A Human-in-the-loop Approach to Social Behavioral Targeting

Jingru Yang†, Xiaoman Zhao†, Ju Fan†, Gong Chen§, Chong Peng§, Sheng Yao§, Xiaoyong Du†
† *Renmin University of China* § *Tencent Inc.*
{hinsover,xiaomanzhao,fanj,duyong}@ruc.edu.cn; {natchen,michaelpeng,windyao}@tencent.com

***Abstract*—Behavioral targeting plays an important role in social media advertising for capturing users' preferences of ads. While existing studies of behavioral targeting mainly focus on the user behaviors that have explicit correlations with ads, such as ad clicking and web search, many implicit relationships between users and ads, which reside in a variety of heterogeneous sources in social media platforms, are not utilized to enhance the prediction of users' preferences of ads.**

In this paper, we propose a two-pronged approach to behavioral targeting that effectively addresses the above difficulties. First, we model the implicit relationships between users and ads as a heterogeneous information network (HIN), and propose a method that first performs representation learning in the HIN and then uses the learned representations to train a prediction model for boosting the performance of behavioral targeting. Second, we develop a human-in-the-loop framework to address the *incompleteness* challenge in HIN construction that may result in inferior performance of model prediction. The framework judiciously selects the most "beneficial" tasks that ask human for completing the HIN and utilizes the results from human to update the representation learning of HIN. We validate the effectiveness of our approach through extensive experiments on real datasets collected from WeChat, the largest social media platform in China. The experimental results show that our approach is effective at constructing a high-quality HIN at a low cost of human involvement, and the HIN can significantly improve the performance of social behavioral targeting.

I. Introduction

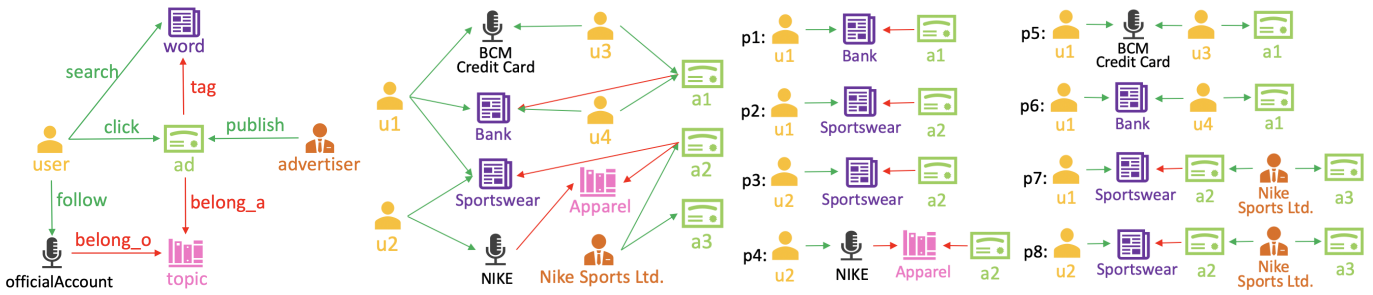
Social media advertising has emerged as the most prevalent and effective advertising method [18], and been widely adopted by major social platform markets like Facebook, Twitter and WeChat. For example, WeChat, one of the largest social platforms in China, utilizes social advertising in its Moments to present ads to *one billion* monthly active users[1]. An essential problem in social media advertising is to capture users' preferences of ads, so as to deliver the right ads to the right users in the right context.

Behavioral targeting (BT) is an effective technique to solve the problem [13], [14]. The basic idea of BT is to collect users' online behaviors, such as web search, and segment the users into different groups. Users in the same group are considered to have similar preferences of ads, and are thus delivered with similar ads. For example, users that frequently search "pasta" and "pizza" can be segmented into one same group and be delivered with similar ads of "pasta" or "pizza". However, in many cases, such user behaviors may not accurately reflect their intent. In our example, a user who searches "pasta" may either want to download recipe Apps or be interested in Italian restaurants. And yet, the existing approaches of BT have limitations on solving the problem, as they only focus on few types of user behaviors with explicit correlation with ads [13]–[17], such as ad clicks and web search. In fact, on real social media platforms like WeChat, there are many implicit relationships between users and ads, which can be utilized to enhance the prediction of users' preferences of ads. Consider again the previous example. Suppose that the user recently follows many official accounts related to cooking. This would increase the likelihood that the user prefers ads of recipe Apps.

