

# MTBRN: Multiplex Target-Behavior Relation Enhanced Network for Click-Through Rate Prediction

Yufei Feng<sup>1\*</sup>, Fuyu Lv<sup>1\*</sup>, Binbin Hu<sup>2</sup>, Fei Sun<sup>1</sup>, Kun Kuang<sup>3</sup>, Yang Liu<sup>4</sup>, Qingwen Liu<sup>1</sup>, Wenwu Ou<sup>1</sup>

<sup>1</sup>Alibaba Group, Hangzhou, China

<sup>2</sup>Ant Financial Services Group, Hangzhou, China

<sup>3</sup>Zhejiang University, Hangzhou, China

<sup>4</sup>Indiana University Bloomington, Indiana, United States

fyf649435349@gmail.com, fuyu.lfy@alibaba-inc.com, bin.hbb@antfin.com, ofey.sf@alibaba-inc.com, kkun2010@gmail.com, yl82@indiana.edu, {xiangsheng.lqw, santong.oww}@alibaba-inc.com

## ABSTRACT

Click-through rate (CTR) prediction is a critical task for many industrial systems, such as display advertising and recommender systems. Recently, modeling user behavior sequences attracts much attention and shows great improvements in the CTR field. Existing works mainly exploit attention mechanism based on embedding product when considering relations between user behaviors and target item. However, this methodology lacks of concrete semantics and overlooks the underlying reasons driving a user to click on a target item. In this paper, we propose a new framework named **Multiplex Target-Behavior Relation enhanced Network (MTBRN)** to leverage multiplex relations between user behaviors and target item to enhance CTR prediction. Multiplex relations consist of meaningful semantics, which can bring a better understanding on users' interests from different perspectives. To explore and model multiplex relations, we propose to incorporate various graphs (e.g., knowledge graph and item-item similarity graph) to construct multiple relational paths between user behaviors and target item. Then Bi-LSTM is applied to encode each path in the path extractor layer. A path fusion network and a path activation network are devised to adaptively aggregate and finally learn the representation of all paths for CTR prediction. Extensive offline and online experiments clearly verify the effectiveness of our framework.

## KEYWORDS

Recommender System; Click-Through Rate Prediction

### ACM Reference Format:

Yufei Feng<sup>1\*</sup>, Fuyu Lv<sup>1\*</sup>, Binbin Hu<sup>2</sup>, Fei Sun<sup>1</sup>, Kun Kuang<sup>3</sup>, Yang Liu<sup>4</sup>, Qingwen Liu<sup>1</sup>, Wenwu Ou<sup>1</sup>. 2020. MTBRN: Multiplex Target-Behavior Relation Enhanced Network for Click-Through Rate Prediction. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20)*, October 19–23, 2020, Virtual Event, Ireland. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3340531.3412729>

\* denotes equal contributions.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

© 2020 Association for Computing Machinery.

ACM ISBN 978-1-4503-6859-9/20/10...\$15.00

<https://doi.org/10.1145/3340531.3412729>

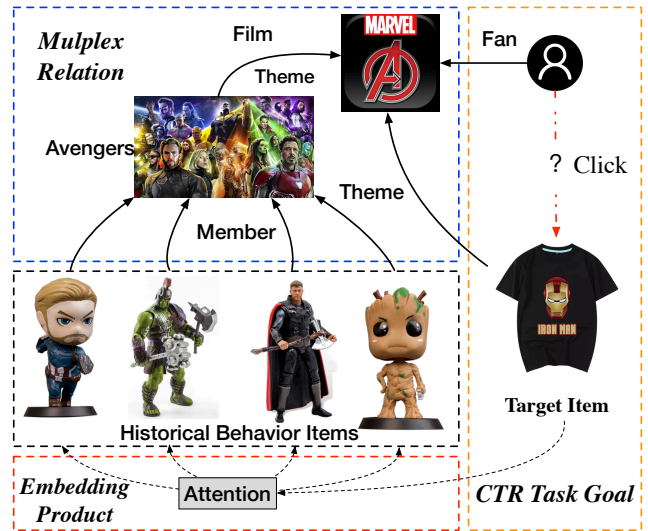


Figure 1: An illustration example of multiplex relations between a user's historical behaviors and the target item in CTR prediction task.

## 1 INTRODUCTION

Click-through rate (CTR) prediction lives the heart at display advertising and recommender systems. It estimates the probability of a user to click on a given target item. The quality of CTR is fundamental to user experience and user retention. Recently, modeling user behavior sequences is prevailing in CTR prediction. Several related algorithms [5, 15–17, 29, 30] have been proposed and achieved good performance in real-world applications. They represent behavior sequences as fixed-length vectors of user interests and feed them into the deep neural network for final CTR prediction. These models mainly exploit attention mechanism based on embedding product to aggregate user behaviors w.r.t target item.

Although these methods have achieved performance improvement to some extent, they still face a few major weaknesses. Most importantly, they lack of concrete semantics and overlook the underlying reasons driving a user to click on a target item. As a result, they fail to precisely comprehend a user's interest. As illustrated in Figure 1, a movie fan of *Avengers* is likely to click on an "Iron Man" graphic T-shirt, even if he has only browsed some spin-off products of *Avengers*. Such underlying reason, also named multiplex relation, is composed of multi-typed semantic relatedness (e.g., "fan of", "member of" and "theme of") and different types of items

or entities (e.g., "members of Avengers" and "film of Avengers"). It is difficult for existing models to capture the multiplex relation where these spin-off products and T-shirts can be explicitly linked to the same theme of *Avengers* by meaningful semantics. Furthermore, there are more than one multiplex relations between user behaviors and target item. These multiplex relations are particularly helpful to reveal the preferences (i.e., reasons) of users on consuming items from different perspectives. For instance, after watching the movie *Avengers: Infinity War*, a user may choose either *Avengers: End Game* or *Sherlock Holmes* to watch next. Because the former is the sequel of *Avengers: Infinity War*, and the latter is also starred by *Robert Downey Jr.* The two movies should be recommended to the user since both relations with *Avengers: Infinity War* must be considered. Therefore, without explicitly modeling multiplex relations, it is conceptually difficult for precise recommendation.

