Towards Interaction-based User Embeddings in Sequential Recommender Models

Marina Ananyeva[1,2], Oleg Lashinin[1], Veronika Ivanova[1], Sergey Kolesnikov[1] and Dmitry I. Ignatov[2]

[1]*Tinkoff, Moscow, Russia*

[2]*National Research University Higher School of Economics, Moscow, Russia*

Abstract

All transductive recommender systems are unable to make predictions for users who were not included in the training sample due to the process of learning user-specific embeddings. In this paper, we propose a new method for replacing identity-based user embeddings in existing sequential models with interaction-based user vectors trained purely on interaction sequences. Such vectors are composed of user interactions using GRU layers with adjusted dropout and maximum item sequence length. This approach is substantially more efficient and does not require retraining when new users appear. Extensive experiments on three open-source datasets demonstrate noticeable improvement in quality metrics for the most of selected state-of-the-art sequential recommender models.

Keywords

sequential recommendation, user-specific embeddings, inductive learning

1. Introduction

Recommender systems are widely used in various online services, such as social networks, e-commerce, and entertainment platforms. These services gather large amounts of sequential data, including the history of interactions between users and items. Some sequential models require learning the ID-based latent user vectors, which are supposed to represent both short-term and long-term preferences based on user-specific information and previous history of interactions. However, there are several drawbacks to this approach.

Firstly, transductive models can recommend items only to users from the training set. The predictions cannot be obtained only based on previous interactions of out-of-sample users because the model's user embeddings depend on users' IDs and additional features (if provided). The problem of making recommendations for new users is solved either by fully retraining the model on the updated data or iterative training on new batches [1]. For industrial purposes, the retraining process on large-scale data is time- and space-consuming, constantly affecting the user coverage with recommendations and the quality of service for new users.

Secondly, storing trainable user vectors may allocate a lot of memory, since the amount of occupied space is usually *O(n)*, where *n* is the number of users. It results in issues associated with model exploitation and storage for a large number of users, which are prevalent in the development of online services. Without the use of user-specific vectors, we don't occupy memory for storing a look-up ID-dependent user embedding matrix, reducing space complexity to *O(1)* by on-the-fly inference of user embedding by the input interactions, which greatly simplifies the operating process.

In this research, we present a method for constructing real-time produced user vectors that is able to overcome the limitations mentioned above. The contributions of this work are summarized as follows:

- We proposed a method of composing user embedding based purely on interaction sequences, which can be employed in architectures of existing recommender sequential models instead of ID-based user-specific embeddings. This approach helps to avoid the need to retrain recommendation models as new interactions emerge. In addition, it does not require storage of per-user embeddings and is therefore more storage efficient and scalable.
- We have comprehensively reviewed existing works in three A and B-ranked conference series (RecSys, CIKM, and SIGIR) in 2019-2021 that use identity-based user embeddings in architectures. This shows that a third of the existing models can be improved using our approach.

The experiments can be reproduced using our open-

ORSUM@ACM RecSys 2022: 5th Workshop on Online Recommender Systems and User Modeling, jointly with the 16th ACM Conference on Recommender Systems, September 23rd, 2022, Seattle, WA, USA

mananeva@hse.ru (M. Ananyeva); o.a.lashinin@tinkoff.ru (O. Lashinin); ext.vvivanova@tinkoff.ru (V. Ivanova); s.s.kolesnikov@tinkoff.ru (S. Kolesnikov); dignatov@hse.ru (D. I. Ignatov)

1234-5678-9012 (M. Ananyeva)

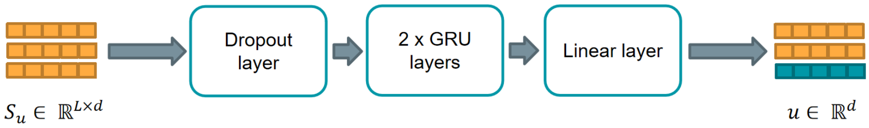


Figure 1: Illustration of the proposed approach. We create user embeddings from its interaction history. Therefore, we do not need to learn ID-based user-specific embeddings. The last known vector of interaction history embeddings is used as user embedding.

Table 1
Full and short papers on sequential recommender models per conference series from 2019 to 2021.