To address the problem, we propose to utilize the *implicit* relationships between users and ads from a variety of *heterogeneous* sources. We introduce a heterogeneous information network (HIN) to capture the implicit relationships. Due to its flexibility and expressiveness, HIN has been proposed as a promising approach to modeling heterogeneous data [1], [19]. Figure 2b shows an HIN example with various relationships between users and ads on WeChat platform, such as official accounts, search words, etc. The relationships, some of which are shown in Figure 2c, can be utilized to capture users' preferences of ads. For example, given that user u_2 follows the official account of *NIKE* that belongs to topic *Apparel*, we can conjecture that u_2 will be interested in apparel ads. Thus, one fundamental idea in our approach is to first learn representation of the HIN and then use the learned HIN representation to train a prediction model to boost behavioral targeting performance.

This paper studies the research challenges that naturally arise in utilizing the HIN for behavioral targeting in real advertising platforms like WeChat. First, construction of a high-quality HIN in the advertising scenario is very challenging. Although some types of nodes and edges can be obtained by automatic processes, many others are *incomplete* and need be provided by the human. We introduce a *human-in-the-loop* framework that leverages human for HIN completion: the framework assigns the most "beneficial" tasks to human for verification under a budget. Existing works for crowdsourcing-based HIN completion [5], [6] are designed to select tasks with an objective of inferring *all* missing triplets in the HIN. However, these methods are impractical to solve our problem as our HIN in advertising is of large-scale. Thus, we propose a new strategy: instead of aiming at completing all missing triplets, we want to select the triplets that are helpful for improving the BT prediction model. To support strategy, we introduce a novel utility function for task selection and an

[1]http://3g.163.com/tech/article/E0JHNVMK00097U7R.html



(a) ADS-HIN Schema. (b) ADS-HIN Example. (c) ADS-HIN Schema.

Fig. 1: An Example of HIN in Social Behavioral Targeting.

active selection algorithm with rigorous proof.

The second challenge is how to use the HIN to enhance behavioral targeting. Although some existing works use HIN for recommendation, e.g., movie recommendation [1], HIN enhanced behavior targeting is more challenging due to the more diverse relationships between users and ads and the flexible and dynamic user interests. We devise a graph representation learning model to encode users and ads into low-dimensional embeddings, and feed the embeddings as additional features to a deep learning model for behavioral targeting, such as DeepFM [20] and DSSM [21]. We also design an incremental training method that allows the representation learning model to be efficiently updated after the HIN is completed by the human while preserving training quality.

The contributions of this paper are summarized as follows.

(1) We systemically study HIN enhanced social behavioral targeting on *large-scale* and *real* advertising datasets collected from WeChat, the largest social media platform in China.

(2) We study the incompleteness challenge in HIN construction and propose a human-in-the-loop framework. We introduce a utility function that determines what constitutes a "beneficial" completion task, devise an active HIN completion algorithm based on the utility, and develop an incremental HIN representation learning algorithm.

(3) We conduct an extensive experimental study on WeChat datasets, and the experimental results show that our approach is effective at constructing a high-quality HIN at low cost of human involvement, and the HIN can significantly improve the performance of behavioral targeting.

II. PRELIMINARIES

A. Problem Formulation

This paper studies the behavioral targeting problem based on a heterogeneous information network (HIN) with the support of various types of information sources in advertising. More formally, let $U = \{u\}$ denote a set of users and $A = \{a\}$ denote a set of ads. We define an HIN in advertising as follows.

