Do Loyal Users Enjoy Better Recommendations? Understanding Recommender Accuracy from a Time Perspective

Yitong Ji
Nanyang Technological University
Singapore
yitong.ji@ntu.edu.sg

Aixin Sun
Nanyang Technological University
Singapore
axsun@ntu.edu.sg

Jie Zhang
Nanyang Technological University
Singapore
zhangj@ntu.edu.sg

Chenliang Li
Wuhan University
China
cllee@whu.edu.cn

ABSTRACT

In academic research, recommender systems are often evaluated on benchmark datasets, without much consideration about the *global timeline*. Hence, we are unable to answer questions like: *Do loyal users enjoy better recommendations than non-loyal users?* Loyalty can be defined by the time period a user has been active in a recommender system, or by the number of historical interactions a user has. In this paper, we offer a comprehensive analysis of recommendation results along global timeline. We conduct experiments with five widely used models, *i.e.,* BPR, NeuMF, LightGCN, SASRec and TiSASRec, on four benchmark datasets, *i.e.,* MovieLens-25M, Yelp, Amazon-music, and Amazon-electronic. Our experiment results give an answer "No" to the above question. Users with many historical interactions suffer from relatively poorer recommendations. Users who stay with the system for a shorter time period enjoy better recommendations. Both findings are counter-intuitive. Interestingly, users who have recently interacted with the system, with respect to the time point of the test instance, enjoy better recommendations. The finding on recency applies to all users, regardless of users' loyalty. Our study offers a different perspective to understand recommender accuracy, and our findings could trigger a revisit of recommender model design. The code is available in https://github.com/putatu/recommenderLoyalty.

CCS CONCEPTS

• **General and reference** → **Empirical studies**; **Performance**; • **Information systems** → **Recommender systems**.

KEYWORDS

Recommender system; Time factor; Loyal and non-loyal users

ICTIR '22, July 11–12, 2022, Madrid, Spain

ACM ISBN 978-1-4503-9412-3/22/07…$15.00
https://doi.org/10.1145/3539813.3545124

ACM Reference Format:
Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. 2022. Do Loyal Users Enjoy Better Recommendations? Understanding Recommender Accuracy from a Time Perspective. In *Proceedings of the 2022 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '22), July 11–12, 2022, Madrid, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3539813.3545124

1 INTRODUCTION

Recommender systems are often evaluated offline on a benchmark dataset in academic research [19, 21]. That is, a dataset is split into training set and test set. Then, a recommender model learns from the training set and is evaluated on the test set. Many papers report overall accuracy of their models on the test set, and conduct ablation studies. However, analysis of recommendation accuracy from time dimension is often lacking. In this paper, we study recommendation accuracy by time-related factors, and aim to answer questions like: *Do loyal users enjoy better recommendations than non-loyal users?*

We consider three time factors along the global timeline, illustrated in Figure 1. Suppose a test instance happens at time t, we have (i) **number of accumulated interactions** of a user before time point t; (ii) **active time period**, *e.g.,* number of days since the user's first interaction with any item in the system till time t; and (iii) **recency**, *e.g.,* number of days from the user's previous interaction to time t. We consider both "number of accumulated interactions" and "active time period" as loyalty indicators, because both factors reflect a user's relationship with the system, or to what extent the user is familiar with the system.

We evaluate five recommendation models - BPR [20], NeuMF [8], LightGCN [7], SASRec [13], and TiSASRec [17]. Among them, BPR, NeuMF, and LightGCN are general batch mode recommenders without considering time in their design. That is, they model a user's preference using all his/her historical interactions, and do not consider at what time points these interactions take place. SASRec is a sequence-aware model which considers user's historical interactions in chronological order, while TiSASRec is a time-aware recommender which takes time into consideration in their model design. Our experiments are conducted on four publicly available datasets, all with 10 years of interactions: MovieLens-25M, Yelp, Amazon-music, and Amazon-electronic.