Conference	Articles	Number of models with user vectors	Explained motivation for using user vectors
RecSys'21	[2]	0/1 (0%)	-
RecSys'20	[3, 4, 5, 6, 7, 8]	2/6 (33%) [3, 5]	Long-term preferences, model personalization
RecSys'19	[9]	0/1 (0%)	-
CIKM'21	[10, 11, 12, 13, 14, 15, 16, 17, 18]	3/9 (33%) [11, 12, 18]	Long-term preferences (2 works), friends' impact
CIKM'20	[19, 20, 21, 22]	1/4 (25%) [19]	Short&long-term preferences
CIKM'19	[23, 24, 25, 26]	3/4 (75%) [24, 25, 26]	Short&long-term preferences
SIGIR'21	[27, 28, 29, 30]	0/4 (0%)	-
SIGIR'20	[31, 32, 33]	1/3 (33%) [33]	For ranking score in BPR and GMF
SIGIR'19	[34, 35]	2/2 (100%) [34, 35]	Lifelong user behavior, user-specific representations

source repository[1].

2. Related work

Sequence-based recommender models are commonly used for recommendation tasks on serial data. Most of them are based on recurrent neural networks (RNNs), for instance, GRU4Rec [36], SASRec [37], and SHAN [38]. Additionally, Transformers4Rec [39] is gaining popularity in usage for sequential and session-based tasks.

Some architectures attempt to model temporal decay effects in user interaction history in order to improve the relevance of recommendations. Customer needs as well as both short-term and long-term preferences change over time, which should be taken into account in the predictions. Intuitively, the most recent interactions should have greater weight than older ones in deciding on the next item. Additionally, users may require substitutions or supplements for an already acquired item. These assumptions have been incorporated into the design of the SLRC [40], Chorus [41] and KDA [42] models.

The approaches mentioned above have serious limitations for industrial applications: item recommendations are made based on user-specific embeddings, which can be trained only for users that were included in the training set. In contrast, inductive learning models can provide recommendations for out-of-sample users, who have interactions but are not included in the training process. For instance, Mult-VAE [43] and CF-LGCN-E [44], which is modified version of LightGCN [45] for inductive learning mode, can provide predictions for users outside of the training sample. Nevertheless, the quality of inductive models is often lower than that of transductive ones.

Thus, one of the open challenges for transductive models which show high performance is to overcome the problem of making predictions for out-of-sample users. Additionally, the effect of user-specific embeddings on the quality of recommendations is not yet sufficiently studied. In this research, we propose researching whether we really need user-specific embeddings or if it is better to train ID- and feature-free user vectors based solely on previous item interactions.

3. METHODOLOGY

3.1. The rationale for Using User-specific Vectors

To determine how frequently trainable user-specific vectors are used in existing sequential recommender models and to systematize the reasons for their use, we examined

[1]https://github.com/tinkoff-ai/use_rs

the proceedings of scientific conferences with relevant articles. The summary in **Table 1** shows that our analysis research includes articles that were presented between 2019 and 2021 in three conference series: RecSys, CIKM, and SIGIR. A paper was considered relevant if it proposed a sequential recommendation model, including session-based and POI recommendation tasks, and if their performance was compared to sequential recommender model baselines. As a result, we compiled a list of 34 relevant studies, 12 of which contain applications of user embeddings. According to the authors, the primary purpose for including user vector processing in the proposed methods, which appeared in five studies, was to represent long-term preferences. Other objectives included modeling both short-term and long-term preferences, learning user-specific vectors from mixed representations of all users sharing the same account, and modeling the impact of friends' behavior.

3.2. Initialization of User Vectors

Each sequential model processes the history of users' interactions in order to represent relationships between interactions and then model the user's behavioral patterns [36, 37].

In our work, we investigate the feasibility of employing this method to obtain vector representations that reflect users' interests, as well as how it influences the quality of sequential models. All of the selected models transform the user ID into a low-dimensional real-valued dense vector representation $\mathbf{u} \in \mathbb{R}^d$, where d is the dimension of the user embedding. The embedding is then processed in accordance with the architecture of each model.

Instead of using this technique, we propose to replace the ID-based user embedding initialization with interaction-based user embedding initialization, suggesting that the user ID be discarded as limiting information for efficiency and scalability.

