An Attention-Based User Preference Matching Network for Recommender System

YUCHEN LIU, TAN YANG, AND TAO QI
School of Software Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
Corresponding author: Yuchen Liu (lycxz94@foxmail.com)

ABSTRACT Click-through rate prediction (CTR) is an essential task in recommender system. The existing methods of CTR prediction are generally divided into two classes. The first class is focused on modeling feature interactions, the second class is focused on solving time-series problems. However, the existing models of the second class are not able to handle time-series problems with user feedback information, so we propose PMN to solve this kind of problem. To be able to take full advantage of historical user behavior along with the user feedback, PMN uses the attention mechanism to get the user historical behavior representation and the user preference representation from the original input. Specially, user preference representation is derived from the user feedback information and it explicitly shows the user’s attitude towards the candidate, which greatly improve the model performance. Finally, we introduced user preference baselines to solve the problem of inconsistent scoring standards for different users. In this paper, we focus on the CTR prediction modeling in the scenario of video recommendation in Video On Demand (VOD) service. Experimental results on multiple data sets have shown that our PMN model is effective.

INDEX TERMS Recommender system, click-through rate prediction, deep learning.

I. INTRODUCTION
The Internet content industry has developed rapidly recently, such as online news, online video sites, and online music platforms. In order to improve the user experience of a content platform, a recommender system is usually required. In order to model user preference, recommender systems generally track different behaviors of users, such as browsing history, viewing history, etc. The general recommender system follows the steps of candidate generation and candidate ranking [1]. In this paper, we will focus on the process of candidate ranking. At present, there are roughly two classes of mainstream ranking models in CTR research field: the first class developed from FM (Factorization Machine) is focused on modeling feature interactions [2]–[8]; the second class is focused on processing time-series data [9], [10]. FM-type model input features are often extracted from the original viewing records, click records, search records, these features are designed and produced by human; while the second class of models generally directly handle users’ browsing records, these data are more primitive and contain more comprehensive information, but also contains more noise.

DIN (Deep Interest Network) [9] proposed by Alibaba is designed to deal with time-series problem appeared in e-commerce scenarios. However, DIN cannot well handle the scenario which user provides feedback. We know that user feedback is an important piece of information in the recommender system. If this kind of information can be fully utilized, the final recommendation accuracy will be improved a lot. In this paper, the PMN model we proposed uses a vector to explicitly represent the user preference over the candidate video. PMN first use the attention mechanism to calculate the similarity weights between the candidate video and the user’s watch history, and then based on the similarity weights, PMN take a weighted sum pooling of the user’s history feedback to get a representation of the user’s preference over the candidate video. The intuition of the above approach is that the more similar the candidate video is to the video that user likes, the more likely the user is to watch the candidate video, and conversely, the more similar a candidate video is to a video that user dislikes, the less likely the user is to watch the candidate video.

We also observe that in some scenarios, the scores of different users are quite different. For example, in the scene where the scores are distributed from 0 to 10, some users tend to give a high score that cause the average score to be
around 7, while some users are strict cause the average score around about 4 points. It is obviously unreasonable to use a uniform standard to determine whether a user likes a certain video. Therefore, we introduce the user preference baseline. The user preference baseline is the average of user historical feedback. By comparing the difference between the current preference and the preference baseline, PMN can give a more accurate judgment.

In summary, this paper makes the following contributions:

- We propose PMN to address the case where user provides feedback in a time-series problem. PMN incorporates the representation of the user’s historical behavior and the user’s preference over the candidate video to give a CTR prediction.
- Different users have different scoring standards and cannot be generalized. Thus, we introduce user preference baseline. By measuring the difference between the user’s current preference and the baseline, the user’s truly preference over the candidate is more accurately reflected.
- We not only performed a series of experiments on both public datasets and real word IPTV dataset to prove the effectiveness of our model, but also offers good model explainability.

