A Deep Behavior Path Matching Network for Click-Through Rate Prediction

Jian Dong[1], Yisong Yu[2,3], Yapeng Zhang[1], Yimin Lv[2,3], Shuli Wang[1], Beihong Jin[2,3*], Yongkang Wang[1*], Xingxing Wang[1], Dong Wang[1]

[1]Meituan
[2]Institute of Software, Chinese Academy of Sciences
[3]University of Chinese Academy of Sciences, Beijing, China
Dongjian03@meituan.com,Beihong@iscas.ac.cn,Wangyongkang03@meituan.com

ABSTRACT

User behaviors on an e-commerce app not only contain different kinds of feedback on items but also sometimes imply the cognitive clue of the user's decision-making. For understanding the psychological procedure behind user decisions, we present the behavior path and propose to match the user's current behavior path with historical behavior paths to predict user behaviors on the app. Further, we design a deep neural network for behavior path matching and solve three difficulties in modeling behavior paths: sparsity, noise interference, and accurate matching of behavior paths. In particular, we leverage contrastive learning to augment user behavior paths, provide behavior path self-activation to alleviate the effect of noise, and adopt a two-level matching mechanism to identify the most appropriate candidate. Our model shows excellent performance on two real-world datasets, outperforming the state-of-the-art CTR model. Moreover, our model has been deployed on the Meituan food delivery platform and has accumulated 1.6% improvement in CTR and 1.8% improvement in advertising revenue.

CCS CONCEPTS

• **Information systems → Recommender systems**.

KEYWORDS

Click-Through Rate Prediction, User Behavior Modeling

1 INTRODUCTION

Meituan Takeout APP is an app for catering and retail. Through the app, users can browse and choose the POIs (Points Of Interest, such as restaurants, food stores, and cafes) and place orders for food that will be fast delivered to users. The app is expected to understand the psychology behind user decisions and push relevant candidates to users, thus increasing the click-through rate (CTR) and further transaction volume and advertising revenue.

We note that user behaviors on an app are an important manifestation of the user's decision-making psychology. However, although some existing models for CTR prediction have analyzed user behaviors, from the perspective of long sequences or multiple kinds of behaviors, they adopt the point-to-point activation of the candidate and individual behavior in the historical behavior sequence, without taking into consideration the influence of sequential behaviors which contain the user decision-making trail. Therefore, for the behavior of clicking a target POI, we view the user's sequential behaviors before that, including browsing the POIs, placing an order and etc., as a behavior path. By observing the historical data on Meituan Takeout APP, we find that there is a close correlation between the behavior path and the click behavior.

The above observations motivate us to develop a model that performs the behavior path matching to predict the user's next click. The core idea is to learn latent factors related to decision-making psychology from the user behavior paths for generating their embeddings. Having in hand the embeddings from behavior paths, the model will perform the matching between historical behavior paths and the current behavior path and estimate the CTRs of candidates.

However, it is challenging to model user behavior paths, because there exist three difficulties: the sparsity of behavior paths, the interference of noise in behavior paths, and the exact matching between behavior paths. Firstly, for a single user, the interactions between the user and the app are not much, which leads to the difficulty in capturing all behavior patterns of the user. For dealing with the sparsity of behavior paths, we leverage contrastive learning to augment the positives of user behavior paths and optimize the learning of the user behavior paths. Secondly, there is a lot of noise in the user behavior paths. For example, a user clicks on a POI due to its attractive cover but immediately returns as soon as the user feels he/she does not like it. Such behavior actually becomes the noise in the path. For reducing the impact of noise, we build a dynamic activation network to focus on several main behaviors of the path. Compared to equally treating all the behaviors in a path, the dynamic activation is more effective and efficient, since some behaviors do have a more obvious effect on subsequent behaviors. Finally, we propose a two-level matching mechanism. In the first level, for the current path, we calculate the activation weight of each historical behavior path and then choose the top-k most similar historical paths. In the second level, given the candidate and

the chosen paths, we calculate the activation weights of the click behaviors which follow the chosen paths for CTR prediction.

Our main contributions are summarized as follows.