Let $\mathbf{S_u}$ be the input representations of previous interactions $\mathbf{S_u} \in \mathbb{R}^{L \times d}$, where L is the maximum history length. First, we apply a Dropout layer [46] to the matrix $\mathbf{S_u}$. The sequence representation is then processed by GRU layers. Note that our goal is to show that our approach is effective even with a simple recurrent layer like GRU. The use of more advanced layers is left for future improvements. The final step is to use a linear layer to reduce the embedding dimension to its initial size d and take the last known vector of interaction history. As seen on **Figure 1**, we obtain a user embedding $\mathbf{u} \in \mathbb{R}^d$ as the output of successively applied layers: Dropout layer, two GRU layers, and Dense layer, the input of which is a sequence of each user's historical data.

This approach has several significant advantages. First, the space complexity is optimized from $O(n)$ to $O(1)$, where n is the number of users. There is no need to store pre-computed embeddings in a look-up matrix with users' IDs as each vector can be derived on-the-fly from the input interaction sequence using the learned neural network weights. It addresses the scalability issues for commercial applications. Secondly, it can be regarded as a step toward users' privacy and confidentiality, because a user identifier is redundant information, and without using it we can not map it back to personal data. Lastly, our approach allows adapting previously introduced sequential recommender models to inductive learning scenarios, when we can infer the recommendations for the users, who were not included in a training sample.

3.3. Models

In our experiments, we decided to use one of the most popular frameworks for sequential recommendation models - ReChorus[2] and RecBole[3]. For the experimental setup, we have selected state-of-the-art models that have proven themselves in many new research papers as reliable baselines for comparison with new models.

Thus, we selected three models from the ReChorus framework - KDA, Chorus, and SLRC - and two models from RecBole - SHAN and HGN - in order to study how different user vector initialization techniques affect model performance on three open-source datasets.

Two RecBole models have been implemented in ReChorus to ensure a fair comparison of the models.

- **Sequential Hierarchical Attention Network (SHAN)** [38] is a two-layer hierarchical attention network. The attention mechanism is needed to assign altered weights of items for the user to capture the dynamic property, while the hierarchical structure integrates the user's long- and short-term preferences. User embedding vector is used as context information to obtain various weights for different users.
- **Hierarchical Gating Network (HGN)** [47] consists of three parts: feature gating, instance gating, and item-item product modules. The feature gating module allows the adaptive selection of effective latent features based on user interests. At the instance gating module, items that reflect short-term user preferences are selected and passed down to lower layers along with item features. User embedding is used in both feature gating and instance gating modules.
- **Chorus** [41] incorporates the representation of different sequence contexts by knowledge and time-aware item modeling. The constructed temporal kernel functions modify the temporal dynamics of relations by representing two sorts of

[2]https://github.com/THUwangcy/ReChorus
[3]https://recbole.io/

Table 2
Descriptive statistics of datasets.

Dataset	#users	#items	#actions	#density
MovieLens-1M	6,040	3,416	1M	4.84%
Grocery&Gourmet	127,496	41,280	1,1M	0.022%
Electronics	192,403	63,001	1,7M	0.014%

items - substitutes and complements - and allowing relational representations to contribute differentially to the final item embedding. User embeddings are used in both the BPR and GMF approaches for making predictions.

- **Short-Term and Life-Time Repeat Consumption (SLRC)** [40] model uses the Hawkes Process and Collaborative Filtering, which requires learning user embeddings to distinguish between user interests and help explore new items. Considering the lack of recurrent interactions in the Amazon and MovieLens datasets, we use this model to derive substitutive and complementary types of relations between items, as implemented in the SLRC model in Chorus.
- **Knowledge-aware Dynamic Attention (KDA)** [42] takes both item relations and their temporal evolution into account. The core idea of KDA is to aggregate the sequence of interactions into multiple relation-specific embeddings via an attention mechanism. Fourier transform with trainable frequency domain embeddings was used in a novel way to simulate the diverse temporal effects of various relational interactions. User vectors, as well as item vectors and interaction representations, are used in the final ranking score.

Overall, we selected all models stated above and compared the original architectures with the architectures without user-specific vectors, based on our approach of learning only from interaction sequences.

4. EXPERIMENTS

In this section, we introduce our experimental setup and compare the performance of original models with modified ones. Our experiments are designed to answer the following research questions:
RQ1: Does the proposed method have a positive effect on the quality of existing sequential recommender models?
RQ2: How does the maximum sequence length affect the models' performance?