In order to address aforementioned problems, we propose a new framework named **Multiplex Target-Behavior Relation enhanced Network (MTBRN)** for CTR prediction. MTBRN transfers the information of multiplex relations between user behaviors and target item in a unified framework, so as to bridge and associate them. Knowledge graph (KG) emerges as an alternative to describe such relations, as the surge of interests in incorporating KG into recommender systems due to its comprehensive auxiliary data [2, 9, 10, 20?–28]. It introduces semantic relatedness among items and various entities, which can capture multiple underlying connections between items. That motivates us to model multiplex relations based on KG.

We explore and construct multiple paths between user behaviors and target item on KG to capture multiplex relations via graph search algorithm. In a modern web-scale recommender system, there are other graphs that can provide useful linking information to describe multiplex relations. An item-item similarity graph, for instance, can establish high-order connection between similar items. In the remaining part of this paper, we demonstrate our framework can incorporate these graphs in the same way as KG. To integrate those relational paths into MTBRN, we use Bi-LSTM to encode each path. Relational paths from various graphs can benefit and complement each other, so we devise a fusion network for their higher order feature interaction. After the fusion network, different path representations are adaptively aggregated into the final representation of multiplex relations through an attention based activation network. At last, the representation and other features are concatenated and fed into feature interacting layer for CTR prediction. Experiments were conducted on a proprietary industrial dataset and a public dataset, on which our framework displays state-of-the-art results. MTBRN has been fully deployed into product recommender of one popular E-commerce Mobile App and achieves significant CTR improvements by 7.9%.

The main contributions of this paper are summarized as follows:

- We highlight the importance of multiplex relations between user behaviors and target item in CTR prediction. We propose a path based method to leverage such relations on different graphs.
- A new CTR prediction framework named MTBRN is proposed to explore and model multiplex relations. Multiple relational paths are extracted from various graphs. Bi-LSTM

**Table 1: Notations.**

Notations	Description
$y_{uv}, \hat{y}_{uv}$	label and the predicted probability
$u, v$	the user and target item
$\mathcal{U}, \mathcal{I}$	the user set and item set
$x_u, x_v, \mathcal{B}$	user profile, item profile and user behaviors
$b_i$	the $i_{th}$ user behavior item
$\mathcal{G}_{cf}, \mathcal{G}_{kg}$	item-item similarity graph and knowledge graph
$\mathcal{P}_{kg}, \mathcal{P}_{cf}$	path sets extracted from $\mathcal{G}_{cf}$ and $\mathcal{G}_{kg}$
$\mathbf{v}_i$	embedding vector for the $i_{th}$ feature
$\mathbf{e}_i$	embedding vector for the $i_{th}$ node of the path
$\mathbf{h}_p$	the concatenated output of Bi-LSTM
$\mathbf{H}_{cf}, \mathbf{H}_{kg}$	the output of $\mathcal{P}_{cf}, \mathcal{P}_{kg}$ in the relational path extractor layer
$\mathbf{H}_{fu}$	the output of $\mathbf{H}_{cf}, \mathbf{H}_{kg}$ in the relational path fusion layer
$\mathbf{A}_{cf}, \mathbf{A}_{kg}, \mathbf{A}_{fu}$	the output of $\mathbf{H}_{cf}, \mathbf{H}_{kg}$ and $\mathbf{H}_{fu}$ in the relational path activation layer

and a path fusion/activation network are employed to adaptively learn the final representation of multiplex relations.

- We performed extensive experiments on a proprietary industrial dataset and a public dataset. Experimental results verify the rationality of each graph and the effectiveness of the proposed MTBRN framework.

## 2 PROBLEM FORMULATION

In this section, we formulate the problem of click-through rate prediction with multiplex relations between user behaviors and target item. Specifically, we construct the following two graphs to extract multiplex relations, which are illustrated in the left part of Figure 2.

**Item-item similarity graph.** The item-item similarity graph is denoted as  $\mathcal{G}_{cf} = \{(i, sim(i, j), j) | i, j \in \mathcal{I}\}$ , where  $\mathcal{I}$  is the set of items and  $sim(i, j)$  describes the similarity between item  $i$  and  $j$ . We calculate  $sim(i, j)$  via the idea of item-based collaborative filtering [19], which can be formulated as follows,

$$sim(i, j) = \frac{\mathbf{Y}[:, i] \odot \mathbf{Y}[:, j]}{\sqrt{\|\mathbf{Y}[:, i]\|_2 \|\mathbf{Y}[:, j]\|_2}}. \quad (1)$$

Here,  $\odot$  refers to element-wise product,  $\mathbf{Y} \in \mathbb{R}^{m \times n}$  is the user-item interaction matrix from users' historical behaviors, where  $m$  and  $n$  is the number of users and items respectively and  $y_{ui} = 1$  indicates that user  $u$  interacted with item  $i$ , 0 otherwise. For example, the triple  $(i, 0.3, j)$  indicates that the similarity of item  $i$  and item  $j$  is 0.3. The triple  $(j, 0.9, k)$  indicates that the similarity between item  $j$  and item  $k$  is 0.9. The item  $j$  is the junction of the two triples.

**Knowledge graph.** The knowledge graph describes semantic correlation among items and real-world entities via various relations, which can be denoted as  $\mathcal{G}_{kg} = \{(h, r, t) | h, t \in \mathcal{E}, r \in \mathcal{R}\}$ , where  $\mathcal{E}$  is the set of items and entities, and  $\mathcal{R}$  is the set of relations. For example, the triple  $(i, category, longuette)$  indicates that the item  $i$  belongs to the *longuette* category. The triple  $(longuette, parent, clothing)$  indicates the parent of *longuette* is *clothing*. The *longuette* entity is the junction of the two triples.

