Clustering based Behavior Sampling with Long Sequential Data for CTR Prediction

Yuren Zhang
Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
yr160698@mail.ustc.edu.cn

Enhong Chen*
Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
cheneh@ustc.edu.cn

Binbin Jin
Huawei Cloud Computing Technologies Co., Ltd.
Hangzhou, Zhejiang, China
jinbinbin1@huawei.com

Hao Wang
Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
wanghao3@ustc.edu.cn

Min Hou, Wei Huang
Anhui Province Key Laboratory of Big Data Analysis and Application, School of Data Science, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
{minho,ustc0411}@mail.ustc.edu.cn

Runlong Yu
Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, Anhui, China
yrunl@mail.ustc.edu.cn

ABSTRACT

Click-through rate (CTR) prediction is fundamental in many industrial applications, such as online advertising and recommender systems. With the development of the online platforms, the sequential user behaviors grow rapidly, bringing us great opportunity to better understand user preferences. However, it is extremely challenging for existing sequential models to effectively utilize the entire behavior history of each user. First, there is a lot of noise in such long histories, which can seriously hurt the prediction performance. Second, feeding the long behavior sequence directly results in infeasible inference time and storage cost. In order to tackle these challenges, in this paper we propose a novel framework, which we name as User Behavior Clustering Sampling (UBCS). In UBCS, short sub-sequences will be obtained from the whole user history sequence with two cascaded modules: (i) Behavior Sampling module samples short sequences related to candidate items using a novel sampling method which takes relevance and temporal information into consideration; (ii) Item Clustering module clusters items into a small number of cluster centroids, mitigating the impact of noise and improving efficiency. Then, the sampled short sub-sequences will be fed into the CTR prediction module for efficient prediction. Moreover, we conduct a self-supervised consistency pre-training task to extract user persona preference and optimize the sampling module effectively. Experiments on real-world datasets demonstrate the superiority and efficiency of our proposed framework.

*Corresponding author.

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3531829

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**; *Clustering and classification.*

KEYWORDS

CTR Prediction; Information Retrieval; Long Sequential User Behavior Modeling

ACM Reference Format:
Yuren Zhang, Enhong Chen, Binbin Jin, Hao Wang, Min Hou, Wei Huang, and Runlong Yu. 2022. Clustering based Behavior Sampling with Long Sequential Data for CTR Prediction. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3477495.3531829

1 INTRODUCTION

Click-through rate (CTR) prediction is fundamental in many industrial applications, which aims to estimate CTR accurately. Since it directly impacts financial revenues, CTR prediction has received much attention from both research and industry community [8, 18, 21, 23, 24]. With the development of the online platforms, the sequential user behaviors grow rapidly. For example, there are 23% of users have more than 1,000 behaviors during six months on the Chinese largest e-commerce shopping platform Taobao [13, 14]. As rich user behaviors have been proven valuable for CTR prediction,

1、无关内容多，噪声大；
2、缺乏动态性；
3、计算和存储消耗大

a crucial demand asks to find ways to model with long sequential user behaviors accurately and efficiently.

Along this line, various approaches have been proposed to model long sequential user behavior data. Traditional frameworks [1, 2] often truncates the whole behavior sequence and just uses the most recent several behaviors, which leads to a problem that sequential patterns such as periodicity or long-term dependency in far back history cannot be effectively utilized. Methods like [16, 19] transform the long sequence into well-designed structures to reduce the sequence length, but also bring information loss and difficulties in practice. Memory-based [12, 15] and search-based [13, 14] methods make some breakthroughs in using longer sequences, but at the same time, a lot of irrelevant behaviors and noises could be introduced and may lead to infeasible inference time and space complexity [14]. For example, memory-based models like MIMN encodes all unfiltered user historical behaviors into a fixed-length memory which makes it hard to capture diverse long-term interest [13]. As for search-based models, it's still challenging for existing search strategies to fully mine the relevance of user behaviors and capture the precise user preference.