4.1. Datasets

We chose the three datasets most commonly used for sequential recommendation: *MovieLens-1M*[4], *Amazon-Grocery and Gourmet Food* and *Amazon-Electronics*[5]. These open-source datasets have different domains, sizes, and sparsity. They contain user interaction sequences with timestamps and item metadata, including the list of *also view* and *also buy* relations in Amazon datasets and the list of genres in the MovieLens data set. We use a common leave-one-out strategy with 99 negative items, similar to [42]. For SHAN, HGN, and SLRC, we only need user interaction sequences, while Chorus and KDA are based on knowledge graphs, so we use metadata to build them. In Amazon datasets, we simply use the relations of *also view* and *also buy*, provided in the metadata data set, as was done in [41]. We chose the most popular movies of the same genre as the equivalent of *also view* items for the MovieLens data set, and the most popular items in the set of movies that the user has watched right after the ground-truth item as the equivalent of *also buy* items.

4.2. Evaluation Metrics

Hit Ratio (HR@k) and Normalized Discounted Cumulative Gain (NDCG@k) were used as evaluation metrics, where k = [5, 10, 20, 50]. HR@k measures whether at least one ground-truth item appears in the top-k recommendation list, whereas NDCG@k considers both the position and relevance of the item in the recommendation list. The values of NDCG@10 and HR@10 for 5 original and 5 modified models are presented in **Table 3**.

4.3. Experiment Settings

All models were implemented using the PyTorch framework [48]. For a fair comparison, we set the embedding size to 64, batch size to 256, and the maximum history length to 20 for all models and datasets, similar to experiments in [41]. Additionally, we demonstrate the results of experiments for other values of the maximum history length: 10, 30, and 50. Other hyperparameters are dependent on the model and are set to their default values the same as in the original implementations. The tuning of the hyperparameters across all methods and datasets is left for future work.

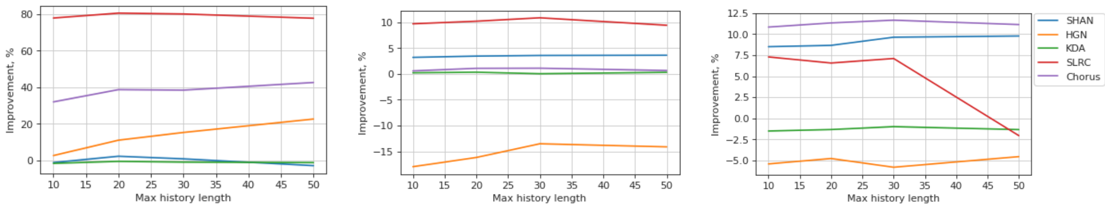
4.4. Baselines

We include two baselines in order to obtain the relative performance of non-sequential methods. Specifically, we include the POP method [49] which is a common non-personal baseline that recommends the most popular

[4]https://grouplens.org/datasets/movielens/1m/
[5]http://jmcauley.ucsd.edu/data/amazon/

Table 3
The results of pairwise comparison of original and modified models. The best result in each pair of sequential models considered is in bold.

	ML-1M		Grocery&Gourmet		Electronics	
	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10
POP	0.2513	0.4575	0.2628	0.4350	0.3087	0.4849
$BPR-MF$	0.4074	0.6844	0.3690	0.5516	0.3444	0.5348
$SHAN$	0.3137	0.5661	0.2480	0.4067	0.2887	0.4418
HGN	0.5233	0.7846	**0.3898**	**0.567**	**0.3845**	**0.5875**
$SLRC$	0.3226	0.5778	0.3334	0.4982	0.3914	0.5569
$Chorus$	0.4309	0.7161	0.4046	0.5862	0.4063	0.5994
KDA	**0.6041**	**0.8386**	0.4442	0.6279	**0.4605**	**0.6733**
$SHAN_{our}$	**0.3209**	**0.5700**	**0.2565**	**0.4306**	**0.3137**	**0.4950**
HGN_{our}	**0.5812**	**0.8086**	0.3268	0.4989	0.3662	0.5666
$SLRC_{our}$	**0.5822**	**0.8091**	**0.3673**	**0.5376**	**0.4171**	**0.6040**
$Chorus_{our}$	**0.5976**	**0.8258**	**0.4089**	**0.5958**	**0.4523**	**0.6527**
KDA_{our}	0.6011	0.8257	**0.4456**	**0.6291**	0.4544	0.6685



(a) MovieLens-1M (b) Amazon Grocery & Gourmet Food (c) Amazon Electronics

Figure 2: Relative change in NDCG@10 of five models on MovieLens-1M, Amazon Grocery&Gourmet Food, and Amazon Electronics Datasets.

items. Additionally, we add a BPR-MF [50] approach that is often adopted as a classic matrix factorization-based method.