II. RELATED WORK

CTR prediction has attracted researchers’ attention for some time, during which the model has undergone changes from simple to complex, from shallow to deep. In order to achieve better performance, many models are dedicated to studying how to effectively modeling feature interactions. We know that the combination of higher-order features is essential for a good performance. For example, a 20-year-old male user who likes watching sci-fi action movies can be described as gender=男, genre=Action, science Fiction, age=20 which is an order-3 feature. The first model that introduced learning feature interactions is FM (Factorization Machine) [2], however, FM is a shallow model that can only learn second-order feature interactions. FNN (Factorization-Machine Supported Neural Networks) [8] first uses the FM component to learn the feature embedding, then uses the learned feature embedding vectors to initialize the MLP (Multilayer Perceptron), and finally the MLP completes the whole learning process. PNN (Product-based Neural Networks) [7] imposes a product layer between the embedding layer and the first hidden layer. Wide&Deep [4] learns low- and high-order feature interactions simultaneously. However, the “wide” part of the W&D has a need for the handcrafted feature engineering. DeepFM [3] integrates FM and Deep Neural Network (DNN). The FM part models low-order feature interactions and the DNN part models the high-order feature interactions. Unlike the Wide&Deep model, DeepFM can be trained end-to-end without any feature engineering. These above models have one thing in common is that they try to model high-order feature interactions from the original features. Since these models all come from FM, we call them FM-type model in this paper.

There are many kinds of time-series data in the real world, such as users’ browsing records or users’ watch history and the FM-type model cannot handle this kind of data very well. Data such as user browsing records are closer to the original data, which means that the data contains the most comprehensive information, but it also means that the data contains much noise, so how to fully mine the user preference information contained in these time-series data becomes a key issue in CTR prediction. Alibaba Group proposed the DIN model to handle the time-series data in e-commerce industry. DIN adaptively calculates the representation vector of user interests by taking into consideration the relevance of historical behaviors given a candidate ad. DIN regards user behavior as interest directly, however user’ interest is difficult to be fully reflected through explicit behaviors.

The attention mechanism we used in the PMN model is originated from the Neural Machine Translation (NMT) problem [17] in the NLP (Natural Language Processing) field. The Attention mechanism skillfully solves the problem of the alignment between the source language and the target language in NMT, and brings interpretation to the parameters learned in the deep model. A subsequent paper Attention is all you need [11] proposed by Google abstracted the Attention mechanism into a QKV model. Q is represented for Query, K and V is represent for Memory. Query and Memory have different meanings in different scenarios. In the video recommendation scenario which we will discuss in the rest of the paper, Query is the candidate video, and Memory is the user’s watch history. In a Question&Answer problem, Query is the question the user asked based on the article, and Memory is article content. The success of the Attention mechanism stems from its ability to focus on the useful information for the current Query in a complex context (Memory) and extract this part of the information for the following process.

III. PREFERENCE MATCHING NETWORK

In this section, we will describe the PMN model in detail. PMN considers the rich viewing history of users, as well as the history feedback records of each user and the baseline of user preference. User feedback is a very important information in a recommender system. Taking full advantage of user feedback can greatly improve the accuracy of a recommender system.

A. MODEL OVERVIEW

In some video recommendation scenarios, users will leave ratings after watching the video. In order to make reasonable recommendations to users based on their historical viewing records and historical feedback scores, our proposed model PMN is used to solve the problem of ranking candidate sets in a recommender system.

The intuition behind the structural design of the PMN model is simple: More similar the candidate video is to the video that user likes, the more likely the user is to watch the candidate video. Therefore, in order to enable the model to perform CTR prediction based on the above assumption,
we use the attention mechanism to calculate the similarity weights between different historical viewing videos and the candidate video, which is performed by Attention Unit shown in figure 1. The similarity weights can tell which video the user has watched is more important according to the current candidate and which is less important. Then based on the similarity weights we get the following two vectors:

1. User historical behavior representation: the representation vector of elements similar to candidate shows in user’s watch history.
2. User preference representation: the representation vector of user’s preference over the candidate by taking into consideration the relevance between the user’s watch history and the candidate video.

For different candidates, the above two vectors can accurately reveal the user’s attitudes hidden in the user’s historical behaviors and user’s historical feedback. Finally, we concatenate the output features of the Attention Layer shown in Figure 1 and input them into the MLP Layer to get the final prediction result.

B. INPUT LAYER
The input of the model during the training phase includes three parts, which are the user’s watch history, candidate videos, and labels.