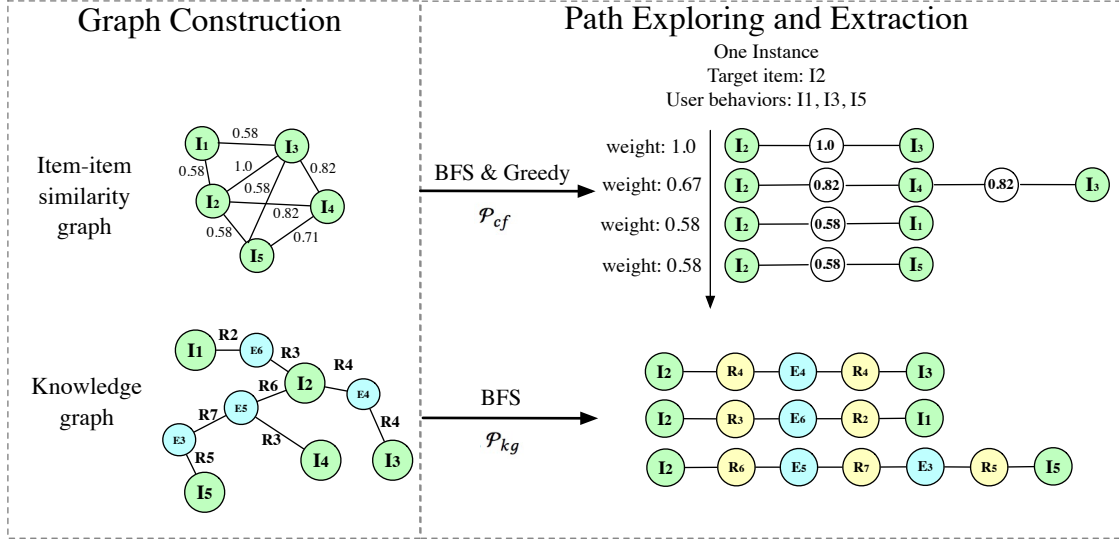


Figure 2: The graph construction methods and path exploration and extraction strategy.

**Multiplex relations enhanced CTR prediction.** Now, we formulate the CTR prediction problem to be addressed in this paper. We assume a set of historical click records between users and items, denoted as  $\mathcal{Y}$ .  $\mathcal{Y}$  is comprised of  $\{u, v, \mathcal{B}, y_{uv} | u \in \mathcal{U}, v \in \mathcal{I}, y_{uv} \in \{0, 1\}\}$ , where  $\mathcal{U}$  and  $\mathcal{I}$  respectively represent the user and item sets,  $\mathcal{B}$  is the set of user behaviors consisting of item ids that the user has recently clicked on and  $y_{uv}$  is set to one if and only if user  $u$  has clicked on item  $v$ . Moreover, each user  $u$  is associated with a user profile  $x_u$  consisting of sparse features (e.g., user id and gender) and numerical features (e.g., user age), while each target item  $v$  is also associated with a item profile  $x_v$  consisting of sparse features (e.g., item id and brand) and numerical features (e.g., price). In order to effectively explore and exploit the multiplex relations between user behaviors and target item, we elaborately construct knowledge graph  $\mathcal{G}_{kg}$  and item-item similarity graph  $\mathcal{G}_{cf}$  to enhance CTR prediction. Formally, our goal is to learn a prediction function  $\hat{y}_{uv} = \mathcal{F}(x_u, x_v, \mathcal{B}, \mathcal{G}_{kg}, \mathcal{G}_{cf}; \Theta)$ , such that  $\hat{y}_{uv}$  represents the predicted probability of user  $u$  to click on target item  $v$  and  $\Theta$  represents the parameters of the prediction function  $\mathcal{F}$ . The notations are summarized in Table 1.

### 3 MTBRN FRAMEWORK

In this section, we introduce our proposed framework MTBRN. We first propose the path-based algorithm that captures and models multiplex relations from different graphs. Afterwards, we elaborate on the deep neural network architecture of MTBRN, which is proposed to encode the relational information of the paths extracted from auxiliary graphs and adaptively learn how paths contribute to the final prediction.

#### 3.1 Path Exploration and Extraction Strategy

In this part, we introduce the strategy to effectively explore and extract paths between user behaviors and target item, which is a natural way to describe multiplex relations on graphs. Previous

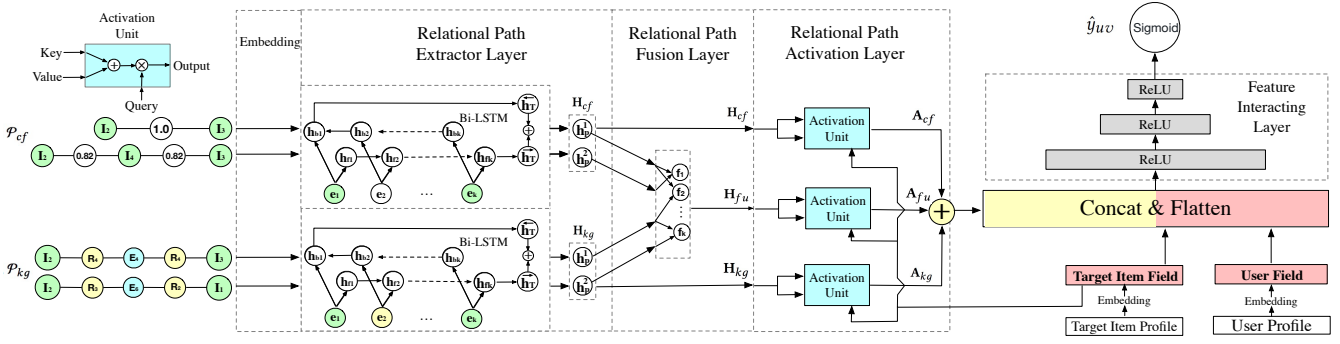
path-based models either use the random walk strategy [25] or design an auxiliary task (e.g., matrix factorization) to assign weights to paths [9]. Unfortunately, random walk based methods may harm the stability of model performance while auxiliary task based methods only work under specific settings. To effectively capture multiplex relations between user behavior  $b_i \in \mathcal{B}$  and target item  $v$ , we adopt different search strategies to extract paths on different graphs, listed as follows:

- For the item-item similarity graph  $\mathcal{G}_{cf}$ , we exhaustively search any potential path following breadth-first search (BFS) and greedy-selection principle according to the similarity score. We only keep top  $k_{\mathcal{G}_{cf}}$  paths with the shortest length. One demo path extracted from the item-item similarity graph is defined as:  $b_i \rightarrow \text{sim}(b_i, A) \rightarrow A \rightarrow \dots \rightarrow \text{sim}(\cdot, v) \rightarrow v$ , where  $(b_i, \text{sim}(b_i, A), A)$  is one triple in  $\mathcal{G}_{cf}$ .
- For the knowledge graph  $\mathcal{G}_{kg}$ , we follow the BFS principle to generate all paths over the linked relations and entities between user behaviors and target item, and keep top  $k_{\mathcal{G}_{kg}}$  paths with the shortest length. One demo path extracted from the knowledge graph can be defined as:  $b_i \rightarrow r_1 \rightarrow e_1 \rightarrow \dots \rightarrow v$ , where  $(b_i, r_1, e_1)$  is one triple in  $\mathcal{G}_{kg}$ .