4.5. Performance Comparison

As can be seen, **Table 3** shows the recommendation performance of original architectures and modified models on three datasets (**RQ1**). The proposed strategy has a significant impact on model quality across all datasets. For instance, on MovieLens-1M we can see increases in both NDCG@10 and HR@10 for 4 modified models, compared to the original ones: $SHAN_{our}$, HGN_{our}, $SLRC_{our}$ and $Chorus_{our}$, while for KDA_{our} quality remains nearly the same. The quality improvement varies widely, ranging from 1% for $SHAN_{our}$ to 81% for $SLRC_{our}$. We even observe a slight boost in evaluation metrics for the strongest baseline, KDA, on Amazon Grocery&Gourmet Food.

However, the quality of HGN_{our} has deteriorated: NDCG@10 has decreased by 16%. As the authors of HGN observed, the predictions of this model are highly dependent on the last items. When our approach constructs a user vector from long sequences, the impact of the last items may be reduced. On Amazon Electronics we can see decreasing of metric values by 1% for KDA_{our} and by 5% for HGN_{our}, while for $SLRC_{our}$, $SHAN_{our}$ and $Chorus_{our}$ the quality improved in range of 7% to 12%. The overall performance of the four models improved dramatically, but changing user-specific vectors had almost no influence on the KDA_{our} model. According to a research article on KDA, one possible explanation is that the architecture of KDA [42] is not highly sensitive to the presence of user vectors at all. $SLRC_{our}$ demonstrates significant improvement in quality for all datasets. It is explained by the fact that the SLRC algorithm's core component is collaborative filtering (CF), which is good for modeling long-term user preferences. Our technique allows us to evaluate short-term preferences in CF, which the original model may have overlooked. If we consider each of the model-dataset pairs as a separate experiment, our approach dramatically increases the quality metrics in 11 out of 15 cases.

Summing up, comparative experiments on three real-world datasets show the effectiveness of our approach and significant improvement of quality for the majority of examined models. A new method with replaced user-specific embeddings provides a significant relative gain in performance (e.g., $0.6\% - 12.1\%$ for SHAN [38], $1.1\% - 38.7\%$ for Chorus [41], $6.6\% - 80\%$ for SLRC[40]).

Figure 2 shows how the maximum history length influences quality improvement when our approach is applied (**RQ2**). The smaller the maximum sequence length, the better the model captures user short-term preferences, while long-term effects outweigh short-term effects for larger lengths. When the length of the sequence shrinks, the long-term influence of modeling, which is the primary reason for using user embeddings in a model, disappears. As a result, replacing user-specific vectors works effectively for both short ($l = 10$) and long ($l = 50$) sequences.

5. CONCLUSION

In this research, we proposed a method of composing vectors based purely on interaction sequences, which can be employed in architectures of existing recommender sequential models instead of user-specific embeddings. Our method does not require constant retraining of the model as the number of users increases, and is memory-efficient. Extensive experiments on 3 real-world datasets reveal that the majority of evaluated models were improved in quality. Additionally, we studied the relationship between the model's relative improvement and item sequence length when our method is applied. Thus, we suggest researchers experiment with our approach in their studies by using ID-based user-specific embeddings. Our results can open up a new research area for ablation studies on the use of user-specific embeddings in recommender systems. In the future, we are going to apply our approach to more modern models and try more complex architectures than GRU. In addition, it is essential to investigate how high-quality and stable this approach is with an extremely small number of user interactions.

Acknowledgements

This research was supported by the Tinkoff Laboratory and the Laboratory for Model and Methods of Computational Pragmatics at the National Research University Higher School of Economics (HSE). The contribution of Dmitry I. Ignatov to the article was done within the framework of the HSE University Basic Research Program.