- We are the first to introduce user behavior path matching into the industrial CTR prediction. We identify the challenges of modeling behavior paths, i.e., the sparsity, noise, and matching problems of behavior paths.
- We propose a Deep Behavior Path Matching Network (DBPMaN) to predict CTRs, which augments behavior paths, provides behavior path self-activation, and performs a two-level matching (at the level of behavior paths first and then at the level of click behaviors) for CTR prediction.
- We conduct offline experiments on two real-world datasets of different scales and the online A/B test in Meituan advertising. The experimental results show DBPMaN is effective and achieves state-of-the-art performance.

2 RELATED WORK

CTR prediction, as a kernel part of the recommender system, has been a hot topic concerned by industry and academia [18].

The classic solution to CTR prediction is to learn feature interactions, where DeepFM [6], xDeepFM [9], and ONN [17] are early representative deep neural network models and CAN [1] has the state-of-the-art performance among current open source CTR models.

Recently, sequential behavior modeling becomes a new driving force for CTR prediction. The granularity of modeled behaviors ranges from the single behavior (e.g., DIN [21]) to multiple kinds of behaviors (e.g., FeedRec [15]), from short sequences (e.g., DIEN [20], DSIN [5]) to ultra-long sequences (e.g., MIMN [11], SIM [12], ETA [2]). These models aim to capture user interests [5, 19–21] or intention [8] and often adopt the point-to-point activation method, whose input only contains a single kind of behaviors such as click, to estimate the tendency of user interests/intention towards candidates from a probabilistic perspective. They are proven to have continued to improve the accuracy of CTR prediction. In addition, with the great success of Transformer [14] and BERT [4] in the field of NLP, it has been introduced into recommender systems to achieve different recall tasks [7, 13] or CTR predication tasks [3].

Compared to the above work, our work emphasizes behavior paths that imply the decision-making signs and employs behavior paths as the evidence base of CTR prediction.

3 OUR MODEL

3.1 Overview

Firstly, we give the following definitions used in the paper.

Definition 1. (User Behavior Sequence) Let $\mathcal{U}$ denote a user set. For the user $u \in \mathcal{U}$, his/her behavior sequence is composed of his/her behaviors, sorted by occurrence time and denoted by $s = [b_1, ..., b_i, ..., b_T]$, where b_i is the i-th behavior and T is the length of the behavior sequence. In our scenario, this sequence includes user behaviors during the past year. Each behavior includes the id of the interacted item, the behavior type, the interval between the occurrence time and the current time and the relative position

in the sequence, etc. In our scenario, there exist three behavior types: click, impression, and order.

Definition 2. (User Click Sequence) In the user behavior sequence s, there exist a large number of click behaviors. Thus we can form a click sequence s^c from s: $s^c = [b_1^c, ..., b_i^c, ..., b_t^c]$, where t denotes the length of the click sequence.

Definition 3. (User Behavior Path) For the i-th click behavior b_i^c in s^c, let $b_{m(i)}$ denote the corresponding behavior in the behavior sequence s, where $m(i)$ denotes the position in s where the i-th click behavior occurs. Then, the user behavior path with respect to the click behavior b_i^c, denoted by p_i, is the subsequence $[b_{m(i)-l}, ..., b_{m(i)-2}, b_{m(i)-1}]$ in s, where l is the preset length of the behavior path.

From the definition of the user behavior path, we can obviously find that the click behavior and the behavior path are one-to-one correspondence. Fig. 1 gives an example of user behavior paths. In the historical user behavior sequence, there are three user behavior paths whose length is preset to 3: one w.r.t. h_4, one w.r.t. h_7, and one w.r.t. h_{11}.
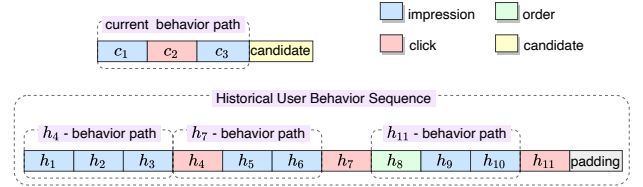

Figure 1: Example of user behavior paths where l=3.