Definition 1 (HIN in Advertising (ADS-HIN)): The heterogeneous information network in advertising (ADS-HIN) is defined as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a node set $\mathcal{V}$, an edge set $\mathcal{E}$ and two type mapping functions: 1) a node type function $\phi : \mathcal{V} \rightarrow \mathcal{V}_{\texttt{type}}$ that maps a node v to a node type in $\mathcal{V}_{\texttt{type}}$, and 2) an edge type function $\psi : \mathcal{E} \rightarrow \mathcal{E}_{\texttt{type}}$ that maps an edge e to an edge type in $\mathcal{E}_{\texttt{type}}$. In particular, we consider users U and ads A must be included as nodes, i.e., $U \subset \mathcal{V}$ and $A \subset \mathcal{V}$. Schema of the ADS-HIN is defined as a directed graph $\mathcal{G}_S = (\mathcal{V}_{\texttt{type}}, \mathcal{E}_{\texttt{type}})$ where $\mathcal{V}_{\texttt{type}}$ and $\mathcal{E}_{\texttt{type}}$ are the sets of node types and edge types respectively.

Example 1: Figure 2a shows an example schema of the ADS-HIN we built on the data from WeChat. Figure 2b shows an example of ADS-HIN that consists of nodes and edges with the types defined in the schema. We can see that the ADS-HIN contains information from different views, which can be used to capture the potential interest of users on ads. For example, the information from keywords that a user searches (e.g., paths $p_1 - p_3$ in Figure 2c) indicates that the user may be interested in the ads with the keywords as their tags. The information from the official accounts that a user follows (e.g., p_4) indicates that the user may be interested in the ads in similar topics with the official accounts. The information from the behavior of other users that follow the same official account with a user (path p_5) may be useful to infer the user tends to have similar behaviors. In addition, more complicated relationships between users and ads, e.g., paths $p_6 - p_8$, can also be exploited.

Given the ADS-HIN $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we formalize the behavioral targeting problem in advertising as learning a classification function $f : U \times A \rightarrow \{0, 1\}$, where various kinds of heterogeneous information in $\mathcal{G}$ are incorporated to learn the function. More formally, let $\boldsymbol{x}_u$ denote the *intrinsic* features of user u, such as education and age, and $\boldsymbol{x}_a$ denote the intrinsic features of ad a, such as media format and size. Moreover, we use $\boldsymbol{e}_u$ and $\boldsymbol{e}_a$ to represent the learned features from our ADS-HIN $\mathcal{G}$ (more details of how to learn these features will be described later). Then, the problem of HIN enhanced behavioral targeting is defined as follows.

Definition 2 (HIN Enhanced BT): Consider a set of training examples $\mathcal{T} = \{(\boldsymbol{x}_u, \boldsymbol{x}_a; y)\}$ for a behavioral targeting task, where $\boldsymbol{x}_u$ and $\boldsymbol{x}_a$ are receptively intrinsic features of user u and ad a and $y \in \{0, 1\}$ is the targeting result. Consider an ADS-HIN $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and the learned features $\boldsymbol{e}_u$ and $\boldsymbol{e}_a$ for u and a. It aims to learn a classification function $\hat{y} = f(\boldsymbol{x}_u, \boldsymbol{x}_a; \boldsymbol{e}_u, \boldsymbol{e}_a)$ that maps features of a user-ad pair (u, a) into a predicted behavioral targeting result $\hat{y}$.

To illustrate the problem defined above, we use click-through rate (CTR) prediction [20], which is a common evaluation for behavioral targeting as an example.

Example 2: CTR prediction is to predict if a user will click an ad. CTR is an important factor in behavioral targeting, as

the advertising platform normally uses $CTR \times bid$ to estimate the total revenue, where bid is the benefit the platform receives if an ad is clicked by a user. To address this task, we use a set $\mathcal{T}$ of historical clicking behaviors of users on ads as training examples, and with the help of our ADS-HIN, we train a classification function $\hat{y} = f(\boldsymbol{x}_u, \boldsymbol{x}_a; \boldsymbol{e}_u, \boldsymbol{e}_a)$ to predict whether a user u will click ad a in the future.

B. ADS-HIN *Representation Model*

To learn the features of u and a from our ADS-HIN $\mathcal{G}$, we adopt the translation-based embedding model TransE [22], which is widely-adopted for HIN or knowledge graph due to its simplicity and effectiveness. We will study more complicated methods for extracting the ADS-HIN features in the future work. The rationality of using the model is that it can transform nodes and edge types in $\mathcal{G}$ into continuous vector space and preserve the structure of their relationship with external information, so that the nodes that are close in the ADS-HIN will be transformed into similar vectors. Thus, we can easily feed the vectors of user u and ad a, denoted as $\boldsymbol{e}_u$ and $\boldsymbol{e}_a$, as ADS-HIN features into our prediction model.