Experiment results show that *loyal users suffer from poorer recommendations*, compared to non-loyal users for general recommenders

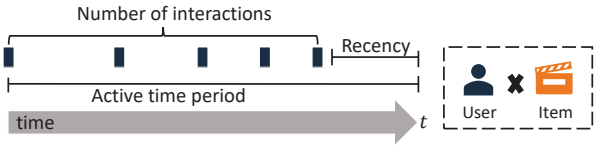

Figure 1: The time factors we considered in our analysis

BPR, NeuMF, and LightGCN. This finding is counter-intuitive. In general, we expect that preferences of loyal users can be more accurately modeled by recommenders, because they have more interactions or have spent more time with the system. This counter-intuitive finding provokes us into thinking whether *all historical interactions of users* are indeed useful for learning user preference. For sequential recommenders, *i.e.,* SASRec and TiSASRec, there are cases whereby loyal users and non-loyal users enjoy comparable recommendation accuracy. One reason is that SASRec and TiSASRec work with each user's local timeline, hence they treat recent interactions of a user differently from his/her past interactions. Here, local timeline refers to the sequential order of a specific user's interactions, without considering the absolute time of these interactions along the global timeline.

Further analysis on the **recency** factor shows that, users who have interactions close to the test time point t, receive better recommendations, than those who last interact with the system some time ago. This observation holds for all the five recommenders, regardless of whether they are time-aware or not. It is a strong evidence that more recent interactions, with respect to the test time, are more helpful in recommendation. Here, recent interactions are defined with respect to the *global timeline* (see Figure 1). Note that, time-aware models SASRec and TiSASRec do not show better ability in distinguishing recent interactions along the global timeline, because they model a user's interactions as a sequence and does not consider interactions with respect to the test time.

Our research findings suggest that user's interests change in a long run. More recent interactions better reflect users' current interests, while past interactions could be considered outdated. However, recent interactions refer to a dynamic and changing set of interactions along the global time dimension because the number of available items in the system may change from time to time. These interactions cannot be accurately modeled based on a user's local timeline. That is, to a specific user, the most recent interaction with respect to his/her past interactions may not be "recent" with respect to a time point along the global timeline. For example, a user's last purchase from an e-commerce website may happen in last day, last week, or even last year. Based on our findings, we call for a revisit on recommender model design. Rather than treating all historical interactions equally, we might need to pay more attention to recent interactions when a recommendation decision is to be made, along the global timeline.

2 RELATED WORK

Many existing recommender models take time into consideration. Commonly, time is taken into account to preserve sequential order of a user's interactions. For instance, GRU4REC [9] applies Recurrent Neural Network (RNN) on sessions to capture sequential dependencies between items in user's interactions. Some other studies [16, 18] combine attention mechanism with RNN network, and model not only the sequential relationship in sessions but also relative importance of items along the sessions. Inspired by *Transformer*, self-attention network based recommenders [13, 17, 22, 23] have also been widely adopted to learn long-term dependencies. Positional embedding is used in self-attention network to model position of an item in an interaction session. Time has also been considered in recommendation as an attribute or contextual information. Analysis on Amazon-datasets shows that demand of T-shirts follows a seasonal cycle [24]. Hansen *et al.* [6] highlight the importance of time context in music consumption. There is also study on the time context in emoji recommendation [25]. These recommenders embed time context into embedding vectors and incorporate time context features in recommendation. Another way to take time into consideration is by relative time. A few studies [17, 24] emphasize the importance of time interval between interactions in recommendation.

Time has also been factored in through recency of past interactions, particularly in the news recommendation task. Recency is defined with respect to the time when a recommendation is made. The authors in [3] acknowledge the importance of recency, and propose a news recommender model that balances the trade-off between recency and relevancy. Here, recency concerns *item*'s age at the time point of recommendation, as the items are news articles. Hence, their definition of recency is different from ours. Other work like [15] considers recency of past interactions in collaborative filtering. It shows that adopting different recency windows in recommender design can lead to different recommendation accuracy. Similarly, the authors of [4] demonstrate the influence of recency based decay functions used in collaborative filtering recommenders.