The user watch history is $S = [s_1, s_2, s_3, \ldots, s_n]$, where $n$ represents the length of the sequence, where $s_i$ represents a single interaction record. Specially, $s_i = [x_1, x_2, x_3, \ldots, x_m]$ where $m$ represents total feature fields, $x_i$ represents an i-th feature. $x_i$ is a one-hot or multi-hot vector if the i-th field is categorical. $x_i$ is a scalar value if i-th field is numerical. A sample of $s_i$ is as follows: [VideoID = 58559, genre = {Action, Crime}, Actor = {Christian Bale, Michael Caine}, Director = {Christopher Nolan}, userFeedback = 4.8].

C. EMBEDDING LAYER
Since the feature representations of the categorical features are very sparse and high-dimensional, we use the embedding layer to map them into low-dimensional spaces. We set $W$ as the embedding matrix, $W = [w_1, w_2, w_3, \ldots, w_n]$, where $w_i$ is the embedding dictionary for the i-th feature field. For different types of features, our mapping method is as follows:

- For one-hot categorical variables, we map them as:
  \[ e_i = w_i \cdot x_i \]  
  where $e_i$ is the embedded vector, $w_i$ is the embedding matrix of the i-th feature, and $x_i$ is the one-hot vector of the i-th feature

- For multi-hot categorical variables, such as genre = {Action, Crime}, we use the following formula:
  \[ e_i = \frac{1}{n} (w_i \cdot x_i) \]  
  where $n$ is the number of values that a sample has for i-th field and $x_i$ is the multi-hot vector representation for this field.

- For numerical variables:
  \[ e_i = w_i \cdot x_i \]  
  where $w_i$ is the embedding vector for field $i$, and $x_i$ is a scalar value.

As shown in Figure 1, after obtaining the embedding of all features, the video embedding vector would be a concatenation of all features belonging to the video.

D. ATTENTION LAYER
The output of the Attention layer includes four features: user historical behavior representation $E_h$, user preference representation $E_f$, user preference baseline $E_{af}$ and the difference between $E_f$ and $E_{af}$.
1) USER HISTORICAL BEHAVIOR REPRESENTATION
Whether the user clicks on the candidate video is related to his historical behaviors. Now in order to use a fixed-length vector to represent the user’s watch history, there are two ways:

- By averaging all user historical behaviors to get a fixed-length vector. This is the simplest but not the most reasonable approach. For different candidate videos, it is obviously unreasonable to use the same vector to represent user preferences. There are many reasons why a user clicks on a video. It may be because the user likes the genre of the video, or because the user likes one of the actors, or because the user appreciates the director. Obviously, a static vector cannot simultaneously represent different interest preference information according to different candidate video.

- Through the attention mechanism, we obtain the weighted sum of user’s watch history, and this vector contains elements which exist in the user watching history that are most relevant to the candidate video. The Attention Unit shown in the Figure 1 follows the classic Attention QKV model [11], we set the candidate video $e_q$ as Query, the embedded user’s watch history $[e_{k1}, e_{k2}, e_{k3}, \ldots, e_{kn}]$ as Key and Value where $e_{ki}$ is a video embedding vector and $n$ is the length of the watch history. We first calculate the similarity vector $a_{qk}$ between Query and Key, then based on $a_{qk}$ we can get the User historical behavior representation vector $E_h$:

$$a_{qk} = \frac{\exp(\text{score}(e_q, e_{ki}))}{\sum_{i=1}^{n} \exp(\text{score}(e_q, e_{ki}))} \quad (4)$$

$$\text{score}(e_q, e_{ki}) = e^T_q W e_{ki} \quad (5)$$

$$E_h = \sum_{i}^n a_{qk} e_{ki} \quad (6)$$

Compared with the first method of simple averaging, the user historical behavior representation obtained by attention can always represent the most relevant information in the user history record that is related to the candidate video and that is why we use the Attention Unit in the PMN model.

2) USER PREFERENCE REPRESENTATION
We first give the calculation formula for user preference representation vector $E_f$:

$$E_f = \sum_{n}^n a_{qk} f_i \quad (7)$$

where $f_i$ is the user feedback corresponding to the i-th user’s watch history. Intuitively, if the video that the user has seen is more similar to the candidate, the user feedback of the video should be more important.