Formally, we use a triplet $s = (v_h, v_t, r)$ to denote an edge with type r from node v_h to node v_t for ease of presentation, where v_h and v_t are called *head* and *tail* nodes. The basic idea of TransE is to consider, in each triplet, head node v_h can be "translated" into tail node v_t via the specific edge type r, i.e., the learned embeddings of nodes and edge type satisfy $\mathbf{e}_{v_h} + \mathbf{e}_r \approx \mathbf{e}_{v_t}$. TransE introduces an energy score function

$$d(s) = ||\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}||_2, \tag{1}$$

where $||\cdot||_2$ is L_2-norm distance function. Based on this, it defines a margin-based ranking loss function for training the embedding-based representation from our ADS-HIN $\mathcal{G}$, i.e.,

$$\mathcal{L}_{\mathtt{G}} = \sum_{s \in \mathcal{G}} \sum_{s' \in \mathcal{G}'} [\delta + d(s) - d(s')]_+, \tag{2}$$

where $[x]_+ = max\{0, x\}$ and δ is the margin, and $\mathcal{G}'$ is the set of negative triplets that are constructed by replacing head or tail node to a random node. More technical details and examples of TransE can be found in the original papers.

III. OUR HUMAN-IN-THE-LOOP FRAMEWORK

The focus of existing studies for BT is on developing classification models, while their features are limited to only few types of user behaviors with *explicit* correlation with ads [13]–[17], such as ad clicks and web search. Compared with them, we want to learn *implicit* relationships between users and ads from a variety of heterogeneous sources on the advertising platform. Taking them as features can capture the fine-grained user interests. To address the problem, we propose an ADS-HIN to model the heterogeneous relationships and aim at learning "high-quality" predictive features from the ADS-HIN. To this end, we propose a human-in-the-loop framework, as shown in Figure 3, with two steps.

Step 1: Human-in-the-loop ADS-HIN construction. This step aims at constructing a high-quality ADS-HIN based on a predefined schema, as shown in Figure 2b. However, it is challenging to construct the ADS-HIN solely using *automatic* processes. While edges with some specific types (in green color), such as `search` and `follow`, can be automatically created by collecting users' and advertisers' behaviors in the advertising platform, edges with other types (in red color) are not easy to be obtained. For example, in a real advertising platform, the tags of an ad are usually provided by the advertiser, depending on which audience the advertiser wants to reach. Moreover, as ads and official accounts may contain contents in heterogeneous formats, such as text, image and video, the existing topic detection algorithms may not achieve superior performance.

To address this problem, this paper introduces a new *human-in-the-loop* method to construct the ADS-HIN. It first constructs an *incomplete* ADS-HIN based on the data automatically collected in the advertising platform (i.e., edges in green color). Then, it introduces the *completion tasks (or tasks for simplicity)* to solicit human to "complete" the edges that are missing in the ADS-HIN (i.e., edges in red color).

Definition 3 (Completion Task): A completion task $\omega = (v_h, r, \mathcal{V}_{\mathtt{cand}})$ consists of a head node v_h, an edge type r and a set of candidate tail nodes $\mathcal{V}_{\mathtt{cand}}$. Each candidate $v_t \in \mathcal{V}_{\mathtt{cand}}$ corresponds to a possible edge (v_h, v_t) with edge type r, which is also denoted as a triplet $s = (v_h, v_t, r)$ for ease of presentation. The answer of the task is a subset $\mathcal{V}^* \in \mathcal{V}_{\mathtt{cand}}$ of the candidates that are selected by the human.

For example, in Figure 2b, suppose that the `belong_o` edge between official account `NIKE` and topic `Apparel` is missing. We could publish a task with `NIKE` as head node v_h, `belong_o` as edge type r and a set of candidate topics as $\mathcal{V}_{\mathtt{cand}}$, and leverage human to select the right topics for `NIKE`.