The above reviewed papers incorporate time factors in their recommender designs, either motivated by pre-experiment analysis on datasets, or by intuition and domain knowledge. In contrast, we conduct *post-experiment analysis* on the recommendation results of various models, on multiple datasets representing different recommendation tasks. Our analysis further explains the influence of time factors (*e.g.,* active time period, and recency) in general recommendation setting.

3 EXPERIMENT

In this section, we detail datasets, the evaluated recommendation models, and the experiment setup.

3.1 Datasets and Evaluated Models

We conduct experiments on four public datasets: MovieLens-25M, Yelp, Amazon-music, and Amazon-electronic. All interactions in these four datasets come with timestamps. From each dataset, we extract interactions in a 10-year time period for experiments (see Table 1 for the starting time of each dataset). After that, we filter the extremely inactive items or users. We adopt k-core filtering for Yelp, Amazon-music, and Amazon-electronic. After k-core filtering, all users and items in the dataset will have at least k interactions. Depending on the dataset size and sparsity, we choose 10-core for Yelp and Amazon-electronic, and 5-core for Amazon-music.

Table 1: Statistics of the four datasets, all covering interactions in 10-year period from their starting time.

Dataset	Starting time	Data filtering	#User	#Item	#Data instances
MovieLens-25M	21 Nov 2009	No filtering	62, 202	56, 774	9, 808, 925
Yelp	13 Dec 2009	10-core	116, 655	61, 027	3, 127, 215
Amazon-music	02 Oct 2008	5-core	11, 651	9, 243	114, 833
Amazon-electronic	05 Oct 2008	10-core	109, 990	39, 552	1, 752, 238

Table 2: Average #Interactions, ATP (days) and Recency (days) for each user group.

Dataset	Baselines	#Interactions		ATP (days)		Recency (days)	
		Loyal Users	Non-loyal Users	Loyal Users	Non-loyal Users	Active Users	Inactive Users
MovieLens-25M	General Recommenders	505.0	61.4	909.2	3.3	$1.6e-4$	32.2
	SASRec	40.0	28.1	385.9	1.1	$1.6e-4$	32.2
	TiSASRec	36	26.4	366.9	1.0	$1.6e-4$	32.2
Yelp	General Recommenders	48.2	11.8	2299.2	964.1	12.5	283.0
	SASRec	17.9	11.8	1899.1	622.9	12.5	283.0
	TiSASRec	32.2	11.8	2173.8	864.9	12.5	283.0
Amazon-music	General Recommenders	17.7	4.7	1592.3	445.0	1.7	447.8
	SASRec	16.1	4.7	1578.5	434.2	1.7	447.8
	TiSASRec	15.6	4.7	1572.0	429.4	1.7	447.8
Amazon-electronic	General Recommenders	21.8	10.2	2208.5	1012.0	15.1	373.8
	SASRec	21.1	10.2	2188.9	1003.1	15.1	373.8
	TiSASRec	18.5	10.2	2101.1	940.1	15.1	373.8

No filtering is performed on MovieLens-25M because MovieLens dataset ensures each user has at least 20 ratings. Table 1 reports the statistics of the four datasets after filtering.

Our findings are made on recommendation results by five widely used baseline models[1]: BPR, NeuMF, LightGCN, SASRec, and TiSASRec. Each baseline represents one type of recommendation. BPR, NeuMF, and LightGCN are general recommender models which do not take time into consideration. **BPR** learns users' and items' latent factors via a pairwise ranking loss in matrix factorization. **NeuMF** learns user and item interaction function with a model that combines both matrix factorization and multi-layer perceptron. **LightGCN** is a graph-based recommender model that learns complex relationships between users and items, by using graph convolutional network. SASRec and TiSASRec are time-aware models which consider time at which interactions take place. **SASRec** is a self-attentive network that models the sequential pattern in user's behaviour. **TiSASRec** is inspired from SASRec but takes time intervals between events as input.

