Deep User Match Network for Click-Through Rate Prediction

Zai Huang* , Mingyuan Tao, Bufeng Zhang
Alibaba Group, Hangzhou, China
{huangzai.hz,mingyuan.tmy,feitong}@alibaba-inc.com

ABSTRACT

Click-through rate (CTR) prediction is a crucial task in many applications (e.g. recommender systems). Recently deep learning based models have been proposed and successfully applied for CTR prediction by focusing on feature interaction or user interest based on the item-to-item relevance between user behaviors and candidate item. However, these existing models neglect the user-to-user relevance between the target user and those who like the candidate item, which can reflect the preference of target user. To this end, in this paper, we propose a novel Deep User Match Network (DUMN) which measures the user-to-user relevance for CTR prediction. Specifically, in DUMN, we design a User Representation Layer to learn a unified user representation which contains user latent interest based on user behaviors. Then, User Match Layer is designed to measure the user-to-user relevance by matching the target user and those who have interacted with candidate item and modeling their similarities in user representation space. Extensive experimental results on three public real-world datasets validate the effectiveness of DUMN compared with state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Personalization**; **Retrieval models and ranking**; **Probabilistic retrieval models**.

KEYWORDS

Click-through Rate Prediction; Deep learning; User Representation

ACM Reference Format:
Zai Huang* , Mingyuan Tao, Bufeng Zhang. 2021. Deep User Match Network for Click-Through Rate Prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3404835.3463078

1 INTRODUCTION

Click-through rate (CTR) prediction which aims at estimating the probability that an item will be clicked by a user, is a crucial task in many applications, such as recommender systems and online advertising. In recommender systems, candidate items are ranked based on the estimated CTR [21]. In online advertising with the Cost per click (CPC) form, the estimated CTR may benefit the ranking and pricing of ads [26]. The performance of CTR prediction model not only impacts user experience, but also influences the revenue of the business. Thus, CTR prediction has attracted extensive concerns of both academia and industry [2, 16, 17, 24, 26].

In the literature, many efforts have been made for CTR prediction [2, 9, 18, 20, 23, 26]. Among them, traditional shallow models such as Logistic Regression (LR) [2, 15] and Factorization Machines (FM) [18], are widely applied for CTR prediction. In recent years, with the strong ability of deep learning on feature representation and interactions from different fields, some deep learning models have been proposed [3, 5, 7, 13, 17]. For example, Wide&Deep [3] model consists of a wide linear model and a deep model to automatically learn low-order features and high-order feature interactions. PNN [17] uses a product layer to capture the high-order feature patterns. DeepFM [5] combines FM with deep network to capture low-order and high-order feature interactions. As user latent interest is not shown clearly, but usually implied in the user behavior data, some deep learning models have been proposed to discover behavioral patterns and extract user interest from user behaviors [14, 21, 25, 26]. For example, DIN [26] and DIEN [25] measure item-to-item relevance between user interacted items and the candidate item for obtaining the adaptive interest representation. Though these existing methods have made a great success, they ignore the user-to-user relevance between the target user and those who like the candidate item, which can reflect the preference of target user [1]. Therefore, the problem of modeling the user-to-user relevance for CTR prediction remains pretty much open.

In this paper, we propose a novel Deep User Match Network (DUMN) for CTR prediction by matching the target user and those who like the candidate item (i.e. who have interacted with candidate item in our work) to measure the user-to-user relevance. Specifically, in DUMN, with the embedding vectors of user behavior and profile features, User Representation Layer is designed to learn a unified user representation which contains information of the user latent interest and profile. Then we design a User Match Layer to measure the user-to-user relevance by matching the target user and those who have interacted with candidate item and modeling their similarities in user representation space. With the user-to-user similarity relevance, we generate the matching representation to reflect the preference of target user. Finally, the matching representation and user representation of the target user, the user-to-user similarity relevance, and the embedding vectors of the candidate item and context are concatenated and fed into several fully connected layers for outputting predicted CTR.