Therefore, there are still many unique challenges inherent in designing an effective and efficient solution to model long sequential user behaviors. On one hand, there is a lot of noise in such long histories, which can seriously hurt the prediction performance. On the other hand, feeding the long behavior sequence directly results in infeasible inference time and storage cost. These two aspects make it hard to explore long sequential user behaviors exhaustively with current CTR prediction models.

In order to address the above issues and recommend the items which users are most likely to be interested in, an advisable solution is to efficiently pick out samples that are most relevant for each CTR prediction target from the long sequence. Based on this idea and inspired by the successful application of sampling in recommendation tasks [6, 10], we propose a sampling based framework, dubbed UBCS, to model with long sequential user behaviors and improve the performance of CTR prediction. In UBCS, short sub-sequences containing most related behaviors for target item will be obtained by two cascaded modules: (i) Behavior Sampling module adopts a new sampling strategy to sample useful user behaviors from long sequential historical data which takes relevance and time information into consideration; (ii) Item Clustering module clusters items into a small number of cluster centroids, mitigating the impact of noise and improving efficiency. Then, the CTR prediction module can use the related sub-sequences to improve the prediction performance. To ensure the sampling module is well-trained, we borrow ideas from Contrastive Learning and design a self-supervised pretraining task to extract user persona preference and optimize the sampling module effectively.

The contributions of the paper can be summarized in three-fold:

- We propose a novel framework named UBCS, to effectively and efficiently utilize the entire behavior history of each user and make accurate CTR prediction.
- We design a new sampling strategy to capture useful user behaviors from long sequential historical data which takes relevance and time information into account and it is easy to extend. To ensure the effectiveness and time efficiency of sampling, we conduct a self-supervised consistency pretraining task to efficiently train the sampling module and adopt clustering method to speed up the sampling process.
- We conduct extensive experiments on two public datasets and compare our framework with several state-of-the-art sequential models, indicating both the effectiveness and efficiency of our proposed framework.

2 USER BEHAVIOR CLUSTERING SAMPLING FRAMEWORK

In this section, we introduce our UBCS. As shown in Figure 1, UBCS is composed of three modules, which are (i) the Behavior Sampling module for sampling short sequences from long sequential user behavior data, (ii) the Item Clustering module for clustering items into few cluster centroids to speed up the sampling process and (iii) the CTR prediction module for accurate prediction. By this framework, a limited number of user behaviors which are most relevant for each candidate items can be obtained and fed into CTR prediction model to make efficient prediction. We will describe the three modules in detail.

2.1 Behavior Sampling Module

Long-term behavior sequences contain not only rich information of user interests, but also lots of irrelevant behaviors and noise. To capture user preferences and eliminate irrelevant behaviors, we design a behavior sampling module for extracting useful sub-sequences of length K from the user history sequence $S_u = [b_1^u, b_2^u, ..., b_N^u]$, where b_i^u is the i-th behavior of user u sorting by timestamp and N is the length of the whole behavior sequence.

Traditional sampling methods like Popularity Sampling [10] usually perform sampling based on specific statistic features and lack the ability to mine the deep interaction between items. Thus, we introduce a new sampling strategy for better information extraction.

Relevance Extraction. Since the relation between two items is an essential factor to model in the sequential tasks and has been widely studied in many years [11, 22], we take it into account and calculate relevant score r_i between each item in the long sequence and the candidate item.

Formally, given the user history sequence S_u and a candidate item i_t, we first develop a encoder $E(\cdot)$ to better extract items' information. Specifically, we build two embedding look up layer to the item b_i and item's category c_i. And then concatenate the output of them to get the final item embedding e_i, the process can be formulated as $e_i = E(b_i, c_i)$. After that, the inner product is performed to get the relevant score between each user previous item embedding e_i^u and the candidate item embedding e_t:

$$r_i = (W_a e_i^u) \cdot (W_b e_t)^T, \quad (1)$$

where W_a and W_b denote the parameters of weight.