In our experiments, we adopt leave-one-out data split but with well consideration of data leakage, to be detailed in Section 3.2. For each model, we tune their hyperparameters by continuous random search, in each run of the experiment. Similar to other work [1, 10], we tune hyperparameters by using a validation set that consists of the second last interaction of each user.

[1]We implement the general recommenders, *i.e.,* BPR, NeuMF and LightGCN, using Recbole [26]. For SASRec and TiSASRec, we follow the implementations in https://github.com/pmixer/SASRec.pytorch and https://github.com/JiachengLi1995/TiSASRec respectively.

We adopt Hit Rate (HR) and Normalized Discounted Cumulative Gain (NDCG) as evaluation metrics. In this research, we rank **all available items** and make top-N recommendations. That is, we do not use sampled metrics, because sampled metrics introduce bias in recommendation accuracy [14].[2]

3.2 Evaluation Setting

Recently a few studies [5, 11, 12] highlight the issue of data leakage in offline evaluation of recommender system. In particular, data partition strategies (*e.g.,* leave-one-out and random-split-by-ratio) that do not follow global timeline will allow the training of recommendation model using *future* training instances. Future training instances are the interactions that take place after the time point of a test instance. To strictly follow the global timeline and completely avoid data leakage, it is suggested to adopt a streaming setting [12], *i.e.,* taking interactions happened before a time point t for training and predict test instances at t, with a changing t along the global timeline. This streaming setting requires the recommenders to be designed to take in incremental inputs along timeline. However, BPR, NeuMF, LightGCN, SASRec and TiSASRec are not incremental. We understand incremental models exist. However, batch recommendation models remain the mainstream in academic research. Here, our objective is to conduct post-experiment analysis of batch models' results by time factors.

To minimize the impact of data leakage on batch models, we follow the evaluation setting used in [12] for our experiments. The key idea is to keep the number of future instances reasonably

[2]In this paper, we only present the HRs and NDCGs on top-10 recommendations. Similar findings hold for top-5 and top-20 recommendations.

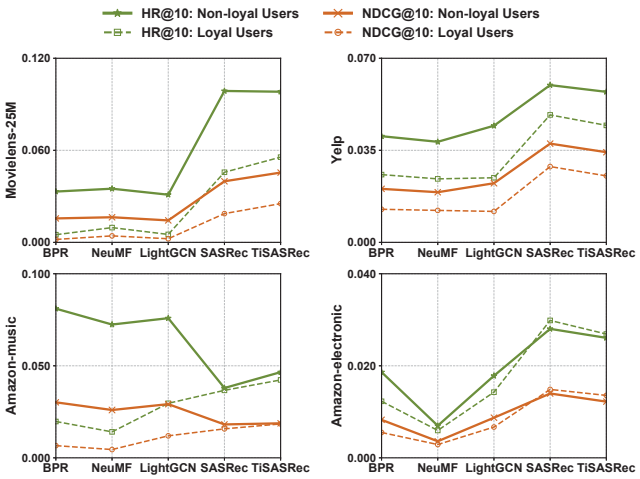

Figure 2: HR@10 and NDCG@10 of loyal and non-loyal users by number of interactions, in test year $Y10$

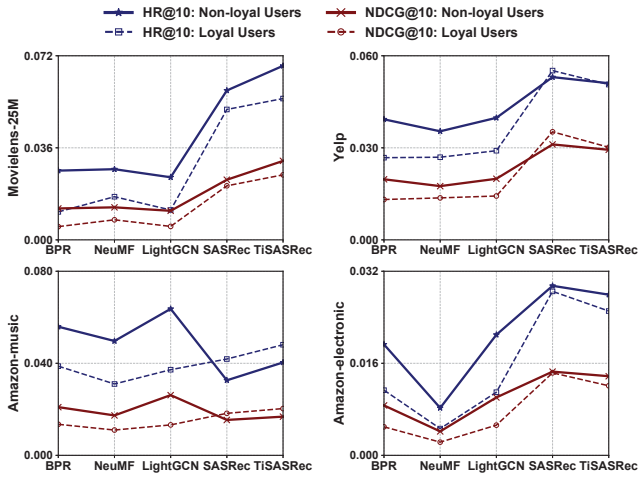

Figure 3: HR@10 and NDCG@10 of loyal and non-loyal users by active time period, in test year $Y10$