The main contributions of this paper are summarized as follows:

- We highlight the importance of capturing the user-to-user relevance between target user and those who like the candidate item. A novel model called DUMN is proposed to measure the user-to-user relevance for CTR prediction.

*Corresponding Author.

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3463078

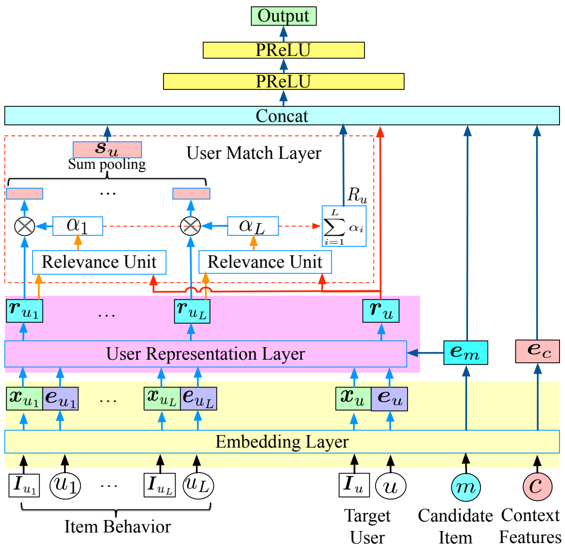


Figure 1: The structure of DUMN. The input of DUMN includes the profile and user behavior of the target user, features and item behavior of the candidate item, and context. The output is the probability the target user will click the candidate item.

- We design two network architectures to learn a unified user representation and model the user-to-user relevance with the similarities based on user representations, respectively.
- We conduct extensive experiments on three public real-world datasets in terms of CTR prediction. The results clearly validate the effectiveness of our proposed DUMN compared with several state-of-the-art baselines.

2 METHODOLOGY

In this section, we introduce the design of Deep User Match Network (DUMN). Specifically, we first give the preliminaries of CTR prediction. Then, we elaborate the technical details of DUMN.

2.1 Preliminaries

There are interactions (e.g. click or rating) between users and items, which are recorded in the historical data. Based on the historical data, we can obtain the interaction behaviors for the target user and candidate item, which can be defined as follows:

Definition 1. ***User Behavior**: Given a user u, the user behavior $\boldsymbol{I_u}$ is the sequential list of the interacted items with corresponding features such as item id, category, etc., i.e. $\boldsymbol{I_u} = [i_1, i_2, \cdots, i_{N_u}]$, where i_k is the k-th interacted item and N_u is the length of $\boldsymbol{I_u}$.*

Definition 2. ***Item Behavior**: Given an item m, the item behavior $\boldsymbol{U_m}$ is the sequential list of users who interact with m with corresponding profile features such as user id, age, etc., as well as their user behaviors when interacting with m. $\boldsymbol{U_m}$ is formalized as $\boldsymbol{U_m} = [(u_1, \boldsymbol{I_{u_1}}), (u_2, \boldsymbol{I_{u_2}}), \cdots, (u_L, \boldsymbol{I_{u_L}})]$, where u_k is the k-th interacted user, $\boldsymbol{I_{u_k}}$ is the user behavior of u_k, L is the length of $\boldsymbol{U_m}$.*

The goal of CTR prediction is to estimate the probability p that the target user u clicks the candidate item m with the model $\mathcal{F}$, given the user behavior $\boldsymbol{I_u}$, item behavior $\boldsymbol{U_m}$, the user features of u, item features of m and context c, i.e., $p = \mathcal{F}(\boldsymbol{I_u}, \boldsymbol{U_m}, u, m, c)$.

2.2 Deep User Match Network

In this subsection, we will introduce the technical details of DUMN. The architecture of DUMN is shown in Figure 1, mainly consisting of three components, i.e., Embedding Layer, User Representation Layer and User Match Layer.

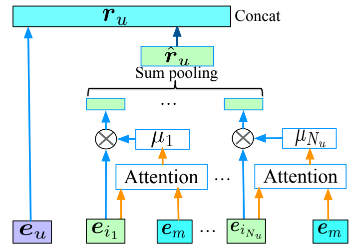


Figure 2: User Representation Layer in DUMN.