For example, if we only consider the genre of movie, user Peter has watched science fiction movie A, romantic movie B, action movie C, and then Peter gives 5 stars, 3 stars, and 1 star to these three movies. At this time, the system recommends the science fiction movie D to peter, since the candidate movie D is very similar to the science fiction movie A that Peter watched, and Peter gave a high rating for movie A, then we think that Peter is likely to watch movie D. If the system recommends an action movie E to Peter, since Peter only scored 1 point for action movie C that he has watched, Peter may ignore the movie E.

3) USER PREFERENCE BASELINE
We noticed that each user’s rating standard is different. Some users can directly give 5 stars to his favorite items, but some users may be strict, for them 4 stars means very satisfied. How can we ensure that the user feedback embedding accurately reflects the user’s preference? Here we introduce the user preference baseline vector $E_{af}$. This baseline vector represents the average level of user historical preferences. The calculation method is to get the average value of the user historical feedback sequence $[f_1, f_2, f_3, \ldots, f_n]$:

$$E_{af} = \frac{\sum_{i=1}^{n} f_i}{n} \quad (8)$$

Difference*: we use the formula bellow to calculate the difference:

$$\text{Difference}^* = E_f - E_{af} \quad (9)$$

When we compare the vectors $E_f$ and $E_{af}$, we can get a measure of the difference between the user’s preferences for the current candidate movie and the user’s historical preference baseline, thereby more accurately reflecting the user’s preferences.

E. MLP LAYER
The input of the MLP Layer is a set of feature vectors which includes the four output features of the Attention layer, the candidate and the context feature. For final CTR prediction, we simply concatenate all of them and apply a three-layers MLP and each layer perform the following computation:

$$a^{(l+1)} = \sigma(W^{(l)} a^{(l)} + b^{(l)}) \quad (10)$$

where $l$ is the layer number and $\sigma$ is the sigmoid activation function, $W^{(l)}$, $a^{(l)}$, $b^{(l)}$ are the model weights, activations and bias at l-th layer. The last layer transforms the output vector of the previous layer to user clicking probabilities.

F. TRAINING
Our loss function is Log Loss, which is defined as follows:

$$\text{loss} = -\frac{1}{N} \sum_{i=1}^{N} (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)) \quad (11)$$

where $N$ is the size of training set, $y_i$ is the ground truth of user preference and $y_i \in \{0, 1\}$, $\hat{y}_i$ is the estimated CTR, the optimizer we use is Adam [12].

IV. EXPERIMENT
In this section, we will detail our experiments. Including the dataset we used, and the way the dataset is constructed, the evaluation metric for the experiment, and the corresponding...
analysis of each part of the model. We will focus on answering the following three questions:

Q1 How does our proposed model PMN perform on the problem of CTR prediction and whether the feedback information can bring a big improvement.

Q2 what are the influences of different model structure?

Q3 How does our model perform on a specific sample? Is our model explainable?

A. DATASET

First, we will explain how we build the dataset. We know that users’ viewing choices change as they continue to watch the show. For example, a user may have watched *The Avengers* because it’s a popular movie. After that this user becomes interested in the cast of *Iron Man* very much and watch other movies acted by Robert John Downey Jr. Therefore, we must pay attention to the chronological factor when constructing the data set.

![FIGURE 2. Labeling method.](image)

So we use the method shown in the Figure 2 to construct each sample in the training set. We follow the way of using historical information to predict future items when making the dataset, instead of randomly digging out some items in the user’s viewing history to make predictions. In terms of the length of each sample, we set a maximum length of 50, since the interests of users are often time-sensitive, we hope to make predictions based on the recent preferences of users.

Kaggle Anime: Kaggle Anime dataset contains 6,994,187 anonymous ratings of 12,294 animes made by 67,041 users. User ratings are based on a 10-point scale, which can include decimal points, and missing ratings are represented by −1. Since there are fewer missing scores, we deleted records with a score of −1. In order to adapt to the CTR prediction task, we need to convert ratings into positive and negative labels. The general approach is to choose a threshold, and then convert the score above the threshold to positive, and the score below the threshold to negative. However, we observe that under the 10-point scale, the average ratings of different users are quite different, which means that some users tend to give high ratings and some users tend to give low ratings, so it is unwise to use a threshold to decide positive and negative labels. Therefore, our approach is to first calculate the average score of each user, and then mark samples higher than the average score in the user’s rating as positive samples and samples that are lower than the average score as negative samples. We think that this can more accurately reflect the preferences of users. Dividing the training set and test set according to userID, we finally got 5,389,882 training samples (2,936,231 of positive samples and 2,452,751 negative samples) and 1,347,587 of test samples (734,529 of positive samples and 613,058 of negative samples).

