SIGIR 2021

LSTPR: Graph-based Matrix Factorization with Long Short-term Preference Ranking

Chih-Hen Lee[1], Jun-En Ding[1], Chih-Ming Chen[2], Jing-Kai Lou[3], Ming-Feng Tsai[2], Chuan-Ju Wang[1]

2021.06.03

Presented by Chih-Hen Lee

Academia Sinica[1]

National Chengchi University[2]

KKStream Limited[3]

OUTLINE

ONE

Motivation

TWO

Analysis

THREE

Method

FOUR

Conclusion

Motivation

Motivation

- We want to leverage two important features, temporal information and high-order graph relations, at the same time for recommendation task.
- However, the sequential models and GCN-based recommenders could be time-consuming or suffer from data sparsity issue.
- Therefore, we propose LSTPR, which explicitly distinguishes long-term and short-term user preferences and enriches the sparse interactions via random surfing on the user-item graph.

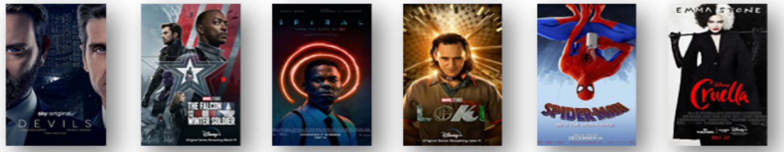
Analysis

Analysis

- We aim to answer the following questions with quantitative analyses:
  - Do short-term preferences really exist?
  - If so, can short-term preferences help predict user's future behaviors?

Analysis - 1

- Firstly, we represent items by the column vectors on the user-item matrix.
- Then, we sort each user's behavior logs in the training period with temporal order.

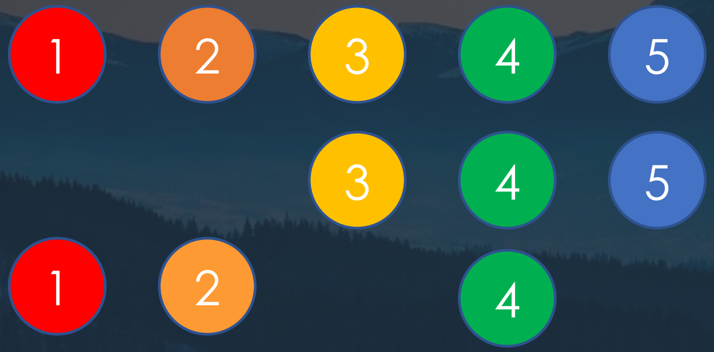


0	1	1	0	1	0
1	1	1	0	1	1
1	0	0	0	1	0
0	1	1	1	0	1
0	1	0	1	1	0

Ref: https://www.imdb.com/whats-on-tv/?ref_=nv_tv_ontv

Analysis - 1

- There are three kinds of preferences for every user, which are represented as normalized sum of item vectors:
  - Long-term: all interacted items
  - Short-term: last n items
  - Random: random n items



- From n=1 to n=15, we calculate every user's Kullback-Leibler Divergence (KLD) between Short-term preference and Long-term preference and Random preference and Long-term preference.

