Deep Match to Rank Model for Personalized Click-Through Rate Prediction

Ze Lyu, Yu Dong, Chengfu Huo, Weijun Ren
Alibaba Group
{lvze.lz, dongyu.dy, chengfu.huocf, afei}@alibaba-inc.com

Abstract

Click-through rate (CTR) prediction is a core task in the field of recommender system and many other applications. For CTR prediction model, personalization is the key to improve the performance and enhance the user experience. Recently, several models are proposed to extract user interest from user behavior data which reflects user's personalized preference implicitly. However, existing works in the field of CTR prediction mainly focus on user representation and pay less attention on representing the relevance between user and item, which directly measures the intensity of user's preference on target item. Motivated by this, we propose a novel model named Deep Match to Rank (DMR) which combines the thought of collaborative filtering in matching methods for the ranking task in CTR prediction. In DMR, we design User-to-Item Network and Item-to-Item Network to represent the relevance in two forms. In User-to-Item Network, we represent the relevance between user and item by inner product of the corresponding representation in the embedding space. Meanwhile, an auxiliary match network is presented to supervise the training and push larger inner product to represent higher relevance. In Item-to-Item Network, we first calculate the item-to-item similarities between user interacted items and target item by attention mechanism, and then sum up the similarities to obtain another form of user-to-item relevance. We conduct extensive experiments on both public and industrial datasets to validate the effectiveness of our model, which outperforms the state-of-art models significantly.

Introduction

Matching and ranking are two classic stages for information retrieval in recommender system. At the stage of matching, aka. candidate generation, a small set of candidates are retrieved from the whole item set by matching user with items. Collaborative filtering based methods are widely used to calculate the user-to-item relevance and select most relevant items. At the ranking stage, ranking models assign comparable scores to the candidates generated by different matching methods, and the top-N scoring items are presented to the end-user. User click is a very important evaluation metric in recommender system, which is the basis of all subsequent conversion behaviors. And click-through rate (CTR) prediction has received much attention from communities of both academia and industry.

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Personalization is the key to improve the performance of CTR prediction and enhance the user experience. Many deep learning based methods have been proposed for CTR prediction, which can learn implicit feature interactions and enhance the model capability. Most of these methods pay attention on designing network structure for automatic feature interactions (Cheng et al. 2016; He and Chua 2017; Wang et al. 2017). Recently, several models are proposed to extract user interest from user behavior data like click and purchase (Zhou et al. 2018; 2019), which is very important for recommendation settings where users do not show their interest explicitly. To represent user interest, item-to-item relevance between user interacted item and target item is considered. However, these models mainly focus on user representation and neglect to represent the user-to-item relevance. The user-to-item relevance directly measures user's personalized preference on target item, which is carefully modeled in collaborative filtering based matching methods.

Based on these observations, we propose a novel model called Deep Match to Rank (DMR) which uses the thought of collaborative filtering in matching methods to capture the user-to-item relevance, consequently improving the performance of CTR prediction. User-to-Item Network and Item-to-Item network are the two sub-networks in DMR to represent the user-to-item relevance respectively. In User-to-Item Network, we directly calculate relevance between user and item by inner product of the corresponding representation in the embedding space, where the user representation is extracted from user behavior. In consideration of that more recent behaviors reflect user's temporal interest better, we apply attention mechanism to adaptively learn the weight for each behavior with regard to its position in behavior sequence. Meanwhile, we propose an auxiliary match network to push larger inner product to represent higher relevance and help fit the User-to-Item Network better. The auxiliary match network alone can be viewed as a matching method, whose task is to predict which item to be clicked next according to user's historical behavior, and we jointly train the matching model and ranking model in DMR. In Item-

to-Item Network, we first calculate the item-to-item similarities between user interacted items and target item by attention mechanism in which position information is also considered. And then we add up the item-to-item similarities to obtain another form of user-to-item relevance. Note that, at the matching stage, candidates are usually generated by multiple matching methods to satisfy the diversity of user requirement, and the user-to-item relevance scores are not comparable across different methods. In DMR, the intensities of relevance can be compared with each other in a uniform way.

The main contributions of this paper are summarized as follows:

- We point out the importance of capturing the relevance between user and item, which can make CTR prediction model more personalized and effective. Motivated by this, we propose a novel model called DMR, which applies the thought of collaborative filtering in matching methods to represent the relevance with User-to-Item Network and Item-to-Item Network respectively.
- We design auxiliary match network which can be viewed as a matching model to help train User-to-Item Network better. To the best of our knowledge, DMR is the first model that jointly trains matching and ranking in CTR prediction.
- Considering that more recent behaviors contribute more to user's temporal interest, we introduce positional encoding in attention mechanism to adaptively learn the weight for each behavior.
- We conduct extensive experiments on both public and industrial datasets, which show significant improvement of proposed DMR over state-of-art models. Our code[1] is publicly available for reproducibility.