2.2.1 ***Embedding Layer***. Embedding Layer aims at converting the features of the input of DUMN to embedding vectors. In Embedding Layer, similar to the work [26], we encode each feature into a one-hot vector with high-dimensional sparse binary encoding. Then the one-hot vectors are transformed into low-dimensional dense vectors by the embedding layer. Through Embedding Layer, we can obtain the embedding vectors for the input. We denote $\boldsymbol{e_u}$, $\boldsymbol{e_m}$, $\boldsymbol{e_c}$, $\boldsymbol{x_u}$ and $\boldsymbol{z_m}$ as the embedding vectors of the profile of target user u , candidate item m, context c, user behavior $\boldsymbol{I_u}$ and item behavior $\boldsymbol{U_m}$, respectively, where $\boldsymbol{x_u} = [\boldsymbol{e_{i_1}}, \boldsymbol{e_{i_2}}, \cdots, \boldsymbol{e_{i_{N_u}}}]$, $\boldsymbol{e_{i_k}}$ is the embedding vector of the k-th interacted item in $\boldsymbol{I_u}$, and $\boldsymbol{z_m} = [(\boldsymbol{x_{u_1}}, \boldsymbol{e_{u_1}}), (\boldsymbol{x_{u_2}}, \boldsymbol{e_{u_2}}), \cdots, (\boldsymbol{x_{u_L}}, \boldsymbol{e_{u_L}})]$, $\boldsymbol{x_{u_k}}$ and $\boldsymbol{e_{u_k}}$ are the embedding vectors of the user behavior and profile of the k-th interacted user in $\boldsymbol{U_m}$.

2.2.2 ***User Representation Layer***. User Representation Layer targets at learning a unified user representation for all users. Actually, User Representation Layer is a general architecture, which can be implemented with different user representation methods such as MV-URL [19] and SUMN [4]. Several efforts have shown that extracting user interest by measuring the item-to-item relevance between user interacted items and candidate item is quite beneficial for CTR prediction [21, 25, 26]. Thus, in our work, for simplicity, with obtained embedding vectors of the user behavior and profile via Embedding Layer, we learn the unified user representation in User Representation Layer by combining user latent interest and profile features. Concretely, in User Representation Layer, we leverage an attention mechanism to measure the item-to-item relevance based on item embedding representations and obtain the user interest representation according to the relevance. The architecture of User Representation Layer is shown in Figure 2.

As shown in Figure 2, for each user u, with the embedding vectors, i.e., $\boldsymbol{x_u} = [\boldsymbol{e_{i_1}}, \boldsymbol{e_{i_2}}, \cdots, \boldsymbol{e_{i_{N_u}}}]$, $\boldsymbol{e_u}$ and $\boldsymbol{e_m}$ of the user behavior, user profile and candidate item, in the attention mechanism of User Representation Layer, we adaptively learn the item-to-item relevance between the candidate item and each item in the user behavior in item embedding representation space. The formulations are as follows:

$$\gamma_k = \boldsymbol{V_a}^T \sigma(\boldsymbol{W_{a_1}} \boldsymbol{e_{i_k}} + \boldsymbol{W_{a_2}} \boldsymbol{e_m} + \boldsymbol{b_a}),$$
$$\mu_k = \frac{exp(\gamma_k)}{\sum_{j=1}^{N_u} exp(\gamma_j)}, \quad (1)$$

where $\boldsymbol{e_{i_k}}$ is the embedding vector of the k-th interacted item in the user behavior of u, γ_k is the item-to-item relevance between the k-th interacted item and candidate item, μ_k is the attention weight, and σ is the sigmoid function. $\boldsymbol{V_a}$, $\boldsymbol{W_{a_1}}$, $\boldsymbol{W_{a_2}}$, and $\boldsymbol{b_a}$ are learning parameters.