Definition 4. (Behavior Path Sequence) For all click behaviors in s, we can obtain their corresponding behavior paths, respectively, thus forming a behavior path sequence $P = [p_1, ..., p_i, ..., p_t]$, where p_i is the user behavior path w.r.t. the click behavior b_i^c.

Secondly, we give the composition, structure and process of the DBPMaN model.

DBPMaN consists of an embedding layer and three modules, i.e., Path Enhancing Module (PEM), Path Matching Module (PMM), and Path Augmenting Module (PAM), whose structure is shown in Fig. 2.

DBPMaN takes multiple features as input, where the chosen features are from (1) item profile, including item id and its side information (e.g., category, location, rating, etc.); (2) user profile, including user id and his/her side information (e.g., age, gender, city, etc.); (3) behaviors in the user behavior sequence. These features are fed into the embedding layer. For all features, we can obtain their embeddings by looking up the embedding tables. Then, we use sum pooling on different kinds of features to calculate the embedding of user behavior sequence $\mathbf{s} = [\mathbf{e}_1, ..., \mathbf{e}_i, ..., \mathbf{e}_T]$, user embedding $\mathbf{e}^u$ and the embedding of candidate item $\mathbf{e}^{ct}$.

The behaviors in a user behavior path tend to contribute differently to the corresponding next click behavior, i.e., the click behavior which follows the user behavior path. PEM is expected to mine this information to learn more accurate path embeddings. In short, given the embeddings of each behavior path and its corresponding click behavior, PEM first activates the important behaviors in a
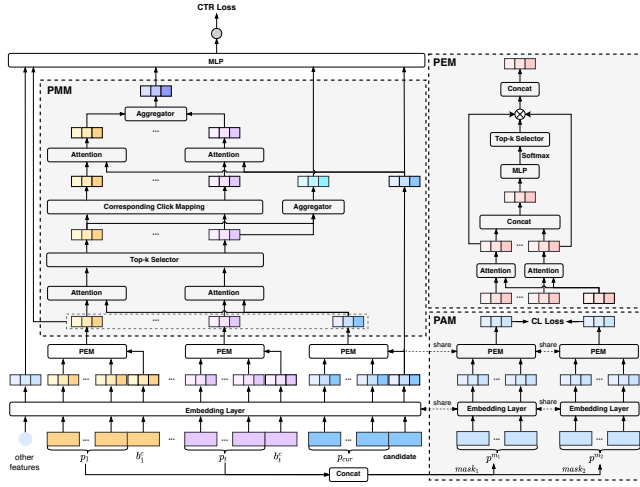
Figure 2: Structure of DBPMaN. (1) PEM enhances the path representations. (2) PMM matches the paths. (3) PAM augments the paths.

behavior path and then optimizes the embedding of the behavior path so as to obtain a more accurate behavior path representation.

Then, PMM uses the embeddings of the current behavior path and historical behavior paths as input to search the k most similar historical behavior paths compared to the current behavior path and then activate the corresponding k click behaviors with the candidate.

In addition, PAM aims to learn more precise and informative behavior path embeddings by contrastive learning. Concretely, we mask each historical behavior path to obtain two augmented paths and feed them into the embedding layer and PEM to calculate their embeddings and then pull embeddings stemming from a same behavior path close by taking the InfoNCE loss [10] as the contrastive loss.

DBPMaN uses the negative log-likelihood function as the main loss, which is widely used in most CTR models. Finally, DBPMaN is trained by combining the main loss with the contrastive loss as the optimization objective.

Due to the limited space, we only describe the PEM and PMM in detail, omitting the other parts of the model.

3.2 Path Enhancing Module (PEM)

For a behavior path $[b_{m(i)-l}, ..., b_{m(i)-2}, b_{m(i)-1}]$ and the following click behavior b_i^c, the embedding layer will generate their embeddings, denoted by $\mathbf{e}_{m(i)-l}, ..., \mathbf{e}_{m(i)-2}, \mathbf{e}_{m(i)-1}$, and $\mathbf{e}_i^c$. Further, a sequence of embeddings $[\mathbf{e}_{m(i)-l}, ..., \mathbf{e}_{m(i)-2}, \mathbf{e}_{m(i)-1}]$ is denoted by $\mathbf{s}_i$.