Analysis - 1

- Calculate the average of all user's KLDs
- Plot Avg. KLDs vs. n

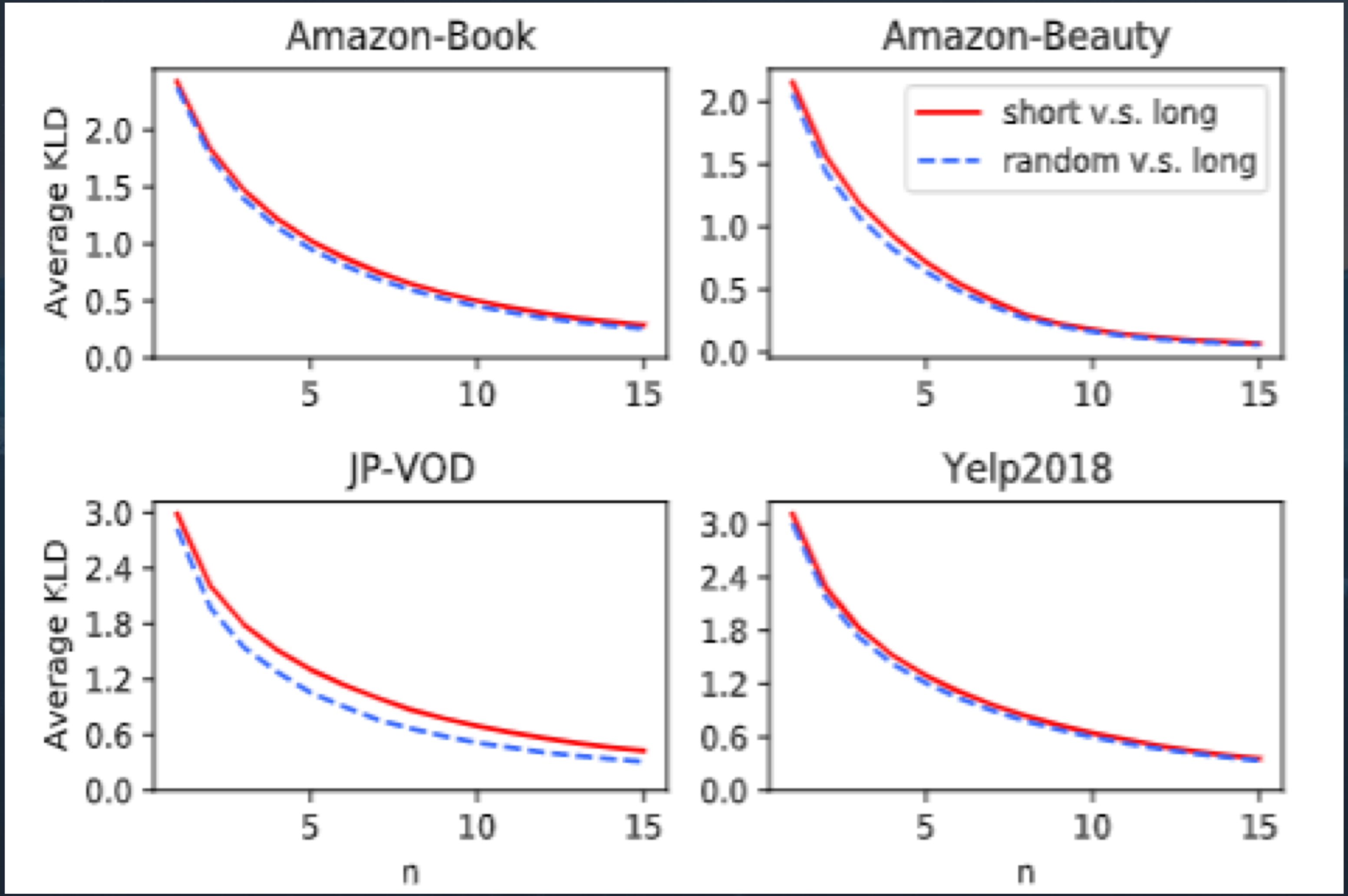
KLDs between short-term and long-term preferences

	n = 1	n = 2	...	n = 14	n = 15
	2.8	2.5	...	1.5	1.4
	1.2	1.0	...	0.6	0.4
	3.2	2.8	...	0.0	0.0
Avg.	2.4	2.1	...	0.7	0.6

KLDs between random and long-term preferences

	n = 1	n = 2	...	n = 14	n = 15
	2.7	2.4	...	1.3	1.2
	1.1	1.0	...	0.5	0.3
	3.1	2.6	...	0.0	0.0
Avg.	2.3	2.0	...	0.6	0.5

Analysis - 1



Amazon-Book
Amazon-Beauty
short v.s. long
random v.s. long
JP-VOD
Yelp2018
Average KLD
Average KLD
2.0
1.5
1.0
0.5
0.0
3.0
2.4
1.8
1.2
0.6
0.0
5
10
15
n
n

Analysis - 2

- After confirming that short-term preferences exist, we want to find a best n for predicting user's future behavior.
- Firstly, we sort each user's behavior logs in the training period with temporal order.
- Then, for each user, we split the item set in to a new training set T_u (80%) and a validation set V_u (20%), and represent items by the column vectors on the user-item table.

Analysis - 2

- There are two kinds of preferences for every user, which are represented as normalized sum of item vectors:
  - Future: all items in V_u
  - Short-term: last n items in T_u

- From n=1 to n=15, we calculate every user's Kullback-Leibler Divergence (KLD) between Short-term preference and Future preference.