References

[1] Y. Zhang, F. Feng, C. Wang, X. He, M. Wang, Y. Li, Y. Zhang, How to retrain recommender system? a sequential meta-learning method, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1479–1488.

[2] W. Song, S. Wang, Y. Wang, S. Wang, Next-item recommendations in short sessions, in: Fifteenth ACM Conference on Recommender Systems, 2021, pp. 282–291.

[3] C. Hansen, C. Hansen, L. Maystre, R. Mehrotra, B. Brost, F. Tomasi, M. Lalmas, Contextual and sequential user embeddings for large-scale music recommendation, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 53–62.

[4] J. Lin, W. Pan, Z. Ming, Fissa: fusing item similarity models with self-attention networks for sequential recommendation, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 130–139.

[5] L. Wu, S. Li, C.-J. Hsieh, J. Sharpnack, Sse-pt: Sequential recommendation via personalized transformer, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 328–337.

[6] F. Mi, X. Lin, B. Faltings, Ader: Adaptively distilled exemplar replay towards continual learning for session-based recommendation, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 408–413.

[7] S. Liu, Y. Zheng, Long-tail session-based recommendation, in: Fourteenth ACM conference on recommender systems, 2020, pp. 509–514.

[8] S. M. Cho, E. Park, S. Yoo, Meantime: Mixture of attention mechanisms with multi-temporal embeddings for sequential recommendation, in: Fourteenth ACM Conference on Recommender Systems, 2020, pp. 515–520.

[9] Z. Zhao, L. Hong, L. Wei, J. Chen, A. Nath, S. Andrews, A. Kumthekar, M. Sathiamoorthy, X. Yi, E. Chi, Recommending what video to watch next: a multitask ranking system, in: Proceedings of the 13th ACM Conference on Recommender Systems, 2019, pp. 43–51.

[10] Q. Wu, C. Yang, S. Yu, X. Gao, G. Chen, Seq2bubbles: Region-based embedding learning for user behaviors in sequential recommenders, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 2160–2169.

[11] Z. Fan, Z. Liu, J. Zhang, Y. Xiong, L. Zheng, P. S. Yu, Continuous-time sequential recommendation with temporal graph collaborative transformer, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 433–442.

[12] Y. Li, Y. Ding, B. Chen, X. Xin, Y. Wang, Y. Shi, R. Tang, D. Wang, Extracting attentive social temporal excitation for sequential recommendation, arXiv preprint arXiv:2109.13539 (2021).

[13] Y. Li, T. Chen, P.-F. Zhang, H. Yin, Lightweight self-attentive sequential recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 967–977.

[14] K. Hu, L. Li, Q. Xie, J. Liu, X. Tao, What is next when sequential prediction meets implicitly hard interaction?, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 710–719.

[15] Z. Fan, Z. Liu, S. Wang, L. Zheng, P. S. Yu, Modeling sequences as distributions with uncertainty for sequential recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 3019–3023.

[16] Z. He, H. Zhao, Z. Lin, Z. Wang, A. Kale, J. Mcauley, Locker: Locally constrained self-attentive sequential recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 3088–3092.

[17] Q. Cui, C. Zhang, Y. Zhang, J. Wang, M. Cai, St-pil: Spatial-temporal periodic interest learning for next point-of-interest recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 2960–2964.

[18] Z. Chen, W. Zhang, J. Yan, G. Wang, J. Wang, Learning dual dynamic representations on time-sliced user-item interaction graphs for sequential recommendation, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 231–240.

[19] W. Ji, K. Wang, X. Wang, T. Chen, A. Cristea, Sequential recommender via time-aware attentive memory network, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 565–574.

[20] W. Chen, P. Ren, F. Cai, F. Sun, M. de Rijke, Improving end-to-end sequential recommendations with intent-aware diversification, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 175–184.

[21] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, J.-R. Wen, S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1893–1902.

[22] M. M. Tanjim, Dynamicrec: A dynamic convolutional network for next item recommendation, in: Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM-2020), 2020.

[23] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 1441–1450.

[24] A. Yan, S. Cheng, W.-C. Kang, M. Wan, J. McAuley, Cosrec: 2d convolutional neural networks for sequential recommendation, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2173–2176.

[25] Y. Wu, K. Li, G. Zhao, X. Qian, Long-and short-term preference learning for next poi recommendation, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 2301–2304.