Related Work

Recently, deep learning based CTR prediction model have received much attention and achieved remarkable effectiveness. Compared with traditional linear model, deep learning based method can enhance the model capability and learn implicit feature interactions by non-linear transformation. By learning low-dimensional representation from high-dimensional sparse feature, deep models have a better estimation on the combinations of features rarely appear (Cheng et al. 2016).

However, large dimension of sparse feature in practical applications brings big challenge: deep models may overfit and over-generalize. Based on this, different models are proposed to better model feature interactions and improve the performance of CTR prediction. Wide&Deep (Cheng et al. 2016) combines the advantage of linear model and non-linear deep model by joint training. Deep Crossing (Shan et al. 2016) applies a deep residual network to learn cross features. PNN (Qu et al. 2016) introduces a product layer along with fully connected layers to explore high-order feature interactions. Based on factorization-machine (FM) (Rendle 2010) which can model second-order feature interactions, AFM (Xiao et al. 2017) learns weighted feature interactions by attention mechanism (Bahdanau, Cho, and Bengio 2015). DeepFM (Guo et al. 2017) and NFM (He and Chua 2017) combine lower-order and high-order feature interactions by applying FM with deep network. DCN (Wang et al. 2017) introduces cross network to learn certain bounded-degree feature interactions. In our proposed DMR, the representation of user-to-item relevance can be viewed as a feature interaction between user and item.

Different from search ranking, in recommender system and many other applications, users do not show their intention clearly. Thus, capturing user interest from user behavior is crucial for CTR prediction, while the models mentioned above pay less attention on this. User behavior feature with variable length is usually transformed into fixed-length vector by simple average pooling (Covington, Adams, and Sargin 2016), meaning that all behaviors are equally important. DIN (Zhou et al. 2018) improves this by weighted sum pooling to represent user interest, where the weights of each user behavior with respect to the target item are adaptively learned by attention mechanism. DIEN (Zhou et al. 2019) not only extracts user interest but also models temporal evolution of interest. DSIN (Feng et al. 2019) leverages session information in behavior sequence to model evolution of interest. In our model, inspired by Transformer (Vaswani et al. 2017), we introduce positional encoding into attention mechanism to capture user's temporal interest. Despite great progress, these methods focus on user representation and neglect to represent the user-to-item relevance, which directly measures the intensity of user's preference on target item. In proposed DMR, we pay attention on representing the user-to-item relevance to improve the performance of personalized CTR model.

Methods based on collaborative filtering (CF) are very successful in building recommender systems at the stage of matching (Su and Khoshgoftaar 2009). Among the methods, item-to-item CF (Sarwar et al. 2001; Linden, Smith, and York 2003) has been widely used in industrial recommendation settings, owing to its interpretability and efficiency in realtime personalization. By precomputing the item-to-item similarity matrix, similar items to user clicked ones are recommended to the user. To compute the item-to-item similarity, early works focus on statistical measures such as cosine similarity and Pearson coefficient. He et al. (2018) propose deep learning based method NAIS for item-to-item CF with attention mechanism to distinguish different importance of user behavior, which shares similar idea with DIN (Zhou et al. 2018). Different from item-to-item CF, matrix factorization based CF methods (Koren 2008; Koren, Bell, and Volinsky 2009) directly calculate the relevance between user and item by inner product of the corresponding representation in the reduced space. With similar inner product based form, deep learning based methods are proposed to learn user representation from user's historical behaviors, which can be regarded as a non-linear generalization of factorization techniques. Covington, Adams, and Sargin (2016) pose matching as extreme multi-classification, where the prediction problem becomes accurately classify-

[1] https://github.com/lvze92/DMR

ing the item that user will click next based on user's historical behaviors. Hidasi et al. (2016) apply GRU (Cho et al. 2014) on the task of session based recommendations. TDM (Zhu et al. 2018) uses a tree-based method to surpass the inner product based methods. In our model, on one hand, we use inner product between user representation and item representation to obtain one kind of user-to-item relevance; on the other hand, we apply attention mechanism to represent item-to-item similarity and further obtain another kind of user-to-item relevance.

Deep Match to Rank Model

In this section, we elaborate the design of Deep Match to Rank (DMR) model. First, we recapitulate the basic structure of deep learning based CTR model from two aspects: feature representation and multiple layer perceptron. And then we introduce the overall structure of DMR with two sub-networks to model the user-to-item relevance.

Feature Representation

There are four categories of features in our recommender system: *User Profile*, *User Behavior*, *Target Item* and *Context*. *User Profile* contains *user ID*, *consumption level* and so on; Features in *Target Item* are *item ID*, *category ID* etc.; *User Behavior* is the sequential list of user interacted items with corresponding features such as *item ID*, *category ID* etc.; *Context* contains *time*, *matching method* and corresponding *matching score* and so on.