After computing the attention weights, the user interest representation, i.e. $\hat{\boldsymbol{r}}_u$, can be modeled as a vector by a weighted sum aggregated result of $\boldsymbol{x_u}$, which is formalized as

$$\hat{\boldsymbol{r}}_u = \sum_{k=1}^{N_u} \mu_k \boldsymbol{e}_{\boldsymbol{i_k}}. \tag{2}$$

By concatenating the user interest representation $\hat{\boldsymbol{r}}_u$ and the embedding vector $\boldsymbol{e_u}$ of user profile, we can obtain a unified user representation $\boldsymbol{r}_u$ for each user u, i.e. $\boldsymbol{r}_u = concat(\hat{\boldsymbol{r}}_u, \boldsymbol{e_u})$.

2.2.3 ***User Match Layer.*** As shown in Figure 1, User Match Layer aims at measuring the user-to-user relevance by matching target user and users in the item behavior of candidate item and modeling their similarities with a Relevance Unit, after getting their user representations through User Representation Layer. With the user-to-user similarity relevance, User Match Layer also generates a matching representation to reflect the preference of target user.

Relevance Unit. Relevance Unit measures the user-to-user relevance for two input matching users (u_a, u_b) by modeling their similarity in the user representation space. Concretely, in our work, we use *cosine similarity* to measure the user-to-user relevance. The similarity $sim(u_a, u_b)$ of u_a and u_b can be calculated as follows:

$$sim(u_a, u_b) = cos(\boldsymbol{r}_{u_a}, \boldsymbol{r}_{u_b}) = \frac{{\boldsymbol{r}_{u_a}}^T \boldsymbol{r}_{u_b}}{|\boldsymbol{r}_{u_a}||\boldsymbol{r}_{u_b}|}, \tag{3}$$

where $\boldsymbol{r}_{u_a}$ and $\boldsymbol{r}_{u_b}$ are the user representations of u_a and u_b.

Through Relevance Unit, we can get the user-to-user similarity relevance α_k between the target user u and each user u_k in the item behavior of candidate item, i.e. $\alpha_k = sim(u, u_k)$. With the similarity relevance, we can obtain the matching representation $\boldsymbol{s_u}$ for target user u by aggregating user representations of users in the item behavior of candidate item, which is formalized as

$$\boldsymbol{s_u} = \sum_{k=1}^{L} \alpha_k \boldsymbol{r}_{\boldsymbol{u_k}}, \tag{4}$$

where $\boldsymbol{r}_{\boldsymbol{u_k}}$ is the user representation of the k-th interacted user in the item behavior of candidate item.

Moreover, we can get the total user-to-user similarity relevance R_u between the target user and item behavior of candidate item, which is model as the sum of their similarity relevance, i.e.

$$R_u = \sum_{k=1}^{L} \alpha_k. \tag{5}$$

2.2.4 ***Output Layer.*** Output Layer targets at calculating the predicted probability that the target user u clicks the candidate item m. As shown in Figure 1, we first concatenate the matching representation $\boldsymbol{s_u}$ and user representation $\boldsymbol{r_u}$ of target user, total user-to-user relevance R_u, and the embedding vectors, $\boldsymbol{e_m}$ and $\boldsymbol{e_c}$ of the candidate item and context, together to a vector $\boldsymbol{h_0}$. Then, we feed $\boldsymbol{h_0}$ to two fully connected layers with the activation function PReLU [6, 26]. Finally, the predicted CTR p can be obtained by feeding the last hidden layer above, i.e. $\boldsymbol{h_2}$ to a fully connected layer with the sigmoid function. The formulations are as follows:

$$\begin{aligned} \boldsymbol{h_0} &= concat(\boldsymbol{s_u}, \boldsymbol{r_u}, R_u, \boldsymbol{e_m}, \boldsymbol{e_c}), \\ \boldsymbol{h_1} &= PReLU(\boldsymbol{W_{o_1}} \boldsymbol{h_0} + b_{o_1}), \\ \boldsymbol{h_2} &= PReLU(\boldsymbol{W_{o_2}} \boldsymbol{h_1} + b_{o_2}), \\ p &= \sigma(\boldsymbol{W_{o_3}} \boldsymbol{h_2} + b_{o_3}), \end{aligned} \tag{6}$$

where $\boldsymbol{W_{o_1}}, b_{o_1}, \boldsymbol{W_{o_2}}, b_{o_2}, \boldsymbol{W_{o_3}}, b_{o_3}$ are learning parameters.