We illustrate the procedure in the Figure 2. Afterwards, we can get two types of path sets:  $\mathcal{P}_{cf}$  and  $\mathcal{P}_{kg}$ , which capture multiplex relations between user behaviors and target item from different perspectives.

#### 3.2 Model Architecture

As shown in Figure 3, MTBRN consists of two parts before feature interacting layer. The right part is the embedding vectors transformed from the user profile feature and target item profile feature. The left part models the extracted relational paths, which is composed of three layers from left to right: (1) *relational path extractor layer* extracts bi-directional relational information on paths; (2) *relational path fusion layer* captures the higher-order interactions of paths; (3) *relational path activation layer* adaptively learns the



**Figure 3: Overview of proposed framework MTBRN. The right part is embedding vectors of user and target item profile features. The left part is our main contribution, which processes the extracted relational paths. We use Bi-LSTM, path fusion and activation network to encode multiplex relational paths. Representations from the two parts are concatenated, flattened and then fed into feature interacting layer for the final prediction.**

representation of the relational paths w.r.t. the target item. Finally the outputs of the two parts are concatenated, flattened and fed into the feature interacting layer for the final prediction. Next, we will present detailed illustration of the proposed MTBRN model.

**Embedding.** Embedding is a popular technique that projects each feature to a dense vector representation. Formally, let  $\mathbf{v}_i \in \mathbb{R}^d$  be the embedding vector for the  $i_{th}$  feature. Since there exist numerical features (e.g., price) in original features, we rescale the embedding vector by its input feature value to account for the real valued features. In this way, the embedding of the  $i_{th}$  feature for the input vector  $\mathbf{x}$  is calculated as  $\mathbf{v}_i x_i$ . Due to the sparsity of the input feature, we only need to preserve the embeddings for non-zero features. Thus, the final embedding of input feature vector  $\mathbf{x}$  is obtain as:

$$\mathbf{e} = \mathbf{v}_1 \mathbf{x}_1 \oplus \mathbf{v}_2 \mathbf{x}_2 \oplus \dots \oplus \mathbf{v}_i \mathbf{x}_i \dots, \quad \mathbf{x}_i \neq 0, \quad (2)$$

where  $\oplus$  is the concatenation operation. Following the above embedding procedure, the user and target item profile feature space can be represented by  $\mathbf{x}_u$  and  $\mathbf{x}_v$ , respectively. One path in  $\mathcal{P}_{cf}$  and  $\mathcal{P}_{kg}$  can be uniformly represented by  $[b_i; node_1; \dots; node_j; \dots; v]$ , where  $b_i$  is the  $i$ -th user behavior item,  $v$  is the target item, and  $node_j$  can be either sparse features (i.e., items, relations and entities), or numerical features (i.e., similarity score). With embedding, the path can be generally represented by  $[\mathbf{e}_1; \dots; \mathbf{e}_k]$ , where  $k$  is the length of the path.

**Relational Path Extractor Layer.** The path sets describe the multiplex relations of target item and user behaviors from different perspectives, and this layer is designed to extract the information passing along the paths. Previous path-based models either use CNN [9] or LSTM [25] to encode the relational paths from user to target item. Whereas items in the user behavior and target item are in the same semantic space and the transmission of information along the path is always asymmetric. Therefore, we naturally apply Bi-LSTM [6] to extract two-way information transmitted on the path asymmetrically. Mathematically, LSTM [8] network is

implemented as follows:

$$\begin{aligned} \mathbf{i}_t &= \sigma(\mathbf{W}_{xi} \mathbf{e}_t + \mathbf{W}_{hi} \mathbf{h}_{t-1} + \mathbf{W}_{ci} \mathbf{c}_{t-1} + \mathbf{b}_i) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_{xf} \mathbf{e}_t + \mathbf{W}_{hf} \mathbf{h}_{t-1} + \mathbf{W}_{cf} \mathbf{c}_{t-1} + \mathbf{b}_f) \\ \mathbf{c}_t &= \mathbf{f}_t \mathbf{c}_{t-1} + \mathbf{i}_t \tanh(\mathbf{W}_{xc} \mathbf{e}_t + \mathbf{W}_{hc} \mathbf{h}_{t-1} + \mathbf{b}_c) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_{xo} \mathbf{e}_t + \mathbf{W}_{ho} \mathbf{h}_{t-1} + \mathbf{W}_{co} \mathbf{c}_{t-1} + \mathbf{b}_o) \\ \mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t) \end{aligned} \quad (3)$$

where  $\sigma(\cdot)$  is the logistic function,  $\mathbf{i}$ ,  $\mathbf{f}$ ,  $\mathbf{o}$  and  $\mathbf{c}$  are the input gate, forget gate, output gate and cell vectors, respectively. Forward and backward LSTMs model the bi-direction information, that is, the last representation of each path  $\mathbf{h}_p$  is calculated as follows:

$$\mathbf{h}_p = \overrightarrow{\mathbf{h}}_T \oplus \overleftarrow{\mathbf{h}}_T \quad (4)$$

where  $\overrightarrow{\mathbf{h}}_T$  and  $\overleftarrow{\mathbf{h}}_T$  represent the last hidden state of the forward LSTM and backward LSTM, respectively. Note that the parameters of Bi-LSTM are shared when encoding paths from the same set. After the relational path extractor layer,  $\mathcal{P}_{cf}$  and  $\mathcal{P}_{kg}$  are represented by  $\mathbf{H}_{cf}$  and  $\mathbf{H}_{kg}$ , respectively. For example,  $\mathbf{H}_{kg} = [\mathbf{h}_p^1; \dots; \mathbf{h}_p^i; \dots]$  denotes last representations of all paths in  $\mathcal{P}_{kg}$ .