Movielens [19]: Movielens dataset contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 Movielens users. Ratings are made on a 5-star scale (whole-star ratings only). To make it suitable for the CTR prediction task, we use the method of processing Kaggle Anime dataset for label generation. After dividing the data into training set and testing dataset based on userID, we finally got 780,710 training data (including 449,930 positive samples and 330,780 negative samples) and 195,339 test sets (positive samples 112,376 negative samples 82,963).

IPTV: The IPTV dataset is from IPTV service in a city from southeast China. Users can order TV series, cartoons, movies and other videos through IPTV. We were able to obtain some of users’ data through cooperation with the IPTV provider. No personally information was collected in connection with this data. The IPTV dataset contains 1,921,588 ratings of 147,820 users and there are about 20,000 unique videos which appeared during a one-month period. User ratings are based on a 5-point scale. We generated labels in the same way as we did for the previous two datasets.

B. EFFECTIVENESS COMPARISON(Q1)

1) COMPETING MODELS

We compare the proposed PMN with two classes of previous models. (A) techniques that focus on modeling high-order feature interactions, (B) methods that use attention mechanism to handle time-series data.

- Wide&Deep [4] (A): Wide&Deep learns the low- and high-order feature interactions simultaneously. It consists of two parts: 1) wide model, which handle the hand-crafted features, 2) deep model, which automatically learn the high-order feature interactions. We follow the practice in [3] to take cross-product of user behaviors and candidates as wide part input.

- PNN [7] (A): PNN imposes a product layer between the embedding layer and the first hidden layer to get the high-order feature interactions of the user behaviors.

- DeepFM [3] (A): DeepFM can be viewed as an improved version of Wide&Deep by using factorization machines to model the wide part input features.

- DIN [9] (B): Comparing Figure 3 with Figure 1, the biggest difference between our proposed model PMN and DIN is that PMN models user feedback. DIN assumes that the user’s historical behavior sequence represents the user’s interests. This assumption is not applicable in the scenario where some users’ historical behavior sequence includes both the user’s favorite items and the user’s disliked items. In response to this problem, PMN uses the attention mechanism to model user feedback to more accurately reflect user preferences.
TABLE 1. Effectiveness comparison of different models.

| Model       | Movielens-1m | Kaggle Anime | IPTV  |
|-------------|--------------|--------------|-------|
| Wide&Deep   | 0.7915       | 0.7529       | 0.7630|
| PNN         | 0.7928       | 0.7537       | 0.7624|
| DeepFM      | 0.7935       | 0.7568       | 0.7638|
| DIN         | 0.7513       | 0.7256       | 0.7281|
| **PMN**     | **0.7975**   | **0.7753**   | **0.7663**|

DIN cannot use user feedback information due to the architecture of the model. DIN is designed to calculate the similarity between the candidate and the user historical behaviors. So the features of the items in user historical behaviors must be consistent with the features of the candidate. However, user feedback can only appear in user historical behaviors, and the candidate will never have user feedback information. Different from DIN, user feedback information is retained in the input of Wide&Deep, PNN, and DeepFM, which have the same input with PMN.

We use AUC [13] as the evaluation metric. Area under the ROC (AUC) measures the probability that a CTR predictor will assign a higher score to a randomly chosen positive item than a randomly chosen negative item.

Table 1 shows the performance of the different models on the Movielens dataset, Kaggle Anime dataset, and the IPTV dataset. We have the following collusions: (1) The user feedback information is a very important type of information. DIN has the worst performance cause it’s loss of user feedback information. (2) PMN is designed to deal with the time-series problem especially the scenario with user feedback and PMN has the best perform. In PMN, user preference vector calculated through the attention mechanism can accurately reflect the user preference of the candidate, while other CTR models always get the same fix-length vector before the MLP layer when facing different candidates. Therefore, the approach that PMN use to deal with user feedback information determines PMN performs better than other models in face of this kind of scenario.