Most of the features are categorical, which can be transformed into one-hot vectors with high dimension. In deep learning based model, the one-hot vectors are transformed into low-dimensional dense features by embedding layer. For example, the embedding matrix of *item ID* can be represented by $\boldsymbol{V} = [\boldsymbol{v}_1; \boldsymbol{v}_2; ...; \boldsymbol{v}_K] \in \mathbb{R}^{K \times d_v}$, where K is the total number of different items, $\boldsymbol{v}_j \in \mathbb{R}^{d_v}$ is the embedding vector with dimension d_v for the j-th item. Without complicated matrix multiplication between one-hot vector and embedding matrix, embedding layer gets embedding vectors by looking up table.

The concatenation of categorical features' embedding vectors and normalized continuous features from *User Profile*, *User Behavior*, *Target Item* and *Context* form feature vectors $\boldsymbol{x}_p$, $\boldsymbol{x}_b$, $\boldsymbol{x}_t$, $\boldsymbol{x}_c$ respectively. Specially, *User behavior* contains multiple items, and the corresponding feature vector is a list of feature vectors, represented by $\boldsymbol{x}_b = [\boldsymbol{e}_1; \boldsymbol{e}_2; \ldots; \boldsymbol{e}_T] \in \mathbb{R}^{T \times d_e}$, where $\boldsymbol{e}_t$ which represents the t-th behavior's feature vector is a concatenation of multiple feature vectors, d_e is the dimension of $\boldsymbol{e}_t$, T is the length of user behavior. Note that, for each user, the number of behaviors is different and T is variable. *User Behavior* and *Target Item* are in the same feature space and share same embedding matrices for reducing memory space.

Multiple Layer Perceptron (MLP)

All the feature vectors are concatenated to form a complete representation of instance and then fed into MLP with fully connected layers. The hidden layer is activated by PRelu (He et al. 2015) and the final output layer is activated by sigmoid function for binary classification task. The input length of MLP needs to be fixed, thus the feature vector list of *User Behavior* $\boldsymbol{x}_b$ is needed to transform into fixed-length feature vector by pooling layer. Cross-entropy loss function is widely used along with the sigmoid function, whose logarithm component can counteract the side effect of exponent in sigmoid function. The loss for input feature vector $\boldsymbol{x} = [\boldsymbol{x}_p, \boldsymbol{x}_b, \boldsymbol{x}_t, \boldsymbol{x}_c]$ and click label $y \in \{0, 1\}$ is:

$$L_{target} = -\frac{1}{N} \sum_{(\boldsymbol{x}, y) \in \mathbb{D}} (y \log(f(\boldsymbol{x})) + (1 - y) \log(1 - f(\boldsymbol{x}))), \tag{1}$$

where $\mathbb{D}$ is the training set with N examples in total, $f(\boldsymbol{x})$ is the prediction result outputs by MLP.

The structure of Deep Match to Rank

It is difficult for the basic structure of deep learning based CTR model to capture the user-to-item relevance by implicit feature interactions. In DMR, we propose two sub-networks, User-to-Item Network and Item-to-Item Network, to model the user-to-item relevance, which can enhance the performance of personalized CTR model. The structure of DMR is illustrated in Figure 1.

User-to-Item Network Following the representation form in matrix factorization based matching methods, User-to-Item Network models the relevance between user and target item directly by inner product of the corresponding representation, which can be viewed as a kind of feature interaction between user and item.

To obtain the user representation, we resort to the user behavior feature. Users do not show their interest explicitly in the settings of recommendation, while user behaviors reflect users' interest implicitly. A naïve method to represent user interest is using average pooling on the user behavior feature, which considers that each behavior contributes equally to the final user interest. However, user interest may change with time, where more recent behaviors reflect user's temporal interest better. Assigning weight to each behavior according to the occurred time may alleviate the problem, but it is also difficult to find the optimal weights.

In User-to-Item Network, we apply attention mechanism with positional encoding as query to adaptively learn the weight for each behavior, where the position of user behavior is the serial number in the behavior sequence ordered by occurred time. The formulations are as follows:

$$a_t = \boldsymbol{z}^\top \tanh(\boldsymbol{W}_p \boldsymbol{p}_t + \boldsymbol{W}_e \boldsymbol{e}_t + \boldsymbol{b}), \tag{2}$$

$$\alpha_t = \frac{\exp(a_t)}{\sum_{i=1}^{T} \exp(a_i)}, \tag{3}$$

where $\boldsymbol{p}_t \in \mathbb{R}^{d_p}$ is the t-th position embedding, $\boldsymbol{e}_t \in \mathbb{R}^{d_e}$ is the feature vector for the t-th behavior, $\boldsymbol{W}_p \in \mathbb{R}^{d_h \times d_p}$, $\boldsymbol{W}_e \in \mathbb{R}^{d_h \times d_e}$, $\boldsymbol{b} \in \mathbb{R}^{d_h}$ and $\boldsymbol{z} \in \mathbb{R}^{d_h}$ are learning parameters, α_t is the normalized weight for the t-th behavior. By weighted sum pooling, the feature vector list of *User Behavior* $\boldsymbol{x}_b$ is mapped into fixed-length feature vector, and then