[26] F. Lv, T. Jin, C. Yu, F. Sun, Q. Lin, K. Yang, W. Ng, Sdm: Sequential deep matching model for online large-scale recommender system, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2635–2643.

[27] R. Cai, J. Wu, A. San, C. Wang, H. Wang, Category-aware collaborative sequential recommendation, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 388–397.

[28] Z. Liu, Z. Fan, Y. Wang, P. S. Yu, Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer, in: Proceedings of the 44th international ACM SIGIR conference on Research and development in information retrieval, 2021, pp. 1608–1612.

[29] X. Yuan, D. Duan, L. Tong, L. Shi, C. Zhang, Icai-sr: Item categorical attribute integrated sequential recommendation, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 1687–1691.

[30] X. Fan, Z. Liu, J. Lian, W. X. Zhao, X. Xie, J.-R. Wen, Lighter and better: low-rank decomposed self-attention networks for next-item recommendation, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 1733–1737.

[31] R. Ren, Z. Liu, Y. Li, W. X. Zhao, H. Wang, B. Ding, J.-R. Wen, Sequential recommendation with self-attentive multi-adversarial network, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 89–98.

[32] L. Zheng, N. Guo, W. Chen, J. Yu, D. Jiang, Sentiment-guided sequential recommendation, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1957–1960.

[33] C. Wang, M. Zhang, W. Ma, Y. Liu, S. Ma, Make it a chorus: knowledge-and time-aware item modeling for sequential recommendation, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 109–118.

[34] K. Ren, J. Qin, Y. Fang, W. Zhang, L. Zheng, W. Bian, G. Zhou, J. Xu, Y. Yu, X. Zhu, et al., Lifelong sequential modeling with personalized memorization for user response prediction, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 565–574.

[35] M. Ma, P. Ren, Y. Lin, Z. Chen, J. Ma, M. d. Rijke, π-net: A parallel information-sharing network for shared-account cross-domain sequential recommendations, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 685–694.

[36] B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Tikk, Session-based recommendations with recurrent neural networks, arXiv preprint arXiv:1511.06939 (2015).

[37] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, in: 2018 IEEE International Conference on Data Mining (ICDM), IEEE, 2018, pp. 197–206.

[38] H. Ying, F. Zhuang, F. Zhang, Y. Liu, G. Xu, X. Xie, H. Xiong, J. Wu, Sequential recommender system based on hierarchical attention network, in: IJCAI International Joint Conference on Artificial Intelligence, 2018.

[39] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, E. Oldridge, Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation, in: Fifteenth ACM Conference on Recommender Systems, 2021, pp. 143–153.

[40] C. Wang, M. Zhang, W. Ma, Y. Liu, S. Ma, Modeling item-specific temporal dynamics of repeat consumption for recommender systems, in: The World Wide Web Conference, 2019, pp. 1977–1987.

[41] C. Wang, M. Zhang, W. Ma, Y. Liu, S. Ma, Make it a chorus: knowledge-and time-aware item modeling for sequential recommendation, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 109–118.

[42] C. Wang, W. Ma, M. Zhang, C. Chen, Y. Liu, S. Ma, Toward dynamic user intention: Temporal evolutionary effects of item relations in sequential recommendation, ACM Transactions on Information Systems (TOIS) 39 (2020) 1–33.

[43] D. Liang, R. G. Krishnan, M. D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of the 2018 world wide web conference, 2018, pp. 689–698.

[44] R. Ragesh, S. Sellamanickam, V. Lingam, A. Iyer, R. Bairi, User embedding based neighborhood aggregation method for inductive recommendation, arXiv preprint arXiv:2102.07575 (2021).

[45] Y. Shen, Y. Wu, Y. Zhang, C. Shan, J. Zhang, B. K. Letaief, D. Li, How powerful is graph convolution for recommendation?, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021, pp. 1619–1629.

[46] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The journal of machine learning research 15 (2014) 1929–1958.

[47] C. Ma, P. Kang, X. Liu, Hierarchical gating networks for sequential recommendation, in: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, 2019, pp. 825–833.

[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[49] N. Neophytou, B. Mitra, C. Stinson, Revisiting popularity and demographic biases in recommender evaluation and effectiveness, in: European Conference on Information Retrieval, Springer, 2022, pp. 641–654.

[50] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, arXiv preprint arXiv:1205.2618 (2012).