We firstly apply a local activation unit on the user behavior path, which performs a weighted concat pooling to adaptively calculate the embedding of the behavior path, as shown in Eq. 1.

$$\mathbf{e}_{m(i)-j}^{te} = a\left(\mathbf{e}_{m(i)-j}, \mathbf{e}_i^c\right) \cdot \mathbf{e}_{m(i)-j}, \quad 1 \le j \le l$$
$$\mathbf{s}_i^{te} = [\mathbf{e}_{m(i)-l}^{te}, ..., \mathbf{e}_{m(i)-2}^{te}, \mathbf{e}_{m(i)-1}^{te}] \tag{1}$$
$$\mathbf{p}_i^{te} = \text{concat}(\mathbf{s}_i^{te})$$

where $a(\cdot)$ is an MLP whose output is used as the first-level activation score.

Then, we feed $\mathbf{p}_i^{te}$ into another MLP and learn the second-level activation score of each behavior in the behavior path by a softmax activation function, as shown in Eq. 2.

$$\text{score}_i = \text{softmax}(\text{MLP}(\mathbf{p}_i^{te})) \tag{2}$$

The score_i is an l-dimensional vector, whose entries represent the second-level activation scores of behaviors in the behavior path. Then, only the top-k scores are chosen. According to the chosen scores, we multiply them with their corresponding behavior embeddings in $\mathbf{s}_i^{te}$. By concatenating the embeddings scaled by scores, we get the enhanced path embedding $\mathbf{p}_i^e$.

In this way, we can obtain the sequence of the enhanced path embeddings $\mathbf{P}_e = [\mathbf{p}_1^e, \mathbf{p}_2^e, ..., \mathbf{p}_t^e]$.

3.3 Path Matching Module (PMM)

For a user, there might exist a large number of behavior paths in the user behavior sequence. However, only a few of them are similar to the current behavior path, which can indicate the user's current interests. PMM is designed to search the first k behavior paths most similar to the current path, and then obtain the corresponding k click behaviors, which are believed to make considerable contributions to the user's current interests.

Specifically, given the sequence of the enhanced historical path embeddings $\mathbf{P}_e = [\mathbf{p}_1^e, \mathbf{p}_2^e, ..., \mathbf{p}_t^e]$ and the enhanced embedding of the current behavior path $\mathbf{p}_{cur}^e$, we feed each $\mathbf{p}_i^e \in \mathbf{P}_e$ and $\mathbf{p}_{cur}^e$ into a scoring gate and obtain a similarity score g_i^p, which reflects the importance of the corresponding historical behavior path. The calculation of the scoring gate is shown in Eq. 3.

$$g_i^p = \text{MLP}(\text{concat}(\mathbf{p}_{cur}^e, \mathbf{p}_i^e, \mathbf{p}_{cur}^e \otimes \mathbf{p}_i^e)) \tag{3}$$

where $\otimes$ denotes the hadamard product and $\text{MLP}(\cdot)$ is implemented as a feed forward neural network.

Thus, we can obtain a list of similarity scores $g^p = [g_1^p, g_2^p, ..., g_t^p]$. We sort all scores and choose the top-k scores. With top-k scores, we can get the corresponding historical paths and the click behaviors corresponding to the historical paths. For the chosen click behaviors, a sequence of these click behavior embeddings is denoted by $\mathbf{s}_c = [\mathbf{e}_{c1}, \mathbf{e}_{c2}, ..., \mathbf{e}_{ck}]$. In addition, we multiply each chosen path embedding by the corresponding score and obtain the adjusted embeddings of chosen paths, as shown in Eq. 4.

$$\mathbf{E}^p = \text{concat}(Filter(g^p, [g_i^p \cdot \mathbf{p}_i^e, 1 \le i \le t], k)) \tag{4}$$

where the function $Filter(score, embedding, k)$ sorts historical behavior paths by score and chooses top-k paths.