Time-aware Sampling. Furthermore, as the personalized time intervals are proven of great value [9, 20], we also view time intervals as one kind of relation between two items and make use of them. From the responding time sequence $T^u = [t_1^u, t_2^u, ..., t_N^u]$, we denote the scaled time intervals between item i and candidate item c as $l_i = \frac{|t_i - t_t|}{t_{min}^u}$, where t_{min}^u is the minimum time interval in

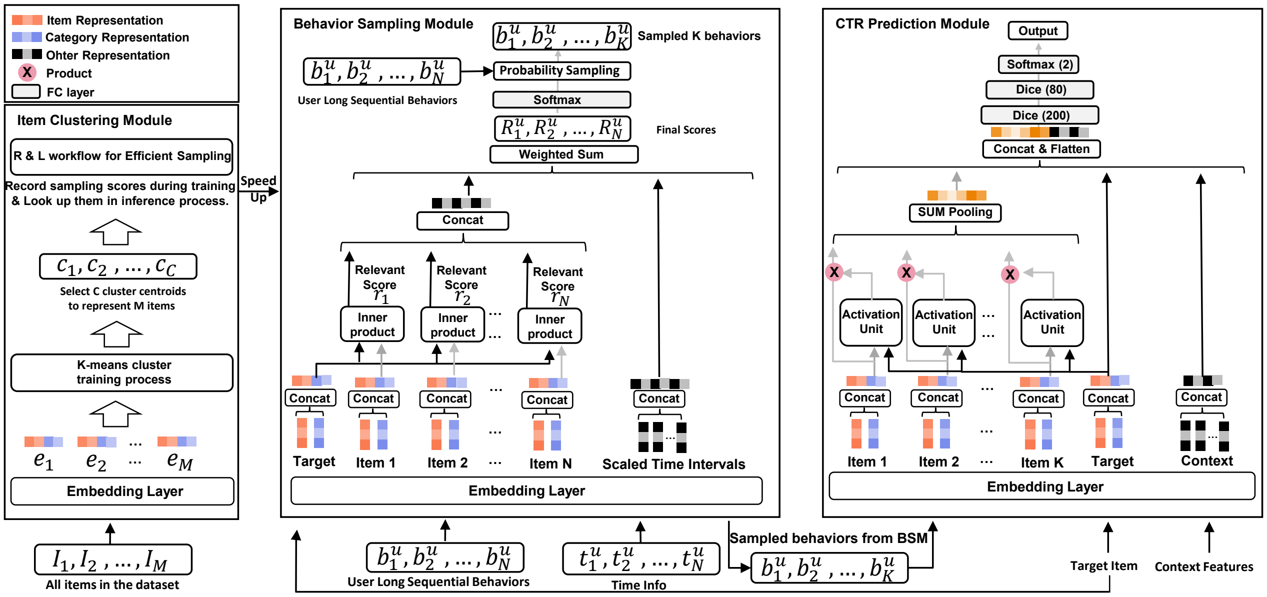

Figure 1: The structure of User Behavior Clustering Sampling framework.

T^u. Then, we can combine r_i and l_i to get the final relevant score:

$$R_i = r_i + W_c \cdot E(l_i), \tag{2}$$

where W_c is the combination weight.

In this way, the sequence of relevant scores $R_c^u = [R_1^u, R_2^u, ..., R_N^u]$ can be obtained. After that, we utilize a classic Softmax method to get the sampling weight W_i^u of each behavior b_i^u:

$$W_i^u = \frac{exp(R_i)}{\sum_{i=1}^{N} exp(R_i)}. \tag{3}$$

Based on W_i^u, we adopt Probability Sampling to sample short subsequences with fix-length K which contain relevant user behaviors w.r.t each candidate items. Subsequently, the sampled subsequences will be fed into the CTR prediction module to help improve the prediction performance.

2.2 CTR Prediction Module

We note that UBCS is compatible for most of CTR prediction model, since the prediction module takes the sampled short sub-sequences as input instead of dealing with user lifelong sequences directly. To further capture the precise user interest, we choose Deep Interest Network (DIN) [26] as the prediction model which has better efficiency and robustness than most Embedding & MLP methods.