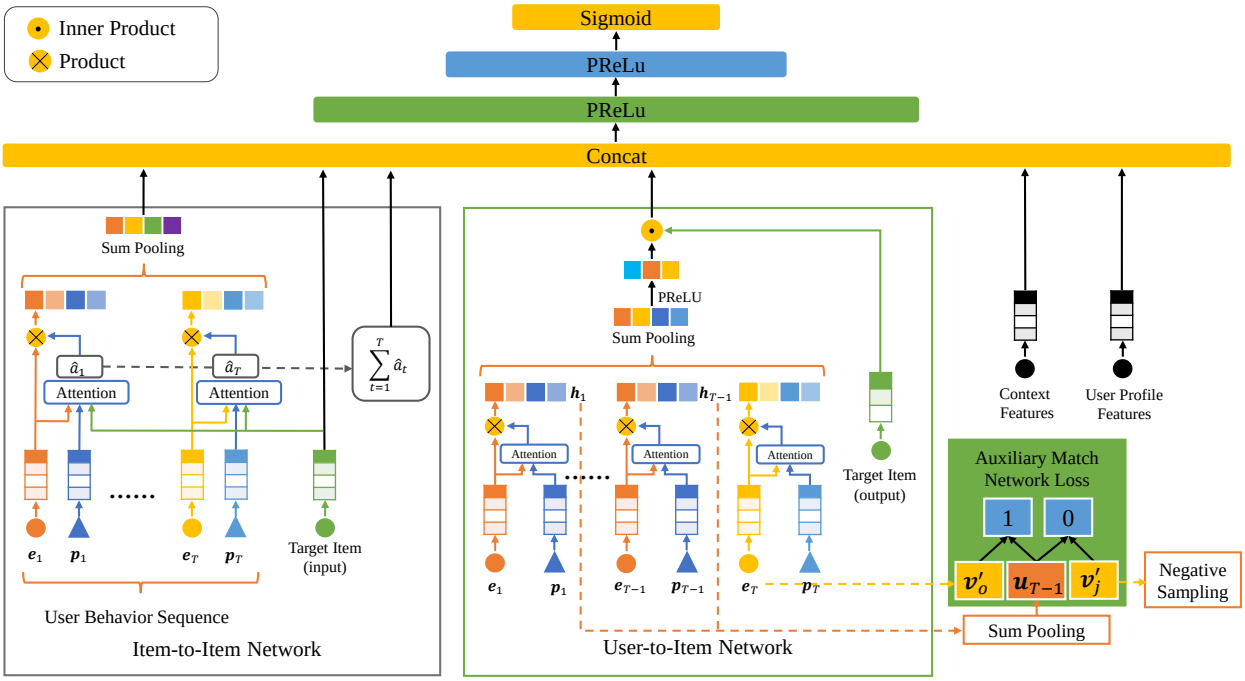


Figure 1: The structure of DMR. The input feature vector is a concatenation of embedded categorical features and normalized continuous features. DMR uses two sub-networks, User-to-Item Network and Item-to-Item Network, to model the user-to-item relevance in two forms. The two kinds of user-to-item relevance along with the user temporal interest representation are concatenated with all the others feature vectors and then fed into MLP. The final loss is composed of MLP target loss and auxiliary match network loss.

transformed by fully connected layer to match the dimension of item representation d_v. The final user representation $\boldsymbol{u} \in \mathbb{R}^{d_v}$ can be formulated as:

$$\boldsymbol{u} = g(\sum_{t=1}^{T}(\alpha_t \boldsymbol{e}_t)) = g(\sum_{t=1}^{T}(\boldsymbol{h}_t)), \tag{4}$$

where function $g(\cdot)$ represents a non-linear transformation with input dimension d_e and output dimension d_v, $\boldsymbol{h}_t$ is the weighted feature vector for the t-th behavior.

Here are three details for attention network which we leave out in the equations for simplification. First, more hidden layers can be added to have a better representation. Second, in addition to positional encoding, more context features which reflect intensity of user interest can be added to the attention network, e.g. behavior type, residence time and so on. Among the features, position influences the weight mostly in our applications. Third, the position is encoded with reverse order of behavior time to make sure that most recent behavior gets the first position.

Recurrent neural network is not used to model the user behavior sequence here, although it is good at processing sequential data, especially in NLP tasks. Different from text which strictly follows certain rules, user behavior sequence is uncertain while may be influenced by what is presented to user. Without specially designed structure, RNN is hard to improve the performance (Zhou et al. 2018). Besides, serial computation in RNN brings challenge to online serving system.