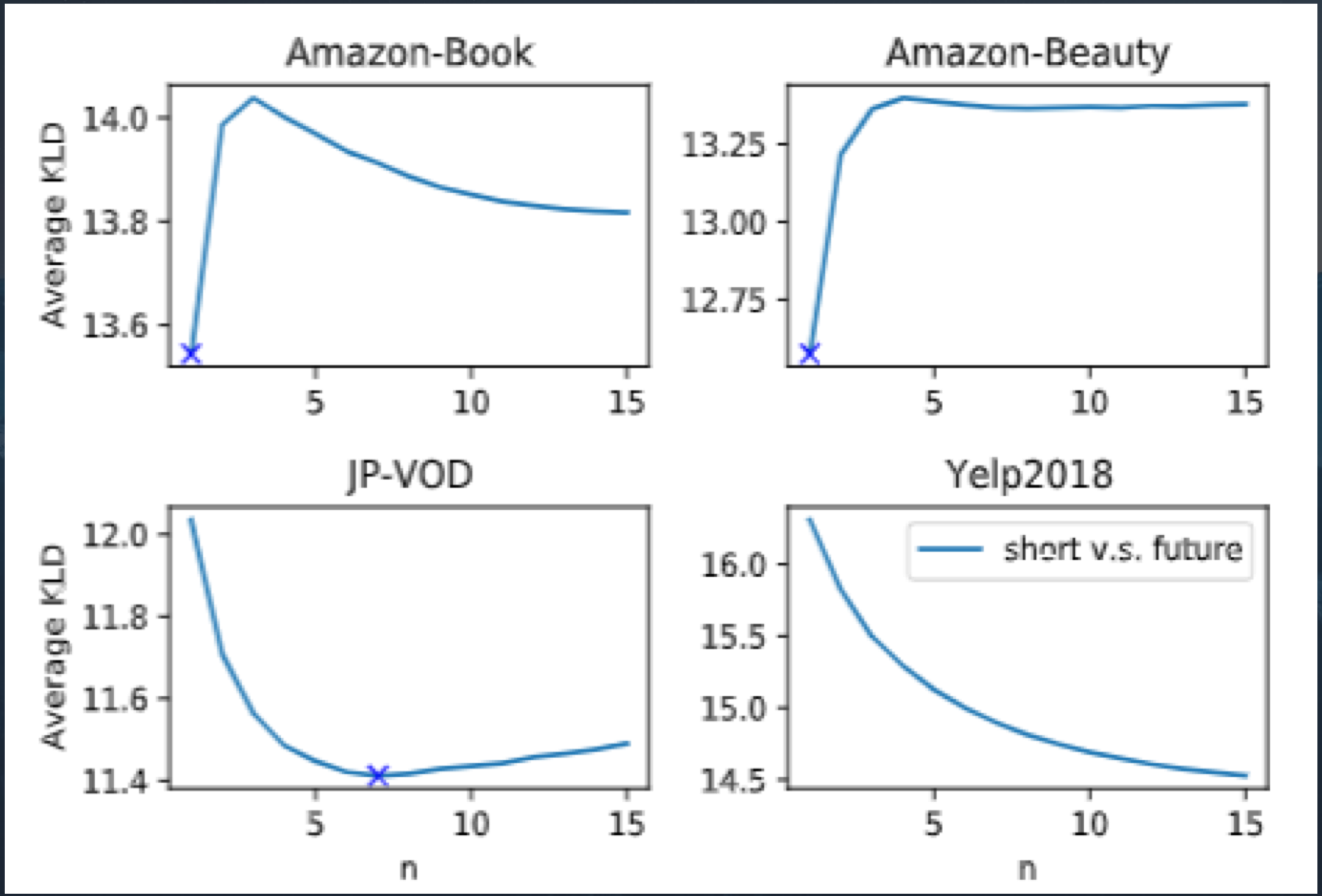
Analysis - 2

- Calculate the average of all user's KLDs
- Plot Avg. KLDs vs. n

KLDs between short-term and future preferences

	n = 1	n = 2	...	n = 14	n = 15
	10.5	11.3	...	9.9	9.8
	13.4	10.4	...	11.7	12.1
	11.5	11.3	...	11.1	11.4
Avg.	11.8	11.0	...	10.9	11.1