Formally, for each sample $s = (u, c, g, y)$ containing an user u, a context c, a candidate item g and label y, the CTR prediction model first embeds features of u, c and g to obtain the feature vector x, and then use a MLP to outputs $p(x)$, where $p(x) \in [0, 1]$ reflects the probability of sample s belonging to positive class. Then, we train the model under the Cross-entropy loss function defined as:

$$\mathcal{L}^{CTR} = -\frac{1}{D} \sum_{(\boldsymbol{x},y)\in B} \left(y \log p\left(\boldsymbol{x}\right) + (1-y)\log\left(1-p\left(\boldsymbol{x}\right)\right)\right), \tag{4}$$

where B is the training set of size D.

Self-Supervised Pre-Training Method. For the long sequential user behaviors modeling, the input user behavior feature sequence is too long to make the original supervised learning labels become relatively sparse, and therefore make it hard for the model to get enough training signals. To alleviate this issue and further making the sampled sequence more effective, we design a contrastive learning [3, 4] training objective at the pre-training stage by constructing additional self-supervised signals relying on the inherent character of sequential user behavior data.

Intuitively, the same user's preferences for items are basically consistent. We leverage this consistent by maximize the mutual information between the two sampled sub-sequence of the same user. We denote the representation of two sampled sub-sequence of the same user as $\boldsymbol{p}$ and $\boldsymbol{q}$, and then formulate the contrastive loss function as:

$$\mathcal{L}^C = -\underset{\mathcal{P}_l}{\mathrm{E}}\left[\log \frac{D_\omega\left(\boldsymbol{p},\boldsymbol{q}\right)}{D_\omega\left(\boldsymbol{p},\boldsymbol{q}\right) + \sum_{\tilde{\boldsymbol{q}}\in\hat{\mathrm{Q}}} D_\omega\left(\boldsymbol{p},\tilde{\boldsymbol{q}}\right)}\right] \tag{5}$$

where $\mathcal{P}_l$ represent the joint distribution of the user sub-sequences. i.e., $(\boldsymbol{p},\boldsymbol{q}) \sim \mathcal{P}_l$, $\tilde{\boldsymbol{q}}$ denotes the negative sample randomly sampled from the different user's sub-sequence within one mini-batch. $D_\omega(\cdot,\cdot)$ is the discriminator parameterized by ω, we define it in the form of log-bilinear:

$$D_\omega\left(\boldsymbol{p},\boldsymbol{q}\right) = \exp\left(\boldsymbol{p}^\top \cdot \boldsymbol{W} \cdot \boldsymbol{q}\right), \tag{6}$$

The loss in Equation (5) is the categorical cross-entropy of classifying the positive sample correctly. Following this manner, our UBCS first performs pre-training under the loss function $\mathcal{L}^C$, and then use the Sum of squares of distances to train the Item Clustering module and optimize the prediction module under $\mathcal{L}^{CTR}$.

Table 1: The statistics of two datasets after preprocessing.

Dataset	Users	Items	Interaction	Avg. Seq. Length
Books	22,377	383,646	2,237,700	100
Cloth	10,483	256,726	524,150	50

2.3 Item Clustering Module

In practice, calculating relevant score R_i^u for each behavior b_i^u w.r.t each candidate items is unacceptable due to the large time complexity $O(UMT)$, where U and M reflect to the number of users and candidate items. To tackle this challenge, we introduce item clustering method and design a sampling workflow to meet the requirement of the inference latency.

Item Clustering. In orter to optimize the time complexity $O(UMT)$, an advisable method is to select a limited amounts of representative items for performing calculation instead of using the whole candidate items set. Hence, we conduct Clustering to find representative items which are the cluster centroids after the convergence of training. Among plenty of Clustering methods, we adopt the widely used K-means for simplicity. The loss function of K-means is the Sum of squares of distances defined as:

$$\mathcal{L}^{KM} = \sum_{i=1}^{M} ||\boldsymbol{x_i} - \boldsymbol{c_j}||_2^2, \tag{7}$$

where M is the amounts of items, $\boldsymbol{x_i}$ is the i-th item and $\boldsymbol{c_j}$ is the corresponding cluster centroid of the j-th cluster.