The target item representation $\boldsymbol{v}' \in \mathbb{R}^{d_v}$ is directly looked up from embedding matrix $\boldsymbol{V}' = [\boldsymbol{v}'_1; \boldsymbol{v}'_2; ...; \boldsymbol{v}'_K] \in \mathbb{R}^{K \times d_v}$. $\boldsymbol{V}'$ is a separate embedding matrix for target item, not sharing embedding with embedding matrix $\boldsymbol{V}$ for input feature in *User Behavior* and *Target Item*. To distinguish the two embedding matrices, we call $\boldsymbol{V}$ the input representation and $\boldsymbol{V}'$ the output representation of *Target Item*, as shown in Figure 1. In this way, though increasing the memory space, the model is much more expressive compared with just doubling the embedding size while sharing embedding. We will verify this conclusion in the following experiments section.

With the user representation $\boldsymbol{u}$ and target item representation $\boldsymbol{v}'$, we apply inner product operation to represent the user-to-item relevance:

$$r = \boldsymbol{u}^\top \boldsymbol{v}'. \tag{5}$$

We hope that larger r represents stronger relevance, and consequently has a positive effect on click prediction. However, from the perspective of back-propagation, it is not easy to ensure this only by the supervision of click label. Besides, the learning of parameters in embedding matrix $\boldsymbol{V}'$ fully relies on the only relevance unit r. Based on these, we propose an auxiliary match network which introduces label from user behavior to supervise the learning of User-to-Item Network.

The task of the auxiliary match network is to predict the

T-th behavior based on previous $T-1$ behaviors, which is an extreme multi-classification task. Following the form of user representation $\boldsymbol{u}$, we can obtain the user representation for the first $T-1$ user behaviors, denoted by $\boldsymbol{u}_{T-1} \in \mathbb{R}^{d_v}$. The probability that user with the first $T-1$ behaviors click item j next can be formulated with the softmax function as:

$$p_j = \frac{\exp(\boldsymbol{u}_{T-1}^\top \boldsymbol{v}'_j)}{\sum_{i=1}^{K} \exp(\boldsymbol{u}_{T-1}^\top \boldsymbol{v}'_i)}, \tag{6}$$

where $\boldsymbol{v}'_j \in \mathbb{R}^{d_v}$ is the output representation for the j-th item. The output representation $\boldsymbol{V}' \in \mathbb{R}^{K \times d_v}$ for target item can be viewed as the parameters in the softmax layer. With cross-entropy as loss function, we have the loss as follows:

$$L_{aux} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} y_j^i \log(p_j^i), \tag{7}$$

where $y_j^i \in \{0, 1\}$ represents the label of the j-th item for instance i and p_j^i is the corresponding prediction result, K is the total number of different classes i.e. items. $y_j^i = 1$ only when item j is the T-th behavior in user behavior sequence.

However, the cost of computing p_j in Equation (6) is huge, which is proportional to the total number of items K. To efficiently train such a classification task with millions of classes, we apply the negative sampling technique (Mikolov et al. 2013) which samples negative examples from the background distribution. We define the auxiliary match network loss with negative sampling as:

$$\begin{aligned} L_{NS} = -\frac{1}{N} \sum_{i=1}^{N} (&\log(\sigma(\boldsymbol{u}_{T-1}^\top \boldsymbol{v}'_o)) \\ &+ \sum_{j=1}^{k} \log(\sigma(-\boldsymbol{u}_{T-1}^\top \boldsymbol{v}'_j))), \end{aligned} \tag{8}$$

where $\sigma(\cdot)$ is the sigmoid function, $\boldsymbol{v}'_o$ is the positive example, $\boldsymbol{v}'_j$ is the negative example, k is the number of negative examples from sampling, which is quite small compared with K. The final loss can be formulated as:

$$L_{final} = L_{target} + \beta L_{NS}, \tag{9}$$

where β is a hyper-parameter to balance the two parts of loss.

By means of introducing label from user behavior, the auxiliary match network can push larger r to represent stronger relevance, and help train embedding matrix $\boldsymbol{V}'$ and other parameters better. Another way to understand User-to-Item Network is that ranking model and matching model are trained jointly in a uniform model, where the matching method is the auxiliary match network. At the stage of matching, candidates are generated by multiple matching methods whose scores are not comparable, and each candidate only has the relevance score of corresponding matching method. Different from just feeding the matching scores into MLP as feature, User-to-Item Network is able to obtain the user-to-item relevance score given arbitrary target item and the relevances are comparable in a uniform way.

Item-to-Item Network In addition to calculating the user-to-item relevance directly, we propose Item-to-Item network to represent the relevance in an indirect way. We first model the similarity between user interacted items and target item, and further sum them up to obtain another kind of user-to-item relevance. To make the relevance representation more expressive, we use attention mechanism other than inner product used in User-to-Item Network to model the item-to-item similarity. With user interacted item, target item and positional encoding as input, the item-to-item similarity is formulated as:

$$\hat{a}_t = \hat{\boldsymbol{z}}^\top \tanh(\hat{\boldsymbol{W}}_c \boldsymbol{e}_c + \hat{\boldsymbol{W}}_p \boldsymbol{p}_t + \hat{\boldsymbol{W}}_e \boldsymbol{e}_t + \hat{\boldsymbol{b}}), \tag{10}$$