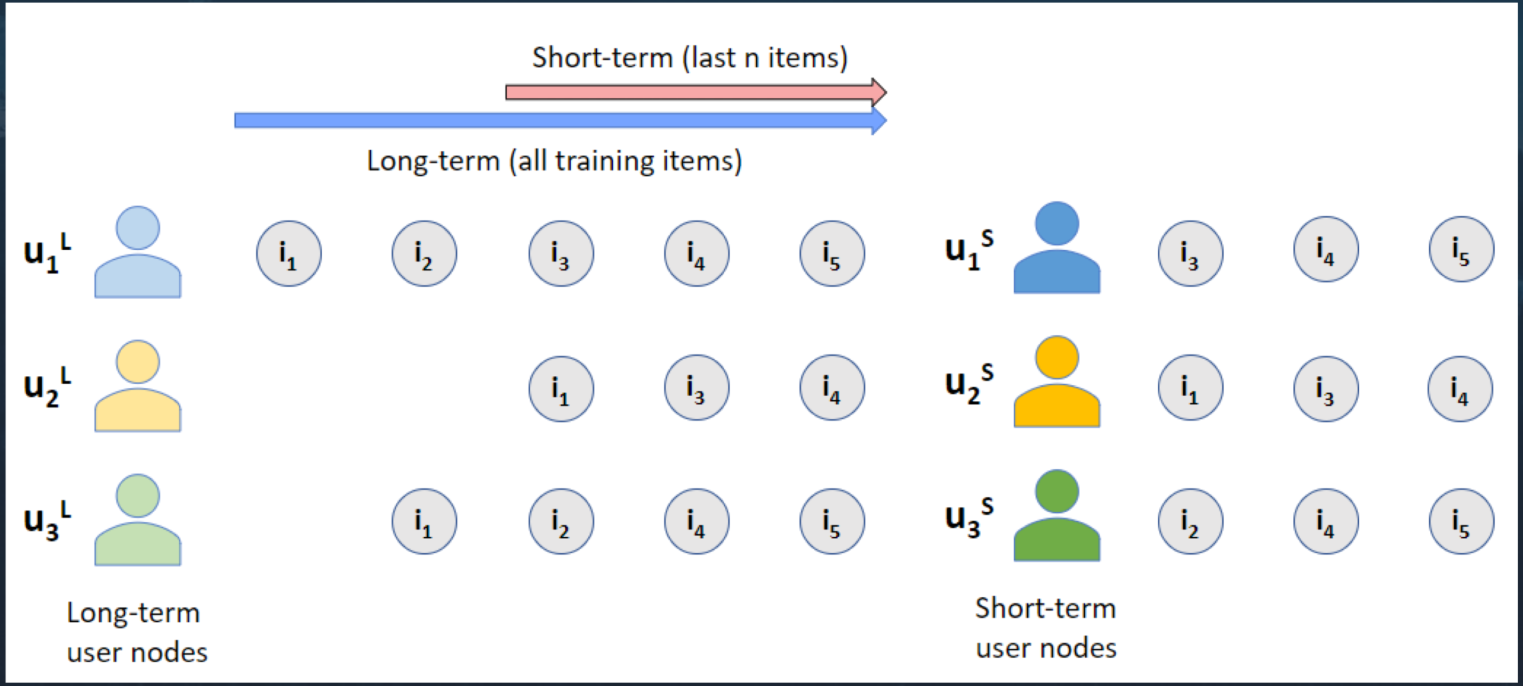
Analysis - 2


Amazon-Book
Amazon-Beauty
JP-VOD
Yelp2018
Average KLD
Average KLD
14.0
13.8
13.6
13.25
13.00
12.75
12.0
11.8
11.6
11.4
16.0
15.5
15.0
14.5
5
10
15
n
n
short v.s. future