**Relational Path Fusion Layer.** Relational paths can benefit and complement each other. The previous two paths  $b_1 - 0.6 - v$  and  $b_1 - category - longuette - category - v$ , for example, may bring much more information to light if considered together. Inspired by the improvements [3, 7, 11, 18] of feature interaction in the CTR field, we further capture the higher order interaction among representations of paths. Mathematically, the interactive representation  $\mathbf{H}_{fu}$  can be calculated as follows:

$$\mathbf{H}_{fu} = \{\mathbf{h}_p^i \cdot \mathbf{h}_p^j \mid 1 \leq i \leq k, i + 1 \leq j \leq k\} \quad (5)$$

where  $\mathbf{h}_p^i \in \mathbf{H}_{cf} \cup \mathbf{H}_{kg}$ ,  $\cdot$  is element-wise multiply and  $k = k_{\mathcal{G}_{cf}} + k_{\mathcal{G}_{kg}}$  is the number of all paths.

**Relational Path Activation Layer.** Intuitively, relational paths contribute unequally to target item and further influence the final prediction. Taking two paths  $b_1 - 0.6 - v$  and  $b_2 - 0.3 - v$  in  $\mathcal{P}_{cf}$  as an example,  $v$  is more similar to target item  $b_1$  than  $b_2$  since the first path has the higher similarity score than the second path. Meanwhile, relational paths extracted from different graphs are not on the same scale. Taking another two paths  $b_1 - 0.6 - v$  and  $b_1 - category - longuette - category - v$  for example, it is hard to distinguish which one is more effective. Therefore, the weights of

path representations in  $\mathbf{H}_{cf}$ ,  $\mathbf{H}_{kg}$ ,  $\mathbf{H}_{fu}$  need to be reassigned w.r.t. the target item. For this reason, the attention mechanism [1] is applied to conduct alignment between paths and the target item. Mathematically, the adaptive representation of relational paths in each path set w.r.t. the target item is calculated as follows:

$$a^i = \frac{\exp(\mathbf{h}_p^i \mathbf{W} \mathbf{x}_v)}{\sum_{j=1}^n \exp(\mathbf{h}_p^j \mathbf{W} \mathbf{x}_v)} \quad (6)$$

$$\mathbf{A} = \sum_{i=1}^n a^i \mathbf{h}_p^i$$

where  $n$  is the number of paths in each path set and  $\mathbf{W}$  is the trainable parameters. After the relational path activation layer,  $\mathbf{H}_{cf}$ ,  $\mathbf{H}_{kg}$  and  $\mathbf{H}_{fu}$  respectively are encoded into vectors  $\mathbf{A}_{cf}$ ,  $\mathbf{A}_{kg}$  and  $\mathbf{A}_{fu}$ .

**Feature Interacting Layer.** Following previous studies [5, 16, 29, 30] in the CTR prediction field, Multiple Layer Perceptron (MLP) is applied for better feature interaction. Here we calculate the final output as follows:

$$\hat{y}_{uv} = \sigma(f(f(f(\mathbf{x}_u \oplus \mathbf{x}_v \oplus \mathbf{A}_{cf} \oplus \mathbf{A}_{kg} \oplus \mathbf{A}_{fu})))) \quad (7)$$

where  $f(\mathbf{x}) = \text{ReLU}(\mathbf{W}\mathbf{x} + \mathbf{b})$ ,  $\sigma(\cdot)$  is the logistic function and  $\hat{y}_{uv}$  represents the prediction probability of the user  $u$  to click on the target item  $v$ .

**Loss Function.** We reduce the CTR prediction task to a binary classification problem with binary cross-entropy loss function, which can be defined as follows:

$$\text{Loss} = -\frac{1}{N} \sum_{(u,v) \in \mathbb{D}} (y_{uv} \log \hat{y}_{uv} + (1 - y_{uv}) \log(1 - \hat{y}_{uv})) \quad (8)$$

where  $\mathbb{D}$  is the training dataset and  $y_{uv} \in \{0, 1\}$  represents whether the user  $u$  clicked on the target item  $v$ .

## 4 EXPERIMENTS

We evaluate the proposed framework MTBRN on a proprietary industrial E-commerce dataset and the public Yelp dataset. Moreover, we conduct strict online A/B testing to evaluate the performance of MTBRN after deployed to real-world settings. Specifically, we will make comprehensive analyses about MTBRN, with the aim of answering the following questions:

- **RQ1** How does MTBRN perform compared with other state-of-the-art (SOTA) user behavior enhanced CTR models?
- **RQ2** How does MTBRN perform compared with competitors that can leverage the same graph in our framework for recommendation?
- **RQ3** How do the multiplex relations between user behaviors and target item derived from different graphs benefit CTR prediction?

### 4.1 Datasets and Graph Description

We report detailed description of the two datasets and all graphs utilized by MTBRN in Table 2.

**4.1.1 E-commerce Dataset.** It is an industrial real-world recommender dataset collected from a popular E-commerce Mobile App. The dataset consists of impression/click logs in 8 consecutive days, where clicked ones are treated as positive instances and negative otherwise. Logs from 2019-08-22 to 2019-08-28 are used for training,

**Table 2: Statistics of the dataset and graphs.**

Description	E-commerce	Yelp
Users	0.2 billion	45.4 thousand
Items	0.1 billion	45.1 thousand
Records	7.2 billion	1.0 million
User behaviors	10	10
Triplets in $\mathcal{G}_{cf}$	26.2 billion	8.1 million
Relations in $\mathcal{G}_{kg}$	34	35
Entities in $\mathcal{G}_{kg}$	14.4 million	83.3 thousand
Triplets in $\mathcal{G}_{kg}$	33.8 billion	1.6 million

and logs from 2019-08-29 are for testing. Moreover, E-commerce dataset contains user profile (e.g., id, age, and gender), item profile (e.g., id, category, and price) and real-time user behaviors<sup>1</sup>.

**4.1.2 Yelp Dataset.** Yelp datasets records interactions between users and local business and contains user profile (e.g., id, review count, and fans), item profile (e.g., id, city, and stars) and real-time user behaviors<sup>2</sup>. To adapt for the CTR prediction task, we treat all observed interactions as positive instance. For each user-item pair in positive instance, we randomly sample 5 negative samples that have no interaction record with the specific user to constitute negative instance set. Then, in chronological order, we take each user’s last 30 instances for testing and last 31 to 120 instances for training.

**4.1.3 Item-item Similarity Graph.** We construct the item-item similarity graph as introduced in section 2. To model the real-world recommender system and prevent information leakage, we only construct the graph based on user behaviors that do not exist in the training and testing datasets. Moreover, considering the tremendous number of items, we only keep top 5 neighbors with highest similarity score for each item and the max depth of each node is set to 3.