where $\hat{\boldsymbol{e}}_c \in \mathbb{R}^{d_e}$ is the feature vector of target item, $\hat{\boldsymbol{p}}_t \in \mathbb{R}^{d_p}$ is the t-th position embedding, $\hat{\boldsymbol{e}}_{\boldsymbol{t}} \in \mathbb{R}^{d_e}$ is the feature vector for the t-th behavior, $\hat{\boldsymbol{W}}_c \in \mathbb{R}^{d_h \times d_e}$, $\hat{\boldsymbol{W}}_p \in \mathbb{R}^{d_h \times d_p}$, $\hat{\boldsymbol{W}}_e \in \mathbb{R}^{d_h \times d_e}$, $\hat{\boldsymbol{b}} \in \mathbb{R}^{d_h}$ and $\hat{\boldsymbol{z}} \in \mathbb{R}^{d_h}$ are learning parameters. The sum of the item-to-item similarities between user behaviors and target item forms another kind of the user-to-item relevance:

$$\hat{r} = \sum_{t=1}^{T} \hat{a}_t. \tag{11}$$

By weighted sum pooling, the feature vector list of *User Behavior* $\boldsymbol{x}_b$ is transformed into fixed-length feature vector $\hat{\boldsymbol{u}}$ to form the temporal interest representation which is target-relevant. The formulations are as follows:

$$\hat{\alpha}_t = \frac{\exp(\hat{a}_t)}{\sum_{i=1}^{T} \exp(\hat{a}_i)}, \tag{12}$$

$$\hat{\boldsymbol{u}} = \sum_{t=1}^{T} (\hat{\alpha}_t \hat{\boldsymbol{e}}_t), \tag{13}$$

where $\hat{\alpha}_t$ is the normalized weight for the t-th behavior. Different from user representation $\boldsymbol{u}$, the target-relevant user representation $\hat{\boldsymbol{u}}$ varies with respect to the target item, which is also fed into MLP for further feature interactions. With the local activation ability of attention mechanism, behaviors more relative to the target item are weighted higher and dominate the target-relevant user representation $\hat{\boldsymbol{u}}$.

The two forms of user-to-item relevance r and $\hat{r}$ along with the user temporal interest $\hat{\boldsymbol{u}}$ are concatenated with other input feature vectors together to feed into MLP. And the final input of MLP is represented by $\boldsymbol{c} = [\boldsymbol{x}_p, \boldsymbol{x}_t, \boldsymbol{x}_c, \hat{\boldsymbol{u}}, r, \hat{r}]$.

Experiments

In this section, we introduce the experiments on both public and industrial datasets. First, we give the datasets and compared methods. Next, we show the experiment results of DMR compared with the state-of-the-art methods. Besides, we conduct ablation study on DMR to verify the effectiveness of our proposed techniques.

Datasets

Public Dataset Alimama Dataset[2] contains ad display and click logs randomly sampled from Taobao in 8 days. It contains 26 million logs with 1.14 million users and 0.84 million

[2]https://tianchi.aliyun.com/dataset/dataDetail?dataId=56

Table 1: Model comparison on public and industrial datasets. Relative Improvement (RI) is based on LR.

Method	Public		Industrial	
	AUC	RI	AUC	RI
LR	0.6394	0.00%	0.6739	0.00%
Wide&Deep	0.6408	0.22%	0.6783	0.65%
PNN	0.6415	0.33%	0.6793	0.80%
DIN	0.6416	0.34%	0.6856	1.74%
DIEN	0.6420	0.41%	0.6871	1.96%
DMR	0.6447	0.83%	0.6921	2.70%

items. The logs in the first 7 days is used for train set, and logs in the last day is used for test set.

Industrial Dataset We collect impression and click logs from our online recommender system[3] in Alibaba to form the industrial dataset. We use logs in the first 14 days as train set and logs in the following day as test set. The whole dataset contains 1.18 billion examples with 10.9 million users and 48.6 million items.

Compared Methods

- ***LR*** Logistic regression (LR) is a classical linear model which can be regarded as a shallow neural network. Linear model usually needs manual feature engineering to perform better. Here we add the cross-product of *User Behavior* and *Target Item*.
- ***Wide&Deep (Cheng et al. 2016)*** Wide&Deep has a wide part and a deep part, which combines the advantage of linear model and non-linear deep model with joint training. The wide part in our implementation is same with LR above.
- ***PNN (Qu et al. 2016)*** PNN (Qu et al. 2016) introduces a product layer along with fully connected layers to explore high-order feature interactions.
- ***DIN (Zhou et al. 2018)*** DIN represents user interest with regard to the target item by adaptively learning the attention weight. Notice that, without the two user-to-item relevances and the usage of positional encoding, our model almost becomes DIN model.
- ***DIEN (Zhou et al. 2019)*** DIEN models the evolution of user interest with respect to the target item by two-layer GRU.