Method

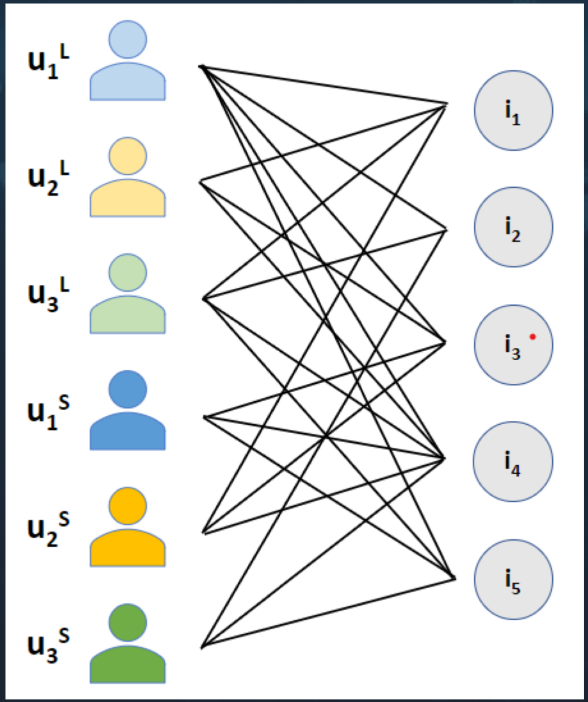
Method

- Sort the user-item interactions with temporal order
- Select a suitable n (5, 10, and 15 are used in our experiments)
- Generate two kinds of user nodes



Method

- Build the user-item bipartite graph
- Optimize the embeddings by the modified loss function and k-order sampling probability



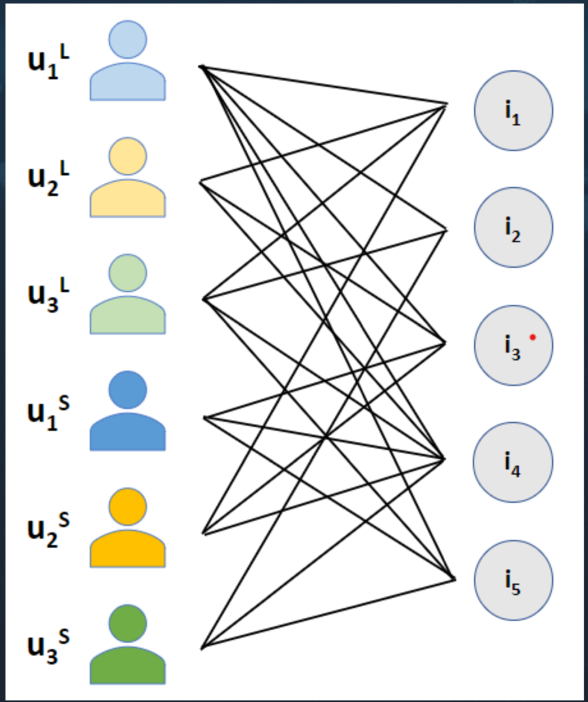
$$p_v^k(\ell) = \begin{cases} \frac{a_{v\ell}\deg(\ell)}{\sum_{\ell'} a_{v\ell'}\deg(\ell')}, & \text{if } k = 1 \text{ and } v \in \mathcal{U}^L \cup \mathcal{U}^S \\ \frac{a_{\ell v}\deg(\ell)}{\sum_{\ell'} a_{\ell' v}\deg(\ell')}, & \text{if } k = 1 \text{ and } v \in \mathcal{I} \\ p_v^1(\beta) p_\beta^{k-1}(\gamma) p_\gamma^1(\ell), & \text{if } k > 1, \end{cases}$$

$$\begin{aligned} \mathcal{L}^{\text{LSTPR}} = & \sum_{\substack{1 \le k \le K \\ u^S,(i,i')}} \rho(k)\, \mathbb{E}_{\substack{i \sim p^k_{u^S} \\ i' \sim P_N}} \left[\mathbb{1}_{\{e_u^{S\top} e_{i'} - e_u^{S\top} e_i > \xi_k\}} \mathcal{K}(e_u^{S\top} e_{i'}, e_u^{S\top} e_i) \right] \\ & + \sum_{\substack{1 \le k \le K \\ u^L,(j,j')}} \rho(k)\, \mathbb{E}_{\substack{j \sim p^k_{u^L} \\ j' \sim P_N}} \left[\mathbb{1}_{\{e_u^{L\top} e_{j'} - e_u^{L\top} e_j > \xi_k\}} \mathcal{K}(e_u^{L\top} e_{j'}, e_u^{L\top} e_j) \right] \\ & + \lambda_\Theta \|\Theta\|^2, \end{aligned}$$

Ref: https://dl.acm.org/doi/10.1145/3240323.3240381

Method

- Build the user-item bipartite graph
- Optimize the embeddings by the modified loss function and k-order sampling probability