Thanks to Item Clustering, the time complexity can be reduced to $O(UCT)$, where C is the number of cluster centrioids which is significantly smaller than M. Though the clustering method may lose some detail information of items, it also removes noise and reduces the impact of sparsity. We will conduct further analysis based on experimental result in section 3.

Sampling workflow with Item Clustering. In particular, we design a Restore and Look-up (R & L) workflow to further accelerate the sampling process. Before every training epoch, we re-train the Item Clustering module to ensure that the cluster centroids is representative. Then, we build a set to storage all the relevant scores R^u for each behavior of each user w.r.t each cluster centroids, most of which will be obtained during training. As a result, when inference, most of the relevant scores can be directly found by a simple look-up operation, greatly reducing the inference latency.

Finally, the model can be trained using stochastic Gradient Descent (SGD) algorithm.

3 EXPERIMENTS

In this section, we evaluate the model performance of the proposed UBCS. We describe the experimental settings and experimental results in detail.

3.1 Experimental settings

Datasets. Model comparisons are conducted on two public datasets of users online behaviors: the Books subset and the Cloth subset of the Amazon dataset which is often used as benchmark dataset. The statistics of the datasets can be found in Table 1. Following [13], we regard reviews as one kind of interaction behaviors and sort the reviews from one user by time. During pre-processing, we select users whose historical sequences are long enough and reserve recent behaviors. We split the recent 10% samples for testing, 10% samples for validation and 80% samples for training. These pre-processing methods have been widely used in related works.

Table 2: Experimental results on public Amazon datasets.

Model	Books		Cloth	
	AUC	LL	AUC	LL
SASRec	0.8743	0.1338	0.6967	0.1619
DIN	0.8958	0.1275	0.6758	0.1662
DIEN	0.8902	0.1306	0.7014	0.1542
MIMN	0.8681	0.1429	0.7055	0.1284
HPMN	0.8897	0.1355	0.7199	0.1198
SIM-hard	<u>0.9035</u>	<u>0.1183</u>	<u>0.7226</u>	<u>0.1146</u>
UBCS-SS	0.8649	0.1603	0.6567	0.1854
UBCS w/o PT	0.9031	0.1278	0.6802	0.1581
UBCS w/o IC	**0.9226**	**0.1097**	0.7488	0.1052
UBCS	0.9201	0.1138	**0.7520**	**0.1033**
Impr	2.1%	7.3%	4.1%	9.8 %

Metrics. In our experiments, we adopt the AUC (Area Under ROC) score as one of our evaluation metrics, which reflects the pairwise ranking performance between click and non-click samples. The other metric is log loss which measures the overall likelihood of the test data. These two metrics have been widely used for the CTR prediction [5, 15].

Compared Methods. We compare our framework with 6 state-of-the-art baselines from both sequential CTR prediction models and recommendation scenarios, including Self-Attentive Sequential Recommendation (SASRec) [7], Deep Interest Network (DIN) [26], Deep Interest Evolution Network (DIEN) [25], Multi-channel user Interest Memory Network (MIMN) [12], Hierarchical Periodic Memory Network (HPMN) [15], and Search-based Interest Model with hard-search (SIM-hard) [13]. Also, UBCS with stochastic sampling (UBCS-SS), UBCS without Pre-train (UBCS w/o PT) and UBCS without Item Clustering (UBCS w/o IC) are introduced to perform an ablation study and analyze the impacts of our sampling strategy, pre-train method and item clustering component.

In order to compare different methods fairly, we take the same experiment setup and hyperparameters with related works. Considering that different models use different ways to deal with long sequences, we feed the full-length sequences to attention-based models (SASRec, DIN and DIEN) and memory-based models (MIMN and HPMN) which are 100 and 50 resperctively on two datasets. And for search-based model (SIM) and our UBCS, the sampled behaviors are limited to 20. The number of cluster centrios are set to 300 and 70 according to the amount of items in the dataset. All models are trained using Adam optimizer with a batch size of 1024. All models are implemented with the help of DeepCTR[17]. We repeat the experiments five times and report the average results.