Results on Public and Industrial Datasets

In the experiments on public dataset, we set learning rate to 0.001, batch size to 256, item embedding size to 32, max length of user behavior sequence to 50. The dimension of hidden layer in MLP are 512, 256, 128 respectively. Besides, the number of negative samples in the auxiliary match network is set to 2000 and the weight of auxiliary loss β is set to 0.1. In the experiments on industrial dataset, the hyper-parameters are almost same as above except that we set item embedding size to 64.

Table 1 shows the comparison results on both public and industrial datasets. We use Area Under ROC (AUC) as evaluation metric, which is widely used in binary classification problems. All experiments are repeated 5 times and the average results are reported.

LR performs worse than Wide&Deep and other deep learning based models significantly, which demonstrates the effectiveness of non-linear transformation and high-order feature interactions in deep neural network. PNN benefits from product layer for better feature interactions and preforms better than Wide&Deep.

Capturing user interest is very important for recommendation settings where users do not show their interest explicitly. Among the deep learning based models, Wide&Deep and PNN perform worst, especially on the industrial dataset, which verifies the importance of extracting user interest from user behavior. DIN represents the user interest with regard to target item, but neglect to consider the sequential information in user behavior. DIEN performs better than DIN, mainly attributed to the two-layer GRU structure which captures the evolution of user interest.

Based on user interest representation, DMR steps further to capture user-to-item relevance in two forms with User-to-Item Network and Item-to-Item Network respectively. With the representation of relevance, DMR takes a full account of user's personalized preference on target item and beats all the compared methods significantly, including LR, Wide&Deep, PNN, DIN and DIEN.

Table 2: Results of ablation study on public and industrial datasets. Relative Improvement (RI) is based on DMR.

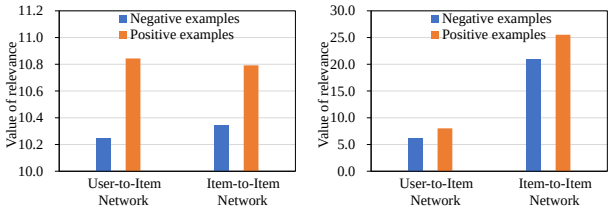
Method	Public		Industrial	
	AUC	RI	AUC	RI
DMR I2I[a]	0.6424	-0.36%	0.6901	-0.29%
DMR U2I[b]	0.6444	-0.05%	0.6916	-0.07%
DMR-NO-AM[c]	0.6432	-0.23%	0.6905	-0.23%
DMR-NO-PE[d]	0.6438	-0.14%	0.6890	-0.45%
DMR-Double[e]	0.6439	-0.12%	0.6910	-0.16%
DMR	0.6447	0.00%	0.6921	0.00%

[a] DMR with Item-to-Item Network alone
[b] DMR with User-to-Item Network alone
[c] DMR without the auxiliary match network
[d] DMR without positional encoding in User-to-Item Network
[e] DMR with double-sized item embedding while sharing the embedding

Ablation Study

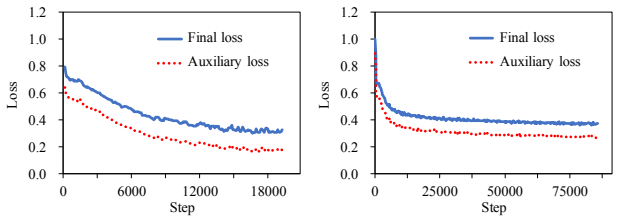
In this section, we conduct ablation study on DMR to verify the effectiveness of our proposed techniques. Table 2 gives the comparison results of DMR with different components on both public and industrial datasets.

[3] https://www.1688.com



(a) Public dataset (b) Industrial dataset

Figure 2: User-to-item relevances on public and industrial datasets.



(a) Public dataset (b) Industrial dataset

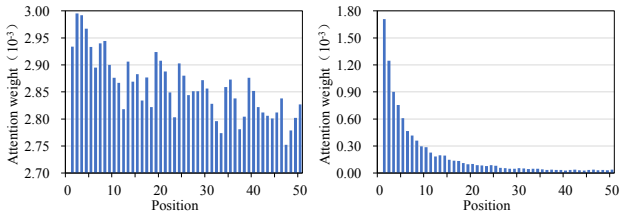
Figure 3: Learning curves on public and industrial datasets. Both losses are training losses, where auxiliary loss is βL_{NS}, and β is set to 0.1.

Effectiveness of User-to-Item Relevance Representation To make the representation more expressive, we apply different operations to model the user-to-item relevance. User-to-Item Network uses inner product based operation, and Item-to-Item Network uses attention network to calculate the relevance. As can be seen from Table 2, the combination of two sub-networks performs better than alone, which demonstrates the effectiveness of two different kinds of user-to-item relevance, and the two forms of user-to-item relevance are complementary, not redundant.