C. MODEL STRUCTURE ANALYSIS(Q2)

As shown in Figure 1, the information entered into the MLP part includes the following five parts. We will explore the impact of the following five parts on the final results: User historical behavior representation $E_h$, User preference representation $E_f$, User preference baseline $E_{af}$, the Difference between $E_f$ and $E_{af}$, and the candidate video $E_t$.

In order to explore the contribution of each part to the final accuracy, we designed four sets of comparative experiments as follows.

- **PMN-1**: Features entered into the classifier include $E_h, E_t$. At this time we do not consider user feedback, which is the same as DIN, so PMN-1 model can be considered as DIN.
- **PMN-2**: Features entered into the classifier include $E_t, E_f, E_{af}$, $(E_f - E_{af})$. PMN-2 only considers the user feedback information and the different user scoring standards, while the user’s historical behavior is not taken into consideration.
- **PMN-3**: Features entered into the classifier include $E_h, E_t, E_f$. PMN-3 considers the user’s historical behavior and user feedback but does not take into account the users’ different scoring standards.
- **PMN-4**: Features entered into the classifier include $E_h, E_t, E_f, E_{af}$, $(E_f - E_{af})$. The PMN-4 model is the final PMN model, and its structure is shown in Figure 1.

As we can see from Table 2, PMN-1 shows the worst performance which does not use the user feedback information. Other models which take user feedback into consideration...
all significantly outperform PMN-1 on these three datasets. Compared with PMN-1, PMN-2 replaces the $E_h$ vector with features related to user feedback. The average performance of PMN-2 on these three datasets is better than PMN-1, which indicates that user feedback information is more important. PMN-3 performs better than PMN-1 and PMN-2, which shows that the user feedback information and user historical behavior information are both essential in the CTR prediction. Compared to PMN-3, PMN-4 additionally considered different scoring standards between different users, and PMN-4 achieves 0.39%, 1.45%, 0.27% AUC gain on these datasets over PMN-3. In addition, the improvement on Kaggle Anime is larger than the other two datasets. This is because the Kaggle Anime dataset is based on a 10-point rating scale, which results in the difference in ratings between users will be more obvious, while the other two dataset is under a 5-point rating scale, and the user’s average score is relatively concentrated. Therefore, PMN-4 gets a larger improvement on the Kaggle Anime dataset.

### D. EXPLAINABLE RECOMMENDATIONS (Q3)

As shown in Figure 4, the candidate recommendation targets are **Beauty and the Beast**, and the movies in the watch history most related to it are **Chicken Run**, **Dinosaur**, and **Toy Story**. However, this user doesn’t seem to like watching animation movies very much, and he gives low ratings to the above three animation movies. So, the CTR of the **Beauty and the Beast** given by the model is estimated to be 0.224 (Our model gives a score between 0-1). A larger number means a higher probability of predicting a user click, which means the model predicated this user may not watch **Beauty and the Beast**.

As shown in Figure 5, this user has watched movies such as **The Silence of the Lambs**, **Indiana Jones**, **The Matrix**. In the user’s watching history, films most relevant to the candidate movie **Indiana Jones and the Raiders of the Lost Ark** are **West World**, **The Good the Bad the Ugly**, **Indiana Jones and the Temple of the Doom**. Since the user gave high ratings to these three movies, the model also gave a high estimated CTR score: 0.910.

### V. CONCLUSION AND FUTURE WORK

In this paper, we explored how to solve the problem of CTR estimation in scenarios with user feedback. Our proposed model PMN use the attention mechanism to model user feedback information. The feedback of items with different relevance have different degrees of importance: the more similar the item to the candidate, the more important the corresponding feedback is. The final weighted sum of the user history feedback represents the user’s preference towards candidate items. At the same time, we also introduced the concept of user preference baseline to distinguish different user rating standards. In the end, we explored the performance of different model structure, and proved that the PMN model incorporating all features is the most effective.

For future work, we want to explore other different structure of the PMN model, e.g. there may be a more effective NN structure instead of the three-layer MLP. Besides, we are also interested in incorporating implicit feedback into our model. There are many kinds of implicit feedback in the real word, such as the user’s viewing time of a movie or the user’s comment of a shopping experience. If we can handle such implicit feedback information, our model will be applicable to a wider range of scenarios.