3.2 Experimental results

Table 2 presents the experimental results of 6 strong baselines and our framework. From the table, we could draw the following conclusions. (i) Compared with attention-based methods and memory-based methods which utilize the whole sequence, search-based model SIM has better performance regarding AUC and log-loss even though it uses much fewer behaviors, demonstrating the importance of extracting the most relevant short behavior sequence from the long sequence to eliminate the irrelevant behaviors and noise. (ii) Our UBCS performs best among all the baselines significantly, which clearly indicates the effectiveness of UBCS. In particular, on the Books and the Cloth datasets, UBCS has 2.1%/4.1% gain on AUC and 7.3%/9.8% gain on log-loss relative to the SIM with hard-search, which shows that our sampling framework has a better ability to capture diverse long-term interest. (iii) By comparing the three variants of UBCS, we could examine the efficiency of each module. First, UBCS with stochastic sampling is significantly surpassed by all the baselines and our UBCS, verifying that our proposed sampling strategy is effective and necessary. Second, UBCS without Pre-train performs closely to DIN which is the base prediction module of UBCS, as it fails to effectively optimize the sampling module and makes its effect limited. Third, UBCS without Item Clustering achieves similar performance to UBCS, firmly validating the fact that appropriate clustering method will not hurt the performance of the model, since it not only eliminates some detailed information of items but also removes noise. Moreover, Item Clustering method is sufficient to speed up the sampling process and help reach good inference time.

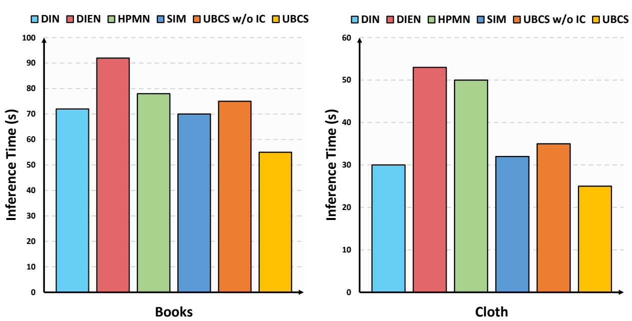

Figure 2: Inference time on test set of two datasets.

To illustrate the practical value of our framework, we plot the inference time comparisons among different models (i.e., DIN, DIEN, HPMN, SIM, UBCS w/o IC and UBCS). As shown in Figure 2, the inference time of SIM and UBCS is relatively shorter, since they use fewer behaviors than other models and reduce the computation costs. Besides, UBCS without Item Clustering takes much more inference time than UBCS and even performs worse than DIN which accepts full-length sequences, because it needs to pay extra time to play sampling. Once equipped with the Item Clustering module, the performance of UBCS increases significantly and achieves the best, which has 21.4%/16.7% gain on inference time relative to the fastest baseline, confirming the practicability of our framework.

4 CONCLUSION

In this paper, we proposed the UBCS framework to model with long sequential user behaviors, which utilized a new sampling strategy to efficiently sample useful sub-sequences from user history sequence related to candidate items to improve the CTR prediction performance. In particular, we conducted a self-supervised consistency pre-training task to well optimize the sampling module and adopted a clustering module to speed up the sampling process. Our extensive experiments on two public datasets presented the superiority and efficiency of UBCS. In time-sensitive online CTR prediction scenarios, our framework can be effectively applied.

ACKNOWLEDGMENTS

This research was partially supported by grants from the National Key R&D Program of China (Grant No. 2021YFF0901005), and the National Natural Science Foundation of China (Grant No. U20A20229).

REFERENCES

[1] Heng Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, and Mustafa and Ispir. 2016. Wide & Deep Learning for Recommender Systems. *ACM* (2016).

[2] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. (2017).