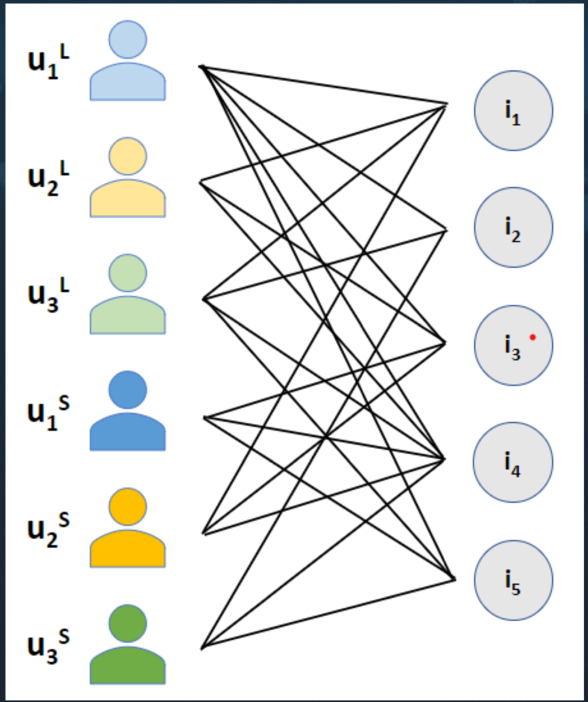
$$p_v^k(\ell) = \begin{cases} \frac{a_{v\ell}\deg(\ell)}{\sum_{\ell'} a_{v\ell'}\deg(\ell')}, & \text{if } k = 1 \text{ and } v \in \mathcal{U}^L \cup \mathcal{U}^S \\ \frac{a_{\ell v}\deg(\ell)}{\sum_{\ell'} a_{\ell' v}\deg(\ell')}, & \text{if } k = 1 \text{ and } v \in \mathcal{I} \\ p_v^1(\beta)p_\beta^{k-1}(\gamma)p_\gamma^1(\ell), & \text{if } k > 1, \end{cases}$$

$$\begin{aligned} \mathcal{L}^{\text{LSTPR}} = & \sum_{\substack{1\le k\le K \\ u^S,(i,i')}} \rho(k)\, \mathbb{E}_{\substack{i\sim P^k_{u^S} \\ i'\sim P_N}} \left[\mathbb{1}_{\{e_u^{S\top} e_{i'} - e_u^{S\top} e_i > \xi_k\}} \mathcal{K}(e_u^{S\top} e_{i'}, e_u^{S\top} e_i) \right] \\ & + \sum_{\substack{1\le k\le K \\ u^L,(j,j')}} \rho(k)\, \mathbb{E}_{\substack{j\sim P^k_{u^L} \\ j'\sim P_N}} \left[\mathbb{1}_{\{e_u^{L\top} e_{j'} - e_u^{L\top} e_j > \xi_k\}} \mathcal{K}(e_u^{L\top} e_{j'}, e_u^{L\top} e_j) \right] \\ & + \lambda_\Theta \|\Theta\|^2, \end{aligned}$$

Ref: https://dl.acm.org/doi/10.1145/3240323.3240381

Method

- Build the user-item bipartite graph
- Optimize the embeddings by the modified loss function and k-order sampling probability



$$p_v^k(\ell) = \begin{cases} \frac{a_{v\ell}\deg(\ell)}{\sum_{\ell'} a_{v\ell'}\deg(\ell')}, & \text{if } k = 1 \text{ and } v \in \mathcal{U}^L \cup \mathcal{U}^S \\ \frac{a_{\ell v}\deg(\ell)}{\sum_{\ell'} a_{\ell' v}\deg(\ell')}, & \text{if } k = 1 \text{ and } v \in \mathcal{I} \\ p_v^1(\beta) p_\beta^{k-1}(\gamma) p_\gamma^1(\ell), & \text{if } k > 1, \end{cases}$$

$$\mathcal{L}^{\text{LSTPR}} = \sum_{\substack{1 \le k \le K \\ u^S,(i,i')}} \rho(k)\, \mathbb{E}_{\substack{i \sim p^k_{u^S} \\ i' \sim P_N}} \left[\mathbb{1}_{\{e_u^{S\top} e_{i'} - e_u^{S\top} e_i > \xi_k\}} \mathcal{K}(e_u^{S\top} e_{i'}, e_u^{S\top} e_i) \right] + \sum_{\substack{1 \le k \le K \\ u^L,(j,j')}} \rho(k)\, \mathbb{E}_{\substack{j \sim p^k_{u^L} \\ j' \sim P_N}} \left[\mathbb{1}_{\{e_u^{L\top} e_{j'} - e_u^{L\top} e_j > \xi_k\}} \mathcal{K}(e_u^{L\top} e_{j'}, e_u^{L\top} e_j) \right] + \lambda_\Theta \|\Theta\|^2,$$