small. Specifically, given a dataset, we first conduct *leave-one-out* split. That is, for each user, we treat his/her last interaction as test instance, and the remaining interactions as training instances. We adopt *leave-one-out* split in this study because it is a widely used data partitioning strategy in recommender system [21]. Moreover, it allows the recommendation models to have a complete picture of a user's historical interactions by masking only the last interaction of the user as test instance. After obtaining training and test sets, we follow [12] to conduct evaluation on test instances that happen in a specific year.

Recall that each dataset contains 10-year instances, we select test instances that happened in a particular year for evaluation. Assuming we take year-6, denoted by $Y6$, as the test year, then test instances that happened in $Y6$ will be in the test set for this run of experiment. All instances before $Y6$ (*i.e.,* $Y1$ to $Y5$) and the training instances in $Y6$ will be used for training. Based on this experiment setting, we run three sets of experiments on each of the four datasets, using $Y6$, $Y8$, and $Y10$ as test years respectively. We experiment on three test years because recommender behaviours can vary along the timeline [2]. Through experiments, we find that results on three test years show very similar trends. Hence, we only present the results for test year $Y10$ to avoid having too many similar plots.

4 EXPERIMENT RESULTS

We now study recommendation results to answer questions on user loyalty. Here, loyalty can be indicated by: (i) number of accumulated interactions a user has, and (ii) active time period of a user.

4.1 Loyalty by Number of Interactions

Recall that we use *leave-one-out-split* to partition the dataset into training and test sets. That is, we treat each user's last interaction as test instance while the remaining interactions that happened before the test instance are training instances. The number of user's interactions in the training set can be an indicator of user loyalty. For simplicity, we rank users by their number of interactions in training set. The users who rank among the top 50% are *loyal users*; the rest are *non-loyal users*. The average number of interactions for both loyal users and non-loyal users can be found in Table 2. Note that, for SASRec and TiSASRec, only the most recent n interactions of each user are used in training. That is because SASRec and TiSASRec are built on top of self-attention network which requires memory quadratic to sequence length of user interactions. Following the original papers of SASRec and TiSASRec, we tune n as a hyperparameter. Hence, in Table 2, the average number of interactions for loyal users and non-loyal users appears to be different for SASRec, TiSASRec and the general recommenders - BPR, LightGCN and NeuMF.

Figure 2 plots the recommendation accuracy by HR@10 and NDCG@10 for loyal and non-loyal users that are defined by number of interactions. For HR@10, all the five models give better recommendations to non-loyal users on MovieLens-25M, Yelp, and Amazon-music. On Amazon-electronics, only SASRec and TiSASRec offer better results for loyal users. As for NDCG@10, non-loyal users enjoy better recommendations than loyal users except for TiSASRec on Amazon-music as well as SASRec and TiSASRec on Amazon-electronic. Nevertheless, the overall trend suggests that *non-loyal users enjoy better recommendations than loyal users*, particularly for general recommenders. This finding violates our intuition that more historical data leads to better learning of user's preference in recommendation. Hence, we hypothesize that not all historical interactions by a user are helpful for making "recent" recommendations.

4.2 Loyalty by Active Time Period

Purely considering *number of interactions* as a loyalty measure is not sufficient. A user can have many interactions within a short time period. In light of this, we introduce a new loyalty indicator: Active Time Period (ATP), which is the number of days between a user's first interaction and last interaction. If a user has a long