The central component in the step is a *Task Selector* module. Most crowdsourcing task selection methods, as surveyed in [2], focus on *general* crowdsourced operations, such as join and sort, and cannot be easily applied to our *specific* problem on HIN completion. Although some methods study crowdsourcing-based HIN completion [5], [6], these methods are designed to select tasks with an objective of inferring *all* missing triplets in the HIN. However, the methods are impractical to solve our problem as the HIN in advertising is of large-scale. Thus, we propose a new strategy: instead of aiming at completing all missing triplets, we want to select the triplets that are helpful for improving the BT prediction model. We introduce a novel utility function and an active selection algorithm with rigorous proof, as introduced in Section IV.

To estimate the utility, we utilize an ADS-HIN *Representation* module that maps each node v and edge type r to a low-dimensional embeddings $\mathbf{e}_v$ and $\mathbf{e}_r$ respectively (see Section II-B). Moreover, we will present the models of deriving utility from embeddings and an active utility-based task selection algorithm in Section IV.

Interface Generator aims at generating user interface for each selected task. As there may be many candidates in $\mathcal{V}_{\mathtt{cand}}$, we first use machine learning (ML) based methods to reduce the size of candidate tail nodes set for each completion task.

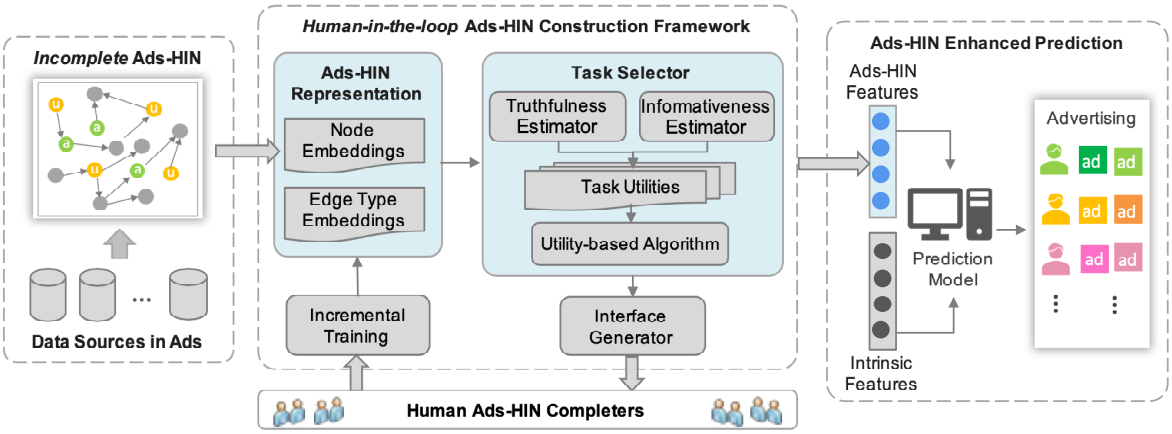


Fig. 2: The Framework of Our Approach.

Then, We design a user interface with the *auto-completion* function that automatically prompting and completing users' inputs to help human easily find the desired answers.

After obtaining answers from the human, our framework introduces an *Incremental Training* module to update the representation of ADS-HIN. As the ADS-HIN contains tens of millions of nodes and edges, we develop an incremental training algorithm that efficiently updates the embeddings while preserving the quality. More details of the algorithm are presented in Section V.

Remarks. (1) In our practice in the real-world advertising platform of WeChat, we have various ways to leverage human for this completion task, e.g., requesting advertisers to provide tags for their ads, asking data labeling employees for the completion task answers and publishing these tasks to the company's internal crowdsourcing platform. (2) Our *auto-completion* interface prompts to complete human's input to help easily find desired answers from many candidates. We also conduct a user study to evaluate the effectiveness of the interface in real-life usage. The result shows that our interface can ensure each task to be completed by human accurately and efficiently. Refer to our technical report [12] for details.

Step 2: ADS-HIN enhanced prediction. This step aims at "extracting" features for users and ads, which are mined from our ADS-HIN constructed in the previous step, and feeds the features into the training process of a prediction model f. Intuitively, the features capture whether user u and ad a are "close" w.r.t. the graph connectivity in $\mathcal{G}$. For example, if u and a are frequently connected in multiple paths in $\mathcal{G}$, we consider that u are more likely to be interested in a.