**4.1.4 Knowledge Graph.** Knowledge-aware recommendation relies highly on the quality of the knowledge graph. We construct the knowledge graph following procedures described in [13]. For the E-commerce dataset, relations include category, parent, season, style, etc. For the Yelp dataset, relations include category, location, attribute, etc.

## 4.2 Experimental Setup

**4.2.1 Competitors.** We consider two kinds of representative CTR prediction methods: user behavior sequence enhanced methods (i.e., YoutubeNet, DIN, DIEN and DSIN), item-item similarity graph based method (i.e., GIN) and knowledge graph based methods (i.e., RippleNet, KPRN and KGAT). To examine the effect of the multiplex relations derived from different graphs and relational path fusion layer, we prepare three variants of MTBRN (i.e., MTBRN<sub>cf</sub>, MTBRN<sub>kg</sub> and MTBRN<sub>r</sub>). The competitors are given below:

- **YoutubeNet** [4] is designed for video recommendation in Youtube. It treats user behaviors equally and applies average pooling operation.

<sup>1</sup>Real-time user behaviors refer to user behaviors before this instance occurs.

<sup>2</sup><https://www.yelp.com/dataset>.

- **DeepFM** [7] is technically designed to capture the multi-order interactions of features. It combines FM and deep model, without the need of complicated feature engineering.
- **DIN** [30] uses the embedding product attention mechanism to learn the adaptive representation of user behaviors w.r.t. the target item.
- **DIEN** [29] designs an auxiliary network to capture user’s temporal interests and proposes AUGRU to model the interest evolution.
- **DSIN** [5] divides the user behavior sequence into multiple sessions and designs the extractor layer and evolving layer to extract the session interests and model how they evolves during time.
- **GIN** [12] is the first to mine and aggregate the user’s latent intention on the co-occurrence item graph with graph attention technique. GIN can be easily applied on item-item similarity graph.
- **RippleNet** [20] explores the multiple *ripples* of user behaviors on the knowledge graph and propagates the representation of the target item recursively layer by layer.
- **KPRN** [25] applies LSTM to directly model the multiple user-item paths via the knowledge graph and then aggregate them for the final prediction.
- **KGAT** [24] recursively propagates the high-order connectivity of the user and item via the knowledge graph and user-item bipartite graph with graph attention technique.
- **MTBRN<sub>cf</sub>** : MTBRN with only item-item similarity graph.
- **MTBRN<sub>kg</sub>** : MTBRN with only knowledge graph.
- **MTBRN<sub>r</sub>** : MTBRN without relational path fusion layer.

**4.2.2 Evaluation Metrics.** In our experiments, we evaluate the performance of different methods for comparison via AUC (Area Under ROC Curve) and Logloss (cross entropy), which are widely adopted in the CTR field. The larger AUC, the better performance. Base on our practice lessons, 0.1% increase of offline AUC on our proprietary dataset is corresponding to relative 1% online CTR lift.

**4.2.3 Implementation.** We implemented all the models in Tensorflow 1.4. We tailored models which were not originally designed for the CTR prediction task, including concatenation with other features and addition of MLP layers at last. We did not apply pre-training, batch normalization and regularization techniques. Instead, random uniform initializer is employed. With the computational cost in mind, only 2 layers of neighbours are reserved for each user behavior item for GIN. For KGAT, the neighbour depth of the user and item is set to 5 and 4, respectively. For RippleNet, the depth of ripple is set to 3. The extracted paths on the item-item similarity graph and knowledge graph are reserved up to 50. All models are tuned using Adagrad optimizer with learning rate 0.001 and batch size 300. Embedding size of each feature is set to 4. The hidden units in MLP layers are set 512, 256, and 128, respectively. We ran each model three times and computed the mean to eradicate any discrepancies.

### 4.3 Performance Comparison (RQ1&RQ2)

In this section, we start off comparing the performance of MTBRN with SOTA user behavior enhanced CTR models, as well as with

**Table 3: Model performance (AUC and Logloss) with each separate graph and all graphs.**

Graph	Model	E-commerce		Yelp	
		AUC	Logloss	AUC	Logloss
-	YoutubeNet	0.6017	0.6279	0.7109	0.4899
-	DeepFM	0.6037	0.6192	0.7334	0.4882
-	DIN	0.6058	0.5735	0.7520	0.4579
-	DIEN	0.6065	0.5643	0.7581	0.4518
-	DSIN	0.6073	0.5394	0.7774	0.4392
$\mathcal{G}_{cf}$	GIN	0.6073	0.5416	0.7604	0.4471
	MTBRN <sub>cf</sub> <sup>†</sup>	0.6094	0.5329	0.7915	0.4129
	MTBRN <sub>cf</sub> <sup>‡</sup>	<b>0.6103</b>	<b>0.5244</b>	<b>0.7936</b>	<b>0.4075</b>
$\mathcal{G}_{kg}$	RippleNet	0.5975	0.6369	0.7324	0.4844
	KGAT	0.6062	0.5624	0.7876	0.4214
	KPRN	0.6091	0.5292	0.8267	0.3897
	MTBRN <sub>kg</sub>	<b>0.6209</b>	<b>0.4628</b>	<b>0.9088</b>	<b>0.3486</b>
ALL	MTBRN <sub>r</sub>	0.6235	0.4509	0.9231	0.3213
	MTBRN	<b>0.6246</b>	<b>0.4482</b>	<b>0.9408</b>	<b>0.3058</b>

† (‡) Paths shorter than 5 (7) are reserved, exploring up to 2 (3) layers of neighborhood.

other competitors leveraging each auxiliary graph. We report the performance of all models on the two datasets<sup>3</sup> in Table 3.