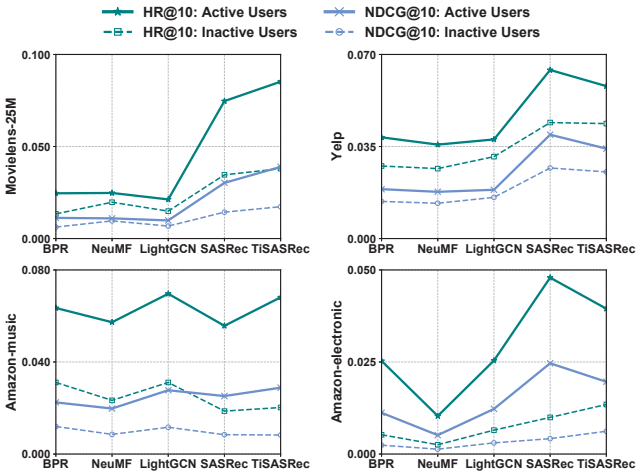

Figure 4: HR@10 and NDCG@10 for users grouped by recency of previous interaction before test instance.

active time period, then the user has a long-term engagement with the system, and is a loyal user.

In our experiments, ATP is the number of days between a user's first interaction in the training set and the time of his/her test instance. Loyal users have ATP longer than the median ATP value. Figure 3 plots HR@10 and NDCG@10 of loyal and non-loyal users indicated by active time period.

In general, non-loyal users, who have shorter active time period, enjoy better HR@10 and NDCG@10 than loyal users. A loyal user, by active time period, is a user who has started interacting with the system since a long time ago. On the one hand, the system has a better chance to capture the user's long-term interest. On the other hand, interactions that happened long time ago may not necessary represent the user's current preference. In Figure 3, our results suggest mostly the latter on multiple datasets. We also observe that loyal and non-loyal users enjoy comparable recommendation accuracy by SASRec and TiSASRec on four datasets. That is because SASRec and TiSASRec are designed to treat recent interactions and old interactions differently for each individual user.

So far, our results show that preference of loyal users may not be well predicted, especially for general recommenders which treat all training instances equally. We hypothesize that this observation is attributed to the outdated interactions that happened a long time ago. These interactions cannot reflect a user's current interest. Hence, it motivates us to investigate another time factor: recency.

4.3 Recency

We define recency by the number of days between a user's test instance and his/her previous interaction just before the test instance. Again, we rank all users and take the 50^{th} percentile recency as threshold. We name the users with short recency as *active users* and the remaining as *inactive users*. As shown in Table 2, average recency for active users and inactive users are of the same values for all recommenders. Regardless of whether the models are general recommenders or sequential recommenders, recency value is not affected because it is calculated based on the time of the last interaction and the time of the test instance.

Figure 4 plots HR@10 and NDCG@10 scores for both user groups. All recommenders, both general and sequential recommenders, deliver better recommendation results for active users, by both HR@10 and NDCG@10 measures. This finding provides strong evidence that more recent interactions matter more. This finding also partially explains the previous findings which suggest that loyal users suffer from poorer recommendations than non-loyal users. Moreover, we note that all recommenders, regardless of whether they consider time information in their model design, show similar trends on the recency factor. Even time-aware recommenders, SASRec and TiSASRec, do not consider the recency factor because "time" is considered only during model training, but not when making recommendations at the test time. In fact, SASRec and TiSASRec both consider local timeline specific to a user (*i.e.,* a user's interaction sequence) but not the global timeline. Based on the finding on recency, we argue that recommender shall consider recent interactions dynamically, with respect to the time point when making recommendations, along the global timeline.

5 CONCLUSION

In this paper, through experiments, we show that **time matters** in recommendation. Here, we refer time with respect to the "global timeline" instead of "local timeline" where the latter is specific to a user. Having many interactions or having a long active time may adversely affect recommendation accuracy. Instead, recommenders give better results if users have recent interactions. On the other hand, recent interactions change continuously along the global timeline. Hence, we call for a revisit of model design in recommender system. The model should be able to capture the dynamics along the global timeline, thus to update user preference with the most recent context.