Long short-term preferences

Graph

Factorization

Ref: https://dl.acm.org/doi/10.1145/3240323.3240381

Method

- After training the model, calculate the dot product of short-term user embedding and item embedding to make top-N recommendation

$$\hat{s}(u, i) = {e_u^S}^\top e_i.$$

	Amazon-Book		Amazon-Beauty		JP-VOD	
	Recall@10 / 20	NDCG@10 / 20	Recall@10 / 20	NDCG@10 / 20	Recall@10 / 20	NDCG@10 / 20
BPR [12]	0.0181 / 0.0302	0.0155 / 0.0198	0.0454 / 0.0753	0.0333 / 0.0449	0.0910 / 0.1423	0.0936 / 0.1110
WARP [20]	0.0330 / 0.0565	0.0279 / 0.0366	0.0457 / 0.0710	0.0342 / 0.0439	0.0973 / 0.1536	0.0968 / 0.1167
HOP-Rec [23]	0.0381 / 0.0651	0.0318 / 0.0418	0.0572 / 0.0948	0.0419 / 0.0562	0.1146 / 0.1808	0.1119 / 0.1359
Skew-OPT [16]	0.0422 / 0.0709	0.0350 / 0.0457	0.0530 / 0.0826	0.0408 / 0.0523	0.1124 / 0.1748	0.1110 / 0.1333
LightGCN [3]	†0.0432 / †0.0721	†0.0355 / †0.0462	0.0584 / 0.0893	0.0412 / 0.0528	†0.1172 / †0.1865	0.1138 / †0.1390
Caser [15]	0.0223 / 0.0393	0.0178 / 0.0241	0.0552 / 0.0927	0.0432 / 0.0577	0.1070 / 0.1636	†0.1212 / 0.1377
CosRec [22]	0.0248 / 0.0423	0.0241 / 0.0304	†0.0634 / †0.0979	†0.0518 / †0.0649	0.0757 / 0.1206	0.0857 / 0.0994
LSTPR ($n = 5$)	**0.0519** / **0.0801**	**0.0485** / **0.0581**	**0.0670** / 0.1008	**0.0558** / **0.0687**	**0.1399** / 0.2070	**0.1524** / **0.1722**
LSTPR ($n = 10$)	0.0478 / 0.0771	0.0443 / 0.0544	0.0649 / **0.1015**	0.0525 / 0.0665	0.1362 / **0.2083**	0.1476 / 0.1698
LSTPR ($n = 15$)	0.0444 / 0.0729	0.0404 / 0.0504	0.0642 / 0.0990	0.0488 / 0.0622	0.1304 / 0.2019	0.1405 / 0.1630
Improv. (%)	+20.13% / +11.10%	+36.62% / +25.76%	+5.68% / +3.68%	+7.72% / +5.86%	+19.37% / +11.69%	+31.72% / +23.88%
Paired t-test	▲▲ / ▲▲	▲▲ / ▲▲	▲▲/▲	▲▲ / ▲▲	▲▲ / ▲▲	▲▲ / ▲▲

Conclusion

Conclusion

- Short-term preferences do exist, but they are not always beneficial to recommendation performances.
- Based on the analyses, we propose LSTPR, which efficiently and effectively leverages long-short term preferences and high-order graph information at the same time.
- LSTPR outperforms seven strong baselines with a significant margin on the three datasets.

Thanks for listening!

If you have any question, please feel free to contact us!

Github: https://github.com/cnclabs/codes.lstp.rec

Chih-Hen Lee: ch.lee@citi.sinica.edu.tw

Jun-En Ding: ding1119@citi.sinica.edu.tw

Chih-Ming Chen: 104761501@nccu.edu.tw