**4.3.1 User behavior Enhanced CTR models.** As shown in Table 3, DIN improves AUC obviously by leveraging the attention mechanism to activate the user’s relevant interests w.r.t. the target item. DIEN achieves better performance with the technically designed auxiliary net and AUGRU to model the interest evolution. DSIN performs better than DIEN by extracting user’s session interests. Nevertheless, most competitors in the CTR field reallocate weights to user behaviors only based on the item embedding with the attention mechanism. It can hardly figure out the complex reasons driving the user to click the target item. Overall, MTBRN significantly outperforms above state-of-the-art competitors on both datasets, which mainly benefits from two aspects: (1) the extracted multiplex relational paths from different graphs are more reasonable and concrete, so as to provide powerful clues why the user will click on the target item; (2) the technical design of MTBRN helps capture the multiplex relations of user behaviors and the target item. Both contribute much to the final prediction and help achieve the best performance. To answer **RQ2**, we gave extensive insights on how each graph and different components of MTBRN contribute to the best performance. More complicated models are used as competitors.

**4.3.2 Item-item Similarity Graph.** As shown in Table 3, GIN outperforms DIN, mainly benefiting from the exploration of users’ latent intention in graphs. However, GIN still ignores the relation between user behaviors and target item as well as the linked similarity score between items. In MTBRN<sub>cf</sub><sup>†</sup>, we explore and model the paths between user behaviors and the target item within the range of 2-layer adjacent neighborhood as the same as GIN. MTBRN<sub>cf</sub><sup>†</sup>

<sup>3</sup>Note that the relative improvements on the public Yelp dataset are much higher than those on the industrial E-commerce dataset, because the negative samples of the public Yelp dataset are generated by random sampling, which means easier to distinguish.

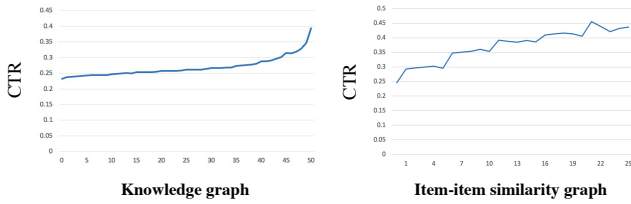


Figure 4: Impact of the number of relational paths.

outperforms GIN in both datasets. It empirically demonstrates the usefulness of the relation paths between user behaviors and target items for CTR prediction. Furthermore, we flexibly extend the neighbour depth of the graph to 3 (*i.e.*,  $MTBRN_{cf}^{\ddagger}$ ) for exploiting high-order information on the graph and, not surprisingly, more improvement is observed on  $MTBRN_{cf}^{\ddagger}$ . It conveys the message that longer paths can capture higher-order similarities of items and benefit the final prediction in the long run.

**4.3.3 Knowledge Graph.** We present the AUC performance of various knowledge-aware models for the CTR prediction task on both datasets in Table 3. KGAT and KPRN both outperform DIN, and specifically KPRN offers more increase on the both datasets. In contrast, RippleNet renders inferior performance than DIN. One reasonable explanation is that modeling relations explicitly between user behaviors and target item is more efficacious than user preference propagation in the KG. KGAT incorporates knowledge graph and user-item bipartite graph, hence it yields more improvements compared with DIN. However, KGAT has been found empirically to involve useless information for the final prediction and fails to capture direct interaction between user behaviors and target item. KPRN benefits from reasonable and explainable user-target paths derived from knowledge graph and outperforms KGAT.  $MTBRN_{kg}$  devotes to capturing the reasonable and explainable knowledge relations of user behaviors and target item. Moreover, the relational path extractor layer and activation layer help obtain the representation of multiplex relational paths and activate those related to the target item. Therefore,  $MTBRN_{kg}$  outperforms other competitors with the same knowledge graph.

**4.3.4 Effect of Relation Path Fusion Layer.** We conduct extensive experiments to verify the effectiveness of the proposed relational path fusion layer. We report the detailed comparison of model performance of MTBRN with or without paths fusion (*i.e.*,  $MTBRN_r$  and MTBRN) in Table 3. Not surprisingly,  $MTBRN_r$  outperforms MTBRN with single graph data (*i.e.*,  $MTBRN_{cf}$  and  $MTBRN_{kg}$ ) since it incorporates multiplex relations derived from different graphs for the final prediction. Moreover, MTBRN performs better than  $MTBRN_r$ , which demonstrates the effectiveness of the proposed relational path fusion layer.

#### 4.4 Validity Analysis of Paths (RQ3)

In this section, we make comprehensive instance-level analyses of the effectiveness of multiplex relations (*i.e.*, the extracted relational paths) derived from different graphs on the E-commerce dataset. As shown in Figure 4, CTR is calculated by averaging real label values (0 for non-click or 1 for click) of instances with the same number of paths in dataset. The number of relational paths (both

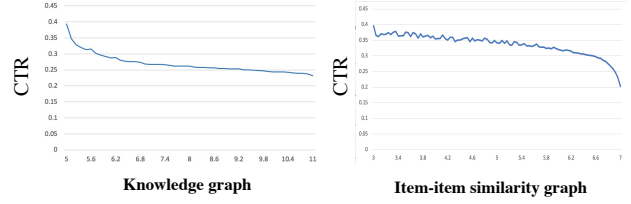


Figure 5: Impact of the average length of relational paths.

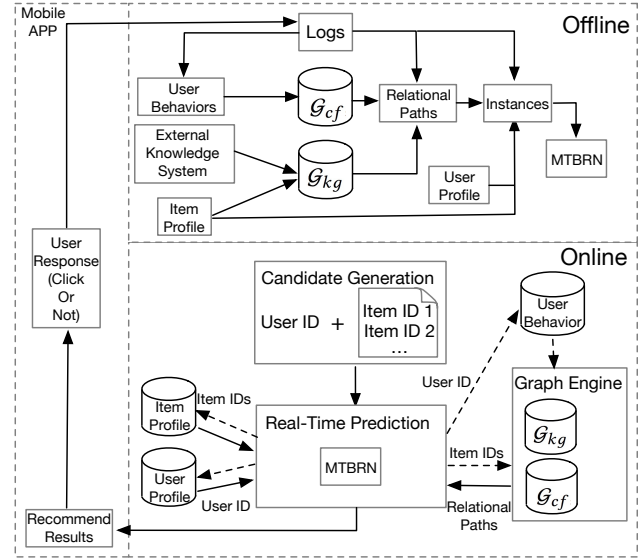


Figure 6: MTBRN deployment for Online CTR.

on knowledge graph and item-item similarity graph) is positively correlated with CTR. This indicates that the more relations between the target item and user behaviors, the more likely the user is to click on the target item. We also investigate the influence of average path length extracted from each sample on CTR. Figure 5 shows that, shorter relational paths from the item-item similarity graph and the knowledge graph contributes to higher CTR. It demonstrates that the shorter distance between user behaviors and the target item in a graph implies closer relation, which makes the user more likely to click on the target item.