To this end, we first take the embeddings of user u and ad a learned in the previous step as the ADS-HIN features (i,e., $\mathbf{e}_u$ and $\mathbf{e}_a$). Then, like the conventional solutions for behavioral targeting, we consider the *intrinsic features* for u and a. We use $\boldsymbol{x}_u$ to describe the user profile of u with multiple fields, such as age, gender, education background, etc. We represent the i-th field of user profile as a one-hot vector $\boldsymbol{x}_u^{(i)}$ and denote $\boldsymbol{x}_u$ as $\boldsymbol{x}_u = \boldsymbol{x}_u^{(1)} \oplus \boldsymbol{x}_u^{(2)} \oplus \ldots \boldsymbol{x}_u^{(m)}$. Similarly, we use $\boldsymbol{x}_a$ to represent features of ad a with multiple fields, such as media format (image or video) and size. We next utilize the feature concatenation method, which is simple and widely adopted for feature enhancement [23], [24], to enhance the intrinsic features of behavioral targeting with the ADS-HIN features.

Finally, we adopt a binary classification model for a specific behavioral targeting task, such as DeepFM [20] for CTR prediction and DSSM [21] for ad matching. Given a user u and an ad a, we use the concatenated feature $\boldsymbol{x} = \boldsymbol{x}_u \oplus \boldsymbol{x}_a \oplus \boldsymbol{e}_u \oplus \boldsymbol{e}_a$ to learn a classification model $\hat{y} = f(\boldsymbol{x})$. The model is learned on training examples $\mathcal{T} = \{(\boldsymbol{x}_u, \boldsymbol{x}_a; y)\}$, and is then used to predict behavioral targeting for new user-ad pairs.

IV. UTILITY-BASED TASK SELECTION

A. Task Utility Estimation

Intuitively, we favor the "informative" tasks, if completed by the human, which would be most helpful to enhance our prediction model. To this end, we propose a *utility* function that considers two factors.

- The first factor is whether a task contains the edges that could be verified to be true, e.g., whether an advertiser can find the desired tags from the candidates for her ads. We prefer tasks that contain edges with high possibility of being true, in order to avoid waste of our budget.
- The second factor is the degree of *informativeness* of edges contained in a task, if completed by the human, on improving the performance of our prediction model. We prefer the edges, which, if added into our ADS-HIN $\mathcal{G}$, would provide additional evidence to our prediction model that a user u may be interested in an ad a.

We formalized the two factors for each triplet $s = (v_h, v_t, r)$. The first factor *truthfulness* $P(v_t|v_h, r)$ represents the probability that triplet $s = (v_h, v_t, r)$ can be successfully completed by the human. The second factor *informativeness* $\mathcal{I}(v_h, v_t, r)$ captures the degree of triplet $s = (v_h, v_t, r)$ on enhancing our prediction model. Based on the factors, utility of task ω is defined as the average degree of informativeness that would be brought by the candidate triplets of ω, i.e.,

$$\mathcal{U}(\omega) = \frac{1}{|\mathcal{V}_{\text{cand}}|} \sum_{v_t \in \mathcal{V}_{\text{cand}}} P(v_t|v_h, r) \cdot \mathcal{I}(v_h, v_t, r). \quad (3)$$

To save time, for each completion task, we randomly sample a subset of candidate tail nodes to estimate utility.

1) Estimation of Truthfulness: Intuitively, the truthfulness $P(v_t|v_h, r)$ of a triplet $s = (v_h, v_t, r)$ captures the probability that v_t will be selected from the set of candidate tail nodes $\mathcal{V}_{\texttt{cand}}$ by the human for the completion task $\omega = (v_h, r, \mathcal{V}_{\texttt{cand}})$. We use our learned HIN embedding using TransE to estimate $P(v_t|v_h, r)$ (see Section II-B). The intuition is thatthe smaller the translational distance $d(s)$ (defined in Equation (1)) is, the larger the probability that an edge with type r exists from v_h to v_t, i.e., the more likely that v_t can be selected from $\mathcal{V}_{\texttt{cand}}$. Thus, we adopt the translation-based energy function (TEF) [9] on top of $d(s)$ to estimate the truthfulness, i.e.,