We explore the value of two user-to-item relevances on both public and industrial datasets, as shown in Figure 2. The values are averaged on positive and negative examples respectively. As expected, the user-to-item relevance is higher on positive examples than on negative examples, meaning that our model on user-to-item relevance is reasonable.

Effectiveness of Auxiliary Match Network From Table 2 we can see that DMR obtains better performance than DMR without the auxiliary match network. The auxiliary match network introduces label from user behavior to supervise the training, and push larger inner product between user representation and item representation to represent higher relevance. Figure 3 shows the learning curves of DMR on both public and industrial datasets. We can see that the target loss L_{target} and auxiliary loss L_{NS} decreases together, meaning that the joint training of matching and ranking works.

User-to-Item Network uses extra embedding matrix $\boldsymbol{V}'$ to represent target item, which can be viewed as the parameters in softmax layer of the auxiliary match network. We try to double the embedding size of item and share embedding to obtain DMR-Double, whose number of parameters is same with DMR. And we observe that DMR performs better than DMR-Double as shown in Table 2, which verifies the effectiveness of the separate embedding matrix.



(a) Public dataset (b) Industrial dataset

Figure 4: Attention weights with regard to different positions in Item-to-Item Network on public and industrial datasets. More recent behavior has smaller position number.

Effectiveness of Positional Encoding As shown in Table 2, DMR performs better than DMR without positional encoding in User-to-Item Network. With positional encoding, DMR considers sequential information in user behavior sequence and extracts temporal interest of user. On the industrial dataset, DMR without positional encoding in User-to-Item Network even performs worse than DMR without the whole User-to-Item Network, meaning that it is hard to fit User-to-Item Network without positional encoding.

We explore the attention weights of different positions in user behavior sequence. Figure 4 shows the averaged weights in Item-to-Item Network on both public and industrial datasets. Though the attention weight is influenced by multiple factors, there is an overall trend that more recent behavior gets higher activated weights as expected, especially on the industrial dataset.

Results from Online A/B Testing

We conduct online A/B testing in our recommender system in Alibaba. DMR improves CTR by 5.5% and clicks per user by 12.8% relatively compared with DIN, which is the last version of CTR model in our system. Based on the significant promotion, we have deployed DMR online to serve the recommendations.

Conclusions

In this paper, a novel model named Deep Match to Rank (DMR) is proposed in the field of personalized CTR prediction to capture the relevance between user and item. Inspired by the thought of collaborative filtering in matching methods, DMR uses User-to-Item Network and Item-to-Item Network to model the user-to-item relevance respectively. An auxiliary match network is proposed to help train User-to-Item Network better. To the best of our knowledge, DMR is the first model that jointly trains matching and ranking in CTR prediction. Besides, positional encoding is introduced to model temporal interest of user in both sub-networks. DMR improves the performance significantly and has been deployed in our online recommender system in Alibaba.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Cheng, H.-T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, 7–10. ACM.

Cho, K.; van Merrienboer, B.; Bahdanau, D.; and Bengio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, 103–111.

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, 191–198. ACM.

Feng, Y.; Lv, F.; Shen, W.; Wang, M.; Sun, F.; Zhu, Y.; and Yang, K. 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.

Guo, H.; Tang, R.; Ye, Y.; Li, Z.; and He, X. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 1725–1731. AAAI Press.

He, X., and Chua, T.-S. 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. ACM.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

He, X.; He, Z.; Song, J.; Liu, Z.; Jiang, Y.-G.; and Chua, T.-S. 2018. Nais: Neural attentive item similarity model for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 30(12):2354–2366.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2016. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* (8):30–37.

Koren, Y. 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. ACM.

Linden, G.; Smith, B.; and York, J. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* (1):76–80.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Qu, Y.; Cai, H.; Ren, K.; Zhang, W.; Yu, Y.; Wen, Y.; and Wang, J. 2016. Product-based neural networks for user response prediction. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, 1149–1154. IEEE.

Rendle, S. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*, 995–1000. IEEE.

Sarwar, B. M.; Karypis, G.; Konstan, J. A.; Riedl, J.; et al. 2001. Item-based collaborative filtering recommendation algorithms. *Www* 1:285–295.

Shan, Y.; Hoens, T. R.; Jiao, J.; Wang, H.; Yu, D.; and Mao, J. 2016. Deep crossing: Web-scale modeling without manually crafted combinatorial features. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 255–262. ACM.

Su, X., and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 6000–6010. Curran Associates Inc.

Wang, R.; Fu, B.; Fu, G.; and Wang, M. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, 12. ACM.

Xiao, J.; Ye, H.; He, X.; Zhang, H.; Wu, F.; and Chua, T.-S. 2017. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 3119–3125. AAAI Press.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 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. ACM.

Zhou, G.; Mou, N.; Fan, Y.; Pi, Q.; Bian, W.; Zhou, C.; Zhu, X.; and Gai, K. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 5941–5948.

Zhu, H.; Li, X.; Zhang, P.; Li, G.; He, J.; Li, H.; and Gai, K. 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. ACM.