#### 4.5 Online A/B Testing

We have deployed MTBRN into product recommender of one popular E-commerce Mobile App for several months. The pipelines of deployment are clearly displayed in Figure 6, which consists of three parts: *user response*, *offline training* and *online serving*. User-item interaction logs from *user response*, as well as knowledge graph and item-item similarity graph are fed into MTBRN model for training. When a user accesses the App, a series of candidate items are generated by MTBRN in real time. Subsequently, candidate items are sorted and truncated by the predicted scores, and recommended to the user. We conducted strict online A/B testing to validate the performance of MTBRN. Our online baseline is the latest deployed DIN. Online CTR increased by 7.9% on average compared to DIN, at the cost of 7 milliseconds more for MTBRN online inference. The

substantial increase in commercial revenue together with tolerable latency in serving serve as proof for the effectiveness of our proposed MTBRN.

## 5 RELATED WORK

### 5.1 User Behaviors Enhanced CTR

Click-through rate (CTR) prediction is an essential task in most industrial recommender systems. Recently, modeling user behavior sequences has attracted much attention and been widely proven effective in the CTR field. DIN [30] uses the attention mechanism with embedding product to learn the adaptive representation of the user behavior sequence w.r.t. the target item. Inspired by DIN, the majority of following up works inherit this kind of paradigm. DIEN [29] and SDM [14] devote to capturing users' temporal interests and modeling their sequential relations. DSIN [5] focuses on capturing the relationships of users' inter-session and intra-session behaviors. Though with great improvements, the embedding product based attention mechanism fails to capture the multiplex relations between user behaviors and target item.

### 5.2 Knowledge-aware Recommendation

Many recent research studies integrate knowledge graph that integrates more side information of items into recommendation to improve the interpretability of recommendation. RippleNet [20] combines the advantages of the previously mentioned two types of methods. KPRN [25] designs multiple paths between the user and item pair and uses LSTM to extract the information of each path. Wang et al. [22] uses graph convolution network to automatically discover both high-order structure information and semantic information of the knowledge graph. KGAT [24] leverages graph attention network to model high-order relation connectivity in the knowledge graph and user-item bipartite graph. Empirically, path-based methods make use of KG in a more natural and efficient way, that are more suitable for depicting the deep relevance between user behaviors and target item in our work for CTR prediction.

## 6 CONCLUSION

In this paper, we propose a new framework MTBRN to transfer the information of multiplex relations between user behaviors and target item in CTR prediction. The integration of different graphs and various connection paths ensure the superior performance of MTBRN over related work. We conducted extensive experiments on an industrial-scale dataset and a public dataset to demonstrate the effectiveness of our method. Empirical analyses show the validity of each component in the proposed framework.

## REFERENCES

- [1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
- [2] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *WWW*. ACM, 151–161.
- [3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishu 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*. ACM, 7–10.
- [4] Covington, Paul, Adams, Jay, Sargin, and Emre. 2016. Deep neural networks for youtube recommendations. In *RecSys*. ACM, 191–198.
- [5] Yufe Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. 2019. Deep Session Interest Network for Click-Through Rate Prediction. In *IJCAI*.
- [6] Alex Graves and Jrgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5-6 (2005), 602–610.
- [7] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).
- [8] Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
- [9] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *SIGKDD*. ACM, 1531–1540.
- [10] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y Chang. 2018. Improving sequential recommendation with knowledge-enhanced memory networks. In *SIGIR*. ACM, 505–514.
- [11] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware factorization machines for CTR prediction. In *RecSys*. ACM, 43–50.
- [12] Feng Li, Zhenrui Chen, Pengjie Wang, Yi Ren, Di Zhang, and Xiaoyu Zhu. 2019. Graph Intention Network for Click-through Rate Prediction in Sponsored Search. In *SIGIR*.
- [13] Xusheng Luo, Luxin Liu, Yonghua Yang, Le Bo, Yuanpeng Cao, Jinhang Wu, Qiang Li, Keping Yang, and Kenny Q. Zhu. 2020. AliCoCo: Alibaba E-commerce Cognitive Concept Net. In *SIGMOD*.
- [14] Fuyu Lv, Taiwei Jin, Changlong Yu, Fei Sun, Quan Lin, Keping Yang, and Wilfred Ng. 2019. SDM: Sequential deep matching model for online large-scale recommender system. (2019), 2635–2643.
- [15] Yabo Ni, Dan Ou, Shichen Liu, Xiang Li, Wenwu Ou, Anxiang Zeng, and Luo Si. 2018. Perceive your users in depth: Learning universal user representations from multiple e-commerce tasks. In *SIGKDD*. ACM, 596–605.
- [16] Qi Pi, Weijie Bian, Guorui Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Practice on Long Sequential User Behavior Modeling for Click-Through Rate Prediction. In *SIGKDD*.
- [17] Kan Ren, Jiarui Qin, Yuchen Fang, Weinan Zhang, Lei Zheng, Weijie Bian, Guorui Zhou, Jian Xu, Yong Yu, Xiaoqiang Zhu, and Kun Gai. 2019. Lifelong Sequential Modeling with Personalized Memorization for User Response Prediction. In *SIGIR*. ACM.
- [18] Steffen Rendle. 2010. Factorization machines. In *ICDM*. IEEE, 995–1000.
- [19] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.
- [20] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *CIKM*. ACM, 417–426.
- [21] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *WWW*. 1835–1844.
- [22] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. [n. d.]. Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. In *SIGKDD*. 968–977.
- [23] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Multi-Task Feature Learning for Knowledge Graph Enhanced Recommendation. In *WWW*. ACM, 2000–2010.
- [24] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *SIGKDD*.
- [25] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable reasoning over knowledge graphs for recommendation. In *AAAI*, Vol. 33. 5329–5336.
- [26] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable reasoning over knowledge graphs for recommendation. In *AAAI*, Vol. 33. 5329–5336.
- [27] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norrick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. In *WSDM*. ACM, 283–292.
- [28] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *SIGKDD*. ACM, 353–362.
- [29] 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 *AAAI*, Vol. 33. 5941–5948.
- [30] 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 *SIGKDD*. ACM, 1059–1068.