Table 1: The statistics of three datasets.

Dataset	# Users	# Items	# Categories	# Reviews	# Samples
Sports	35598	18357	1073	296337	521478
Beauty	22363	12101	221	198502	352278
Grocery	14681	8713	129	151254	273146

2.3 Loss Function

The negative log-likelihood loss function is widely used to learn parameters of a CTR model [2, 17, 26], which can be formulated as

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} (y_i log(p_i) + (1 - y_i) log(1 - p_i)), \tag{7}$$

where p_i is the predicted CTR by DUMN for the i-th sample in the dataset with the size N and $y_i \in \{0, 1\}$ is the click label.

3 EXPERIMENTS

In this section, we conduct experiments on three real-world datasets to verify the effectiveness of DUMN by comparing with several state-of-the-art methods.

3.1 Datasets

We conduct experiments on three public real-world datasets of the Amazon product reviews and metadata, i.e., Sports, Beauty and Grocery datasets[1]. In three datasets, the product reviews consist of the information of the user, item, rating and timestamp. As we focus on the CTR prediction task, we treat all existing rating reviews as positive interactions in which the click label is 1. We filter out users and items that have less than five reviews. We sort the reviews by the timestamp to build the user behaviors and item behaviors. We build the negative samples by replacing the item in each review with another item randomly selected from the non-clicked item set, which is often used in the works [25, 26]. The statistics of three datasets are shown in Table 1.

we split samples of each dataset into the training and testing parts according to the timestamp, which can effectively avoid feature leakage. In this way, for each dataset, by sorting samples by the timestamp, we can find a suitable splitting timestamp and then take 85% of the whole dataset as the training set and the rest 15% of data as the testing set.

3.2 Baselines

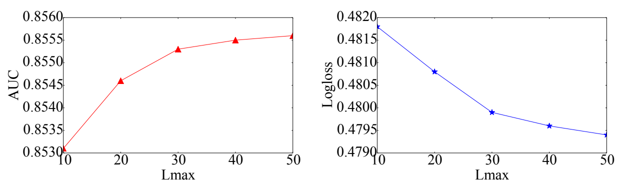
In order to demonstrate the effectiveness of DUMN[2], we compare it with the state-of-the-art methods.

- **SVD++** [11] is a collaborative filtering method combining a latent factor model and a neighborhood model. We take the users in item behavior as neighbors of the target user in experiments.
- **Wide&Deep** [3] model consists of a wide linear model and a deep model to automatically learn low-order features and high-order feature interactions
- **PNN** [17] uses a product layer to capture the high-order feature patterns.
- **DIN** [26] uses an attention mechanism to capture the adaptive interest representation with user behaviors.
- **DIEN** [25] utilizes a two-layer GRU to model user interest evolving process and extract user latent temporal interest.
- **GRU4Rec** [8] uses GRU to model sequential interacted items. We extend it to model sequential users in item behavior as well.

[1] http://jmcauley.ucsd.edu/data/amazon/
[2] https://github.com/hzzai/DUMN

Table 2: Performance comparison on three datasets. The percentage in the last row of DUMN is the relative improvement compared to the best baseline with t-test at p-value of 0.05.

Model	Sports		Beauty		Grocery	
	AUC	Logloss	AUC	Logloss	AUC	Logloss
SVD++	0.7070	0.6347	0.6867	0.6831	0.6385	0.8306
Wide&Deep	0.7926	0.5488	0.8064	0.5516	0.6823	0.6634
PNN	0.8012	0.5408	0.8081	0.5509	0.7033	0.6324
DIN	0.8074	0.5334	0.8178	0.5375	0.7053	0.6284
DIEN	0.8086	0.5331	0.8231	0.5218	0.7141	0.6249
GRU4Rec	0.8136	0.5263	0.8416	0.4923	0.7949	0.5360
DUMN	**0.8173**	**0.5227**	**0.8555**	**0.4796**	**0.8107**	**0.5159**
	+0.45%	**-0.68%**	**+1.65%**	**-2.58%**	**+2.0%**	**-3.75%**



(a) AUC (b) Logloss

Figure 3: Influence of the maximum length L_{max} of item behavior on DUMN in the training phase on Beauty dataset.