**REFERENCES**

[1] P. Covington, J. Adams, and E. Sargin, “Deep neural networks for YouTube recommendations,” in Proc. 10th ACM Conf. Recommender Syst., Sep. 2016, pp. 191–198.

[2] S. Rendle, “Factorization machines,” in Proc. IEEE Int. Conf. Data Mining, Dec. 2010, pp. 995–1000.

[3] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “DeepFM: A factorization-machine based neural network for CTR prediction,” in Proc. 26th Int. Joint Conf. Artif. Intell., Aug. 2017, pp. 1–8.

[4] H.-T. Cheng, “Wide & deep learning for recommender systems,” in Proc. 1st Workshop Deep Learn. Recommender Syst., Sep. 2016, pp. 7–10.

[5] R. Wang, “Deep & cross network for ad click predictions,” in Proc. ADKDD, 2017, pp. 1–17.

[6] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, “XDeepFM: Combining explicit and implicit feature interactions for recommender systems,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2018, pp. 1754–1763.

[7] Y. Qu, H. Cai, K. Ren, W. Zhang, Y. Yu, Y. Wen, and J. Wang, “Product-based neural networks for user response prediction,” in Proc. IEEE 16th Int. Conf. Data Mining (ICDM), Dec. 2016, pp. 1149–1154.

[8] W. Zhang, T. Du, and J. Wang, “Deep learning over multi-field categorical data,” in Proc. Eur. Conf. Inf. Retr. Cham, Switzerland: Springer, 2016.

[9] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, “Deep interest network for click-through rate prediction,” in Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2018, pp. 1059–1068.

[10] G. Zhou, N. Mou, Y. Fan, Q. Pi, W. Bian, C. Zhou, X. Zhu, and K. Gai, “Deep interest evolution network for click-through rate prediction,” in Proc. AAAI Conf. Artif. Intell., vol. 33, Jul. 2019, pp. 5941–5948.

**FIGURE 4.** Visualization of model prediction A.

**FIGURE 5.** Visualization of model prediction B.
[11] A. Vaswani, “Attention is all you need,” in Proc. 31st Int. Conf. Neural Inf. Process. Syst. Red Hook, NY, USA: Curran Associates, 2017.
[12] D. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in Proc. 3rd Int. Conf. Learn. Represent., 2015, pp. 1–15.
[13] T. Fawcett, “An introduction to ROC analysis,” Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.
[14] S. Zhai, K.-H. Chang, R. Zhang, and Z. M. Zhang, “DeepIntent: Learning attentions for Online advertising with recurrent neural networks,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD), 2016, pp. 1295–1304.
[15] C. Zhou, “ATRank: An attention-based user behavior modeling framework for recommendation,” in Proc. 32nd AAAI Conf. Artif. Intell., 2018.
[16] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang, “AutoInfl: Automatic feature interaction learning via self-attentive neural networks,” in Proc. 28th ACM Int. Conf. Inf. Knowl. Manage. (CIKM), 2019, pp. 1161–1170.
[17] D. Bahdanau, “Neural machine translation by jointly learning to align and translate,” in Proc. 3rd Int. Conf. Learn. Represent., 2015, pp. 1–15.
[18] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, “Attentional factorization machines: Learning the weight of feature interactions via attention networks,” in Proc. 26th Int. Joint Conf. Artif. Intell., Aug. 2017, pp. 1–7.
[19] F. M. Harper and J. A. Konstan, “The MovieLens datasets: History and context,” ACM Trans. Interact. Intell. Syst., vol. 5, no. 4, pp. 1–19, Dec. 2015.

YUCHEN LIU received the B.E. degree from the Beijing University of Posts and Telecommunications, in 2017, where he is currently pursuing the Grade 3 master’s degree with the School of Software Engineering. His research interests include recommender systems and data mining.

TAN YANG received the Ph.D. degree from the Beijing University of Posts and Telecommunications, in 2010. She is currently an Associate Professor with the Beijing University of Posts and Telecommunications. Her research interests include data mining, network performance evaluation, and the mobile Internet.

TAO QI received the Ph.D. degree from Xi’an Jiaotong University, in 1992. He is currently a Professor with the Beijing University of Posts and Telecommunications. His research interests include data mining, information security, and the mobile Internet.

* * *