[3] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[4] Min Hou, Chang Xu, Yang Liu, Weiqing Liu, Jiang Bian, Le Wu, Zhi Li, Enhong Chen, and Tie-Yan Liu. 2021. Stock Trend Prediction with Multi-granularity Data: A Contrastive Learning Approach with Adaptive Fusion. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 700–709.

[5] Jianqiang Huang, Ke Hu, Qingtao Tang, Mingjian Chen, Yi Qi, Jia Cheng, and Jun Lei. 2021. Deep Position-wise Interaction Network for CTR Prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1885–1889.

[6] B. Jin, D. Lian, Z. Liu, Q. Liu, J. Ma, X. Xie, and E. Chen. 2020. Sampling-Decomposable Generative Adversarial Recommender. (2020).

[7] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[8] F. Li, Z. Chen, P. Wang, Y. Ren, and X. Zhu. 2021. Graph Intention Network for Click-through Rate Prediction in Sponsored Search. (2021).

[9] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*. 322–330.

[10] Defu Lian, Qi Liu, and Enhong Chen. 2020. Personalized ranking with importance sampling. In *Proceedings of The Web Conference 2020*. 1093–1103.

[11] Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 825–833.

[12] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on long sequential user behavior modeling for click-through rate prediction. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2671–2679.

[13] Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. 2020. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2685–2692.

[14] Jiarui Qin, Weinan Zhang, Xin Wu, Jiarui Jin, Yuchen Fang, and Yong Yu. 2020. User behavior retrieval for click-through rate prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2347–2356.

[15] Kan Ren, Jiarui Qin, Yuchen Fang, Weinan Zhang, Lei Zheng, Weijie Bian, Guorui Zhou, Jian Xu, Yong Yu, Xiaoqiang Zhu, et al. 2019. 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*. 565–574.

[16] Qiu Ruihong, Li Jingjing, Huang Zi, and Yin Hongzhi. 2019. Rethinking the Item Order in Session-based Recommendation with Graph Neural Networks. *ACM* (2019).

[17] Weichen Shen. 2017. DeepCTR: Easy-to-use,Modular and Extendible package of deep-learning based CTR models. https://github.com/shenweichen/deepctr.

[18] Hao Wang, Defu Lian, Hanghang Tong, Qi Liu, Zhenya Huang, and Enhong Chen. 2021. HyperSoRec: Exploiting Hyperbolic User and Item Representations with Multiple Aspects for Social-aware Recommendation. *ACM Transactions on Information Systems (TOIS)* (2021).

[19] Likang Wu, Zhi Li, Hongke Zhao, Zhen Pan, Qi Liu, and Enhong Chen. 2020. Estimating early fundraising performance of innovations via graph-based market environment model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 6396–6403.

[20] Wenwen Ye, Shuaiqiang Wang, Xu Chen, Xuepeng Wang, Zheng Qin, and Dawei Yin. 2020. Time matters: Sequential recommendation with complex temporal information. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1459–1468.

[21] Runlong Yu, Yuyang Ye, Qi Liu, Zihan Wang, Chunfeng Yang, Yucheng Hu, and Enhong Chen. 2021. Xcrossnet: Feature structure-oriented learning for click-through rate prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 436–447.

[22] Xu Yuan, Hongshen Chen, Yonghao Song, Xiaofang Zhao, Zhuoye Ding, Zhen He, and Bo Long. 2021. Improving Sequential Recommendation Consistency with Self-Supervised Imitation. *arXiv preprint arXiv:2106.14031* (2021).

[23] Kai Zhang, Hao Qian, Qing Cui, Qi Liu, Longfei Li, Jun Zhou, Jianhui Ma, and Enhong Chen. 2021. Multi-interactive attention network for fine-grained feature learning in ctr prediction. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 984–992.

[24] Pu Zhao, Chuan Luo, Cheng Zhou, Bo Qiao, Jiale He, Liangjie Zhang, and Qingwei Lin. 2021. RLNF: Reinforcement Learning based Noise Filtering for Click-Through Rate Prediction. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[25] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.

[26] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1059–1068.