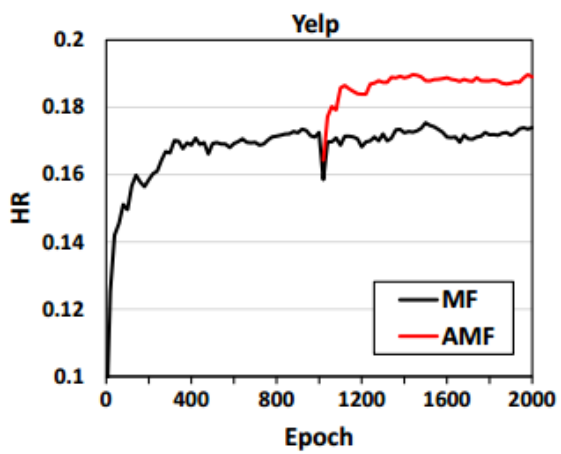
$$\Delta_{adv} = \epsilon \frac{\Gamma}{||\Gamma||} \quad \text{where} \quad \Gamma = \frac{\partial l_{adv}((u,i,j)|\Delta)}{\partial \Delta}. \tag{8}$$

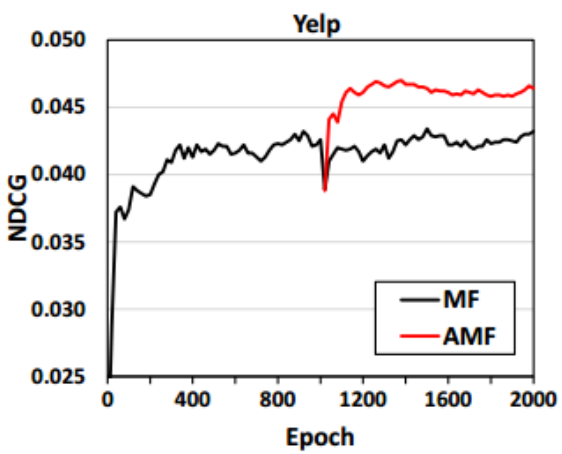
$$l_{APR}((u,i,j)|\Theta) = -\ln \sigma(\hat{y}_{ui}(\Theta) - \hat{y}_{uj}(\Theta)) + \lambda_\Theta ||\Theta||^2 - \lambda \ln \sigma(\hat{y}_{ui}(\Theta+\Delta_{adv}) - \hat{y}_{uj}(\Theta+\Delta_{adv})).$$

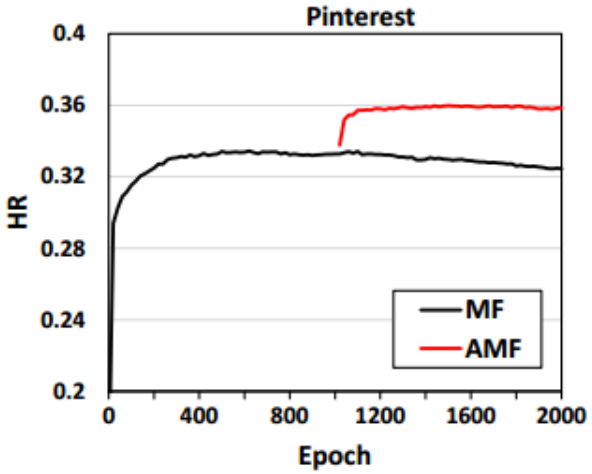
$$\Theta = \Theta - \eta \frac{\partial l_{APR}((u,i,j)|\Theta)}{\partial \Theta}, \tag{11}$$

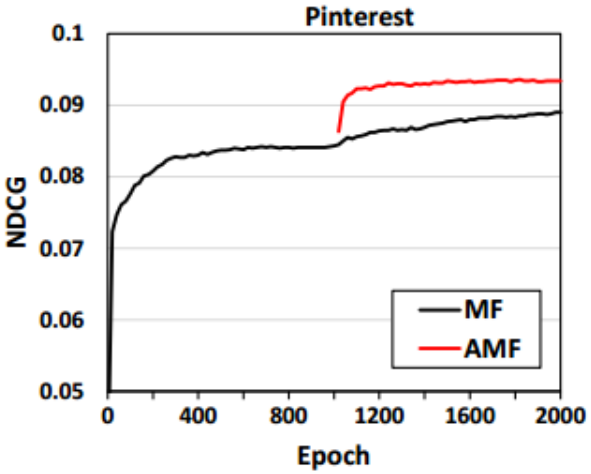
Experiments

RQ1 How is the effect of adversarial learning? Can AMF improve over MF-BPR by performing adversarial learning?