The baselines are implemented based on the public code[3] provided in the work [25]. Note that, based on the implementation of User Representation Layer in our work, DUMN will become DIN when removing User Match Layer.

3.3 Evaluation Metrics

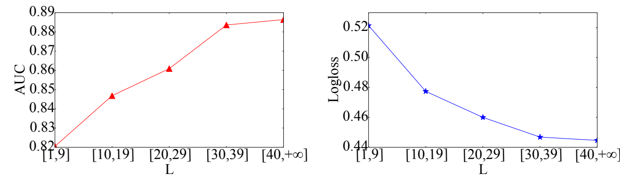
We use AUC (Area Under ROC Curve) and Logloss as the evaluation metrics [2, 12, 17, 22, 26], which are widely used to assess the performance of a CTR model.

3.4 Experimental Results

3.4.1 ***Performance Comparison.*** To investigate the performance of DUMN and baselines, we repeat all experiments 10 times and report the average results. In experiments, the dimensions of embedding vectors of item and user profile are set as 36 and 18, respectively. Besides, we set the maximum lengths of the user behavior and item behavior as 20 and 40, respectively. We set the batch size as 128, the learning rate as 0.001 to minimize the loss function i.e. Eq. (7) with Adam [10] for training all models.

Table 2 shows the comparison results on three datasets. We can find that our proposed DUMN achieves the best performance over all the metrics. Comparing to the best baseline GRU4Rec, the average relative improvement of DUNM on AUC is 1.37% and the number on Logloss is 2.34%, which is significant in the CTR prediction task [3, 5, 26]. Specifically, first, PNN and Wide&Deep perform better than SVD++, as they can learn low-order and high-order feature interactions which are important in CTR prediction [2, 3, 5, 17], while SVD++ cannot. Second, DIN performs better than PNN and Wide&Deep by extracting user interest from user behavior data, indicating that extracting user interest is very effective in CTR prediction. Third, DIEN beats DIN, as DIEN can model user interest evolving process and extract user latent temporal interest. Fourth, GRU4Rec performs better than DIEN, implying that modeling item behaviors is effective for CTR prediction. Fifth, in the comparative

[3]https://github.com/mouna99/dien



(a) AUC (b) Logloss

Figure 4: Performance of DUMN on 5 groups of testing samples on the Beauty dataset.

experiment with DUMN and DIN, DUMN performs better than DIN, which demonstrates that User Match Layer measuring user-to-user similarity relevance between the target user and users in the item behavior of candidate item can be beneficial for CTR prediction. Finally, DUMN achieves the best performance on all three datasets, which clearly validates the effectiveness of DUMN by learning user representations with user interest in User Representation Layer and measuring the user-to-user relevance in User Match Layer.

3.4.2 ***Influence of the length of item behavior.*** As the length L of item behavior, i.e., the number of interacted users matched with the target user, impacts the performance of DUMN, we conduct experiments to investigate the influence of the length of item behavior in both training and testing phases.

Training phase. In the training phase, we set the maximum length L_{max} of item behavior from the set $\{10, 20, 30, 40, 50\}$ for training DUMN. We report average results by repeating experiments with 10 times on the Beauty dataset, which are shown in Figure 3. We can find that DUMN improves greatly the performance on both AUC and Logloss as L_{max} increases when L_{max} <= 30. The performance of DUMN rises slowly when L_{max} >= 30.

Testing phase. We train DUMN with the maximum length L_{max} as 50 of the item behavior for testing. In the testing phase, we first filter out testing samples having no item behavior and bucket the rest samples into 5 groups according to the length L of item behavior, i.e. $[1, 9]$, $[10, 19]$, $[20, 29]$, $[30, 39]$, $[40, +\infty]$. Then, we investigate the performance of DUMN on 5 groups of testing samples on the Beauty dataset and report average results by repeating experiments with 10 times. The results are shown in Figure 4. We can find that DUMN improves average performance on both AUC and Logloss as L of the test sample group increases when L <= 30. DUMN achieves nearly the same performance when L >= 30.