All chosen click behaviors are supposed to make different contributions to the user's current interests. Thus, we use the same way as one in Eq. 3 to calculate the scores of similarity between the candidate and the chosen click behaviors so as to adaptively calculate the representation vector of user interests by taking into consideration the relevance between them, as shown in Eq. 5.

$$g_i^c = \text{MLP}(\text{concat}(\mathbf{e}^{ct}, \mathbf{e}_{ci}, \mathbf{e}^{ct} \otimes \mathbf{e}_{ci}))$$
$$\mathbf{E}^c = \text{concat}([g_i^c \cdot \mathbf{e}_{ci}, 1 \le i \le k]) \tag{5}$$

At last, $\mathbf{P}_e$, $\mathbf{E}^p$, $\mathbf{E}^c$, $\mathbf{e}^u$ and $\mathbf{e}^{ct}$ are concatenated and then fed into an MLP layer which outputs the predicted CTR.

4 EXPERIMENTAL EVALUATION

4.1 Experimental Setup

Datasets. We adopt the following two datasets for experiments.

- Taobao: a public dataset [22], containing 10-day interactions. We preprocess the data in the same way as what CAN do in [1].
- Meituan: an industrial dataset collected by the Meituan Takeout App, which contains 14-day interactions of 100 million users.

Table1 lists the statistics of processed datasets.

Competitors. We choose the following CTR models which focus on feature interaction modeling as comparison models.

- DeepFM [6]. It combines the factorization machines and deep learning for low-order and high-order feature interactions.
- xDeepFM [9]. It generates feature interactions using the proposed Compressed Interaction Network (CIN) and further combines a CIN and a basic DNN into one unified model.
- DIN [21]. It designs a local activation unit to learn the representation of user interests from historical behaviors w.r.t. a candidate.
- DIEN [20]. It designs an interest extractor layer and an interest evolving layer to capture interests from behavior sequences.
- ONN [17]. It learns different representations for different operations.
- CAN [1]. It disentangles the representation learning and feature interaction modeling via the co-action unit.

Metrics. We use AUC and RelaImpr [16] as the metrics in offline experiments, CTR and CPM (Cost-Per-Mille) as metrics in online experiments.

Implementation Details. We implement DBPMaN [1] by Tensorflow. For all models, we use Adam as the optimizer with a learning rate of 0.001. The model parameters are initialized with a Gaussian distribution (with a mean of 0 and a standard deviation of 0.01). The item embedding dimension is set to 18.

4.2 Performance Comparison

We conduct comparative experiments, comparing our model with the above competitors. The performance results of different models on two datasets are shown in Table 2.

From the results, we find the performance ranking of all the models over two datasets is the same and our DBPMaN surpasses all the competitors. We think that the attention mechanism on path-to-path activation in DBPMaN helps defeat the other models, including attention-based models (i.e., DIN, DIEN, and ONN) which employ point-to-point activation of a candidate and individual behavior in the historical behavior sequence.

An interesting finding is that our DBPMaN model achieves more significant improvement over other models on the Meituan dataset rather than the Taobao dataset. This may stem from the nature of the food takeout scenario. The relatively few food stores surrounding the user, combined with characteristics of user interest in food, lead to repetitive interactions with the same food store in the history of user behaviors. This makes it easy to activate historical paths relevant to the current path.

---

[1]The code is available at https://github.com/Ethan-Yys/DBPMaN.

Table 1: Statistics of datasets.

Datasets	#Users	#Items	#Categories	#Interactions
Taobao	987991	4161138	9437	100095182
Meituan	100000000	15755909	184	5648932411

Table 2: Performance results on offline evaluations.

Model	Taobao		Meituan	
	AUC	RelaImpr	AUC	RelaImpr
DeepFM	0.8125	-13.19%	0.6673	-1.55%
xDeepFM	0.8366	-10.61%	0.6693	-1.25%
ONN	0.8689	-7.16%	0.6705	-1.08%
DIN	0.9308	-0.54%	0.6753	-0.37%
DIEN	0.9324	-0.37%	0.6761	-0.25%
CAN	0.9359	0.00%	0.6778	0.00%
DBPMaN	**0.9381**	**0.24%**	**0.6812**	**0.50%**

Table 3: Ablation study on Meituan dataset.