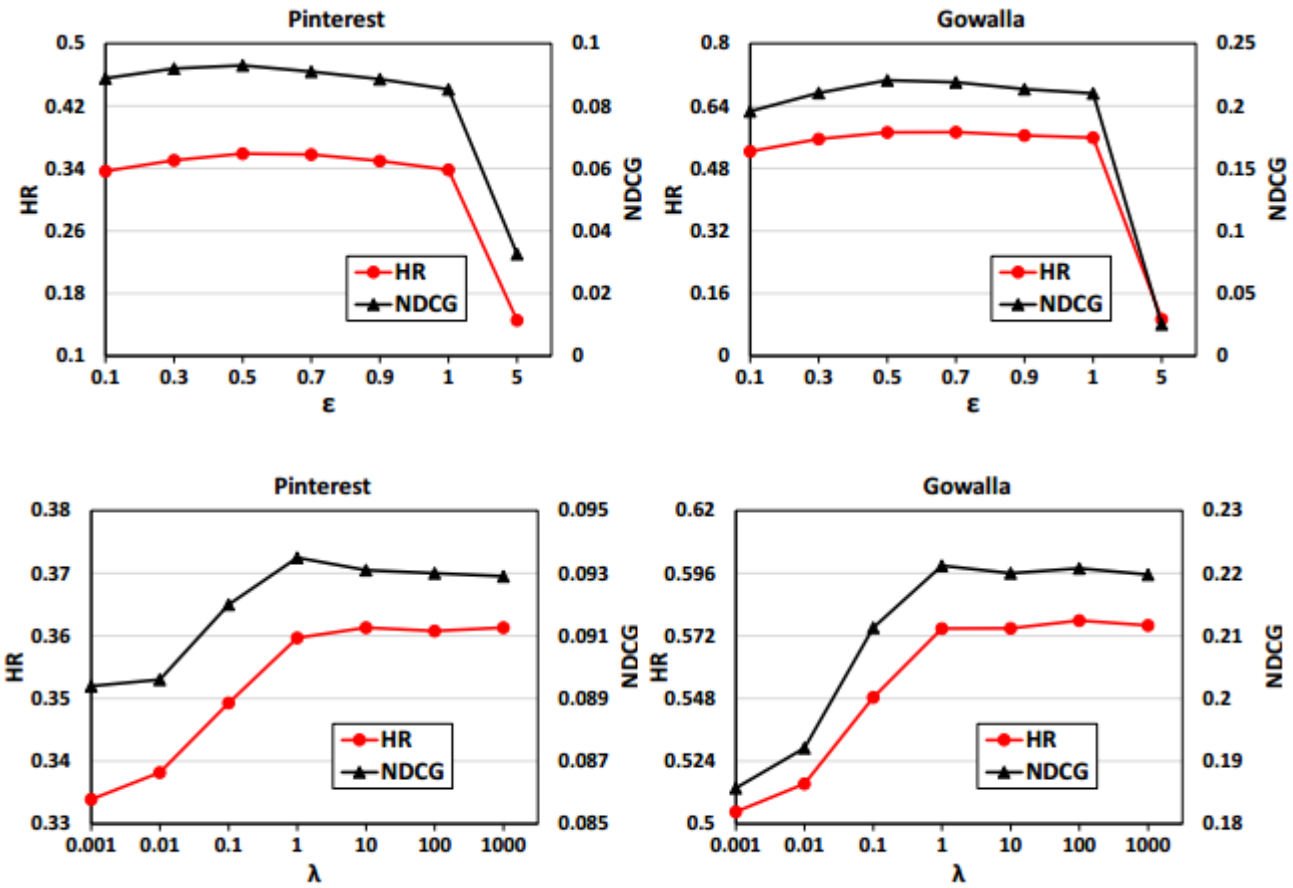
Experiments

RQ2 How does AMF perform compared with state-of-the-art item recommendation methods?

	Yelp, HR		Yelp, NDCG		Pinterest, HR		Pinterest, NDCG		Gowalla, HR		Gowalla, NDCG		RI
	K=50	**K=100**	**K=50**	**K=100**	**K=50**	**K=100**	**K=50**	**K=100**	**K=50**	**K=100**	**K=50**	**K=100**	
ItemPop	0.0405	0.0742	0.0114	0.0169	0.0294	0.0485	0.0085	0.0116	0.1183	0.1560	0.0367	0.0428	+416%
MF-BPR	0.1053	0.1721	0.0312	0.0420	0.2226	0.3403	0.0696	0.0886	0.4061	0.5072	0.1714	0.1878	+11.2%
CDAE [35]	0.1041	0.1733	0.0293	0.0405	0.2254	0.3495	0.0672	0.0873	0.4435	0.5483	0.1837	0.2007	+9.5%
IRGAN [31]	0.1119	0.1765	**0.0361***	**0.0465***	0.2254	0.3363	0.0724	0.0904	0.4157	0.518	0.1853	0.2019	+5.9%
NeuMF [17]	0.1135	0.1817	0.0335	0.0445	0.2342	0.3526	0.0734	0.0925	0.4558	0.5642	0.1962	0.2138	+2.9%
AMF	**0.1176***	**0.1885***	0.0350	**0.0465***	**0.2375***	**0.3595***	**0.0741***	**0.0938***	**0.4693***	**0.5763***	**0.2039***	**0.2212***	-

Experiments

RQ3 How do the hyper-parameters ϵ and λ affect the performance and how to choose optimal values?



Conclusion and Others(maybe in next pre)

- Online recommendation(1+3)
- Recommendation with Social Networks(2+1)
  - Group representation, community detection, sequence-aware Rec
- Improve traditional methods(3)
  - APR, CMN, Bandit problem
- Recommendation with Knowledge Base(1)
- Some specific tasks(5)
  - Recommend email, mention, citation, Wikipedia article section
  - conversational recommender system
- User modeling: Geo-social based(1)

Thank you