These evidences indicate that the more interacted users matched to measure the user-to-user relevance with target user, the better performance DUMN achieves. Also, they imply that DUMN is effective for CTR prediction by measuring the user-to-user relevance.

4 CONCLUSIONS

In this paper, we proposed a novel model called Deep User Match Network (DUMN) to measure the user-to-user relevance for CTR prediction. Specifically, In DUMN, User Representation Layer was designed to learn a unified user representation containing user latent interest based on user behaviors. Then, User Match Layer was developed to measure the user-to-user relevance by matching the target user and those who have interacted with the candidate item and modeling their similarities in user representation space. Extensive experimental results on three public datasets demonstrated the effectiveness of our proposed DUMN.

REFERENCES

[1] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2014. Neighbor selection and weighting in user-based collaborative filtering: a performance prediction approach. *ACM Transactions on the Web (TWEB)* 8, 2 (2014), 1–30.

[2] Olivier Chapelle, Eren Manavoglu, and Romer Rosales. 2015. Simple and Scalable Response Prediction for Display Advertising. *ACM Transactions on Intelligent Systems and Technology* 5, 4 (2015), 61.

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[4] Jie Gu, Feng Wang, Qinghui Sun, Zhiquan Ye, Xiaoxiao Xu, Jingmin Chen, and Jun Zhang. 2020. Exploiting Behavioral Consistence for Universal User Representation. *arXiv preprint arXiv:2012.06146* (2020).

[5] 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 26th International Joint Conference on Artificial Intelligence*. 1725–1731.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*. 1026–1034.

[7] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 355–364.

[8] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[9] Zai Huang, Zhen Pan, Qi Liu, Bai Long, Haiping Ma, and Enhong Chen. 2017. An Ad CTR prediction method based on feature learning of deep and shallow layers. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2119–2122.

[10] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[11] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 426–434.

[12] Xiang Li, Chao Wang, Bin Tong, Jiwei Tan, Xiaoyi Zeng, and Tao Zhuang. 2020. Deep Time-Aware Item Evolution Network for Click-Through Rate Prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 785–794.

[13] 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.

[14] Ze Lyu, Yu Dong, Chengfu Huo, and Weijun Ren. 2020. Deep Match to Rank Model for Personalized Click-Through Rate Prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[15] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1222–1230.

[16] Zhen Pan, Enhong Chen, Qi Liu, Tong Xu, Haiping Ma, and Hongjie Lin. 2016. Sparse factorization machines for click-through rate prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 400–409.

[17] Yanru Qu, Han Cai, Kan Ren, Weinan Zhang, Yong Yu, Ying Wen, and Jun Wang. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining*. IEEE, 1149–1154.

[18] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.

[19] Weiqing Wang, Hongzhi Yin, Xingzhong Du, Wen Hua, Yongjun Li, and Quoc Viet Hung Nguyen. 2019. Online user representation learning across heterogeneous social networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 545–554.

[20] Shu Wu, Feng Yu, Xueli Yu, Qiang Liu, Liang Wang, Tieniu Tan, Jie Shao, and Fan Huang. 2020. TFNet: Multi-Semantic Feature Interaction for CTR Prediction. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1885–1888.

[21] Zhibo Xiao, Luwei Yang, Wen Jiang, Yi Wei, Yi Hu, and Hao Wang. 2020. Deep Multi-Interest Network for Click-through Rate Prediction. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2265–2268.

[22] Runlong Yu, Qi Liu, Yuyang Ye, Mingyue Cheng, Enhong Chen, and Jianhui Ma. 2020. Collaborative List-and-Pairwise Filtering from Implicit Feedback. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[23] 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. *arXiv preprint arXiv:2104.10907* (2021).

[24] 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.

[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.