Model	AUC	CTR	CPM
DBPMaN	0.6812	-	-
DBPMaN w/o PEM	0.6789	-0.6%	-0.5%
DBPMaN w/o PMM	0.6769	-1.0%	-1.1%
DBPMaN w/o PAM	0.6801	-0.2%	-0.2%

Table 4: Online A/B test results.

Model	CTR	CPM	Inference Time(ms)
Baseline	-	-	48
DBPMaN (l=8)	+1.6%	+1.8%	+0.8
DBPMaN (l=16)	+1.3%	+1.5%	+2.1

4.3 Ablation Study

We conduct an ablation study on the Meituan dataset to evaluate the contributions of key modules of DBPMaN. We compare our model with three variants, i.e., DBPMaN w/o PEM, DBPMaN w/o PMM, and DBPMaN w/o PAM. The results are shown in Table 3.

From Table 3, we can find that three variants suffer a decrease in all three metrics, compared to the original DBPMaN. The performance of DBPMaN w/o PMM declines the most, which indicates that PMM plays a more important role than the other two modules.

4.4 Online A/B test

The A/B test is conducted on the Meituan food delivery platform and lasts for 14 days from 2022-08-10 to 2022-08-23, where the baseline model is our last online CTR model which only uses the point-to-point activation method. The results are shown in Table 4, where l denotes the length of the behavior path. Now DBPMaN (l=8) has been deployed online and serves the main traffic of users.

5 CONCLUSION

In this paper, we propose DBPMaN, which models user behavior paths into CTR prediction for the first time. Besides the excellent performance, DBPMaN shows the possibility of exploring user decision-making psychology by modeling behavior paths.

REFERENCES

[1] Weijie Bian, Kailun Wu, Lejian Ren, Qi Pi, Yujing Zhang, Can Xiao, Xiang-Rong Sheng, Yong-Nan Zhu, Zhangming Chan, Na Mou, et al. 2022. CAN: Feature Co-Action Network for Click-Through Rate Prediction. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 57–65.

[2] Qiwei Chen, Changhua Pei, Shanshan Lv, Chao Li, Junfeng Ge, and Wenwu Ou. 2021. End-to-End User Behavior Retrieval in Click-Through RatePrediction Model. *arXiv preprint arXiv:2108.04468* (2021).

[3] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. 4171–4186.

[5] Yufei Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. 2019. Deep Session Interest Network for Click-Through Rate Prediction. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. 2301–2307.

[6] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, Carles Sierra (Ed.). 1725–1731.

[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] Feng Li, Zhenrui Chen, Pengjie Wang, Yi Ren, Di Zhang, and Xiaoyu Zhu. 2019. Graph intention network for click-through rate prediction in sponsored search. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 961–964.

[9] Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. 2018. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1754–1763.

[10] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

[11] 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.

[12] 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.

[13] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[15] Chuhan Wu, Fangzhao Wu, Tao Qi, Qi Liu, Xuan Tian, Jie Li, Wei He, Yongfeng Huang, and Xing Xie. 2022. Feedrec: News feed recommendation with various user feedbacks. In *Proceedings of the ACM Web Conference 2022*. 2088–2097.

[16] Ling Yan, Wu-Jun Li, Gui-Rong Xue, and Dingyi Han. 2014. Coupled group lasso for web-scale ctr prediction in display advertising. In *International Conference on Machine Learning*. PMLR, 802–810.

[17] Yi Yang, Baile Xu, Shaofeng Shen, Furao Shen, and Jian Zhao. 2020. Operation-aware neural networks for user response prediction. *Neural Networks* 121 (2020), 161–168.

[18] Weinan Zhang, Jiarui Qin, Wei Guo, Ruiming Tang, and Xiuqiang He. 2021. Deep Learning for Click-Through Rate Estimation. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. 4695–4703.

[19] Yu Zheng, Chen Gao, Jianxin Chang, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2022. Disentangling Long and Short-Term Interests for Recommendation. In *Proceedings of the ACM Web Conference 2022*. 2256–2267.

[20] 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.

[21] 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.

[22] Han Zhu, Xiang Li, Pengye Zhang, Guozheng Li, Jie He, Han Li, and Kun Gai. 2018. Learning tree-based deep model for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1079–1088.