REFERENCES

[1] Ting Bai, Lixin Zou, Wayne Xin Zhao, Pan Du, Weidong Liu, Jian-Yun Nie, and Ji-Rong Wen. 2019. CTRec: A Long-Short Demands Evolution Model for Continuous-Time Recommendation. In *SIGIR*. ACM, 675–684.
[2] Jöran Beel. 2017. It's Time to Consider "Time" when Evaluating Recommender-System Algorithms [Proposal]. *CoRR* abs/1708.08447 (2017).
[3] Abhijnan Chakraborty, Saptarshi Ghosh, Niloy Ganguly, and Krishna P. Gummadi. 2017. Optimizing the Recency-Relevancy Trade-off in Online News Recommendations. In *WWW*. ACM, 837–846.
[4] Guglielmo Faggioli, Mirko Polato, and Fabio Aiolli. 2020. Recency Aware Collaborative Filtering for Next Basket Recommendation. In *UMAP*. ACM, 80–87.
[5] Milena Filipovic, Blagoj Mitrevski, Diego Antognini, Emma Lejal Glaude, Boi Faltings, and Claudiu Musat. 2021. Modeling Online Behavior in Recommender Systems: The Importance of Temporal Context. In *RecSys (CEUR Workshop Proceedings, Vol. 2955)*. CEUR-WS.org.
[6] Casper Hansen, Christian Hansen, Lucas Maystre, Rishabh Mehrotra, Brian Brost, Federico Tomasi, and Mounia Lalmas. 2020. Contextual and Sequential User Embeddings for Large-Scale Music Recommendation. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*. ACM, 53–62.
[7] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yong-Dong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*. ACM, 639–648.
[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *WWW*. ACM, 173–182.
[9] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[10] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR*. ACM, 505–514.
[11] Olivier Jeunen. 2019. Revisiting offline evaluation for implicit-feedback recommender systems. In *RecSys*. ACM, 596–600.
[12] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. 2022. A Critical Study on Data Leakage in Recommender System Offline Evaluation. *CoRR* abs/2010.11060 (2022).
[13] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM*. IEEE Computer Society, 197–206.
[14] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD*. ACM, 1748–1757.
[15] Santiago Larrain, Christoph Trattner, Denis Parra, Eduardo Graells-Garrido, and Kjetil Nørvåg. 2015. Good Times Bad Times: A Study on Recency Effects in Collaborative Filtering for Social Tagging. In *RecSys*. ACM, 269–272.
[16] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM*. ACM, 1419–1428.
[17] Jiacheng Li, Yujie Wang, and Julian J. McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *WSDM*. ACM, 322–330.
[18] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: Short-Term Attention/Memory Priority Model for Session-based Recommendation. In *KDD*. ACM, 1831–1839.
[19] Ladislav Peska and Peter Vojtás. 2020. Off-line vs. On-line Evaluation of Recommender Systems in Small E-commerce. In *HT*. ACM, 291–300.
[20] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. AUAI Press, 452–461.
[21] Zhu Sun, Di Yu, Hui Fang, Jie Yang, Xinghua Qu, Jie Zhang, and Cong Geng. 2020. Are We Evaluating Rigorously? Benchmarking Recommendation for Reproducible Evaluation and Fair Comparison. In *RecSys*. ACM, 23–32.
[22] Md. Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane Hu, Liangjie Hong, and Julian J. McAuley. 2020. Attentive Sequential Models of Latent Intent for Next Item Recommendation. In *WWW*. ACM / IW3C2, 2528–2534.
[23] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item Recommendation with Sequential Hypergraphs. In *SIGIR*. ACM, 1101–1110.
[24] Wenwen Ye, Shuaiqiang Wang, Xu Chen, Xuepeng Wang, Zheng Qin, and Dawei Yin. 2020. Time Matters: Sequential Recommendation with Complex Temporal Information. In *SIGIR*. ACM, 1459–1468.
[25] Guoshuai Zhao, Zhidan Liu, Yulu Chao, and Xueming Qian. 2020. CAPER: Context-Aware Personalized Emoji Recommendation. *IEEE Trans. Knowl. Data Eng.* 32.
[26] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2021. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. In *CIKM*. ACM, 4653–4664.