$$P(v_t|v_h, r) = \frac{1}{1 + e^{-\alpha(\tau_r - d(s))}}, \tag{4}$$

where α is a hyper-parameter used for smoothing, and τ_r is a threshold related to edge type r. We can easily obtain τ_r by using a common method in the existing triple classification works [9], [10]. The basic idea is to use a validation set containing triplets of type r and compute τ_r by optimizing classification accuracy on the validation set. We can see that, when $d(s) = \tau_r$, the value of $P(v_t|v_h, r)$ is 0.5, and when $d(s) < \tau_r$, $P(v_t|v_h, r) > 0.5$. In our experiments, we use the source code of OpenKE [11] to this end.

2) Estimation of Informativeness: Intuitively, the informativeness $\mathcal{I}(s)$ of triplet s captures the influence power of the selected triplet on the prediction model, in other words we introduce *informativeness* to measure how helpful a triplet is to influence our prediction model. The key idea is that a triplet, if completed by human, will influence the translational distance between users and ads in the ADS-HIN. Take Figure 2b as an example. We consider the completion from u_2 to `Sportswear` may be more informative than that from u_1 to `Bank`. The reason is that the former, if completed by the human, will connect two disconnected nodes u_1 and a_2 by path p_2, which indicates that the user u_1 may be interested in the ad a_2 with the keywords `Sportswear` as her tag. In contrast, the latter may be less useful as it uses path p_1 to connect u_2 and a_1, which are already connected by paths p_5 and p_6 as shown in Figure 2c. In other words, after connecting two disconnected nodes u_1 and a_2 by the completion from u_2 to `Sportswear`, TransE model will transform nodes u_1 and a_2 into more "close" vectors in the embedding space, which would provide additional evidence to our prediction model that u_1 may be interested in a_2.

From the example, we can see that we prefer the triplets, which, if added into our ADS-HIN $\mathcal{G}$, would have large effect on updating the learned embeddings of users and ads, i.e., $\{\boldsymbol{e}_u\}$ and $\{\boldsymbol{e}_a\}$, which would result in the translational distance between u and a to be significantly changed. As our prediction model takes the embeddings as its input, these triplets essentially give us more chances to update our prediction model f for behavioral targeting. Note that this is conceptually similar to a widely-adopted criterion in active learning [25], [26] that prefers the data points having larger effect on updating the model.

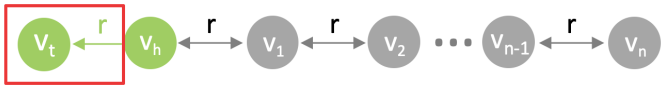


Fig. 3: Simple Example Graph.

Formally, let us consider the set of user-ad pairs in our training set $\mathcal{T} = \{(u, a)\}$ and our ADS-HIN $\mathcal{G}$. We use $d_{\mathcal{G}}(u, a, r_c)$ to represent the translational distance between u and a given the edge type r_c corresponding to the behavioral targeting task. For example, considering the CTR prediction task, r_c corresponds to the edge type `click`. Moreover, let $d_{\mathcal{G}\cup\{s\}}(v_u, v_a, r_c)$ denotes the translational distance of (u, a) after a triplet s is added to $\mathcal{G}$. We define the informativeness $\mathcal{I}(s)$ of triplet s as *changes* on the translational distances among all user-ad pairs in our training set $\mathcal{T}$, i.e.,

$$\mathcal{I}(s) = \sum_{(u,a)\in\mathcal{T}} (d_{\mathcal{G}}(v_u, v_a, r_c) - d_{\mathcal{G}\cup\{s\}}(v_u, v_a, r_c))^2. \tag{5}$$

The non-trivial part in Equation (5) is the computation of the translational distance $d_{\mathcal{G}\cup\{s\}}(v_u, v_a, r_c)$ in the updated $\mathcal{G}\cup\{s\}$. A straightforward way is to perform graph embedding learning, which is described in Section II-B, in $\mathcal{G} \cup \{s\}$. However, this method is very prohibitive, as it needs to rerun graph embedding learning for each possible triplet s.

To address the problem, we propose an effective approach to estimate $d_{\mathcal{G}\cup\{s\}}(v_u, v_a, r_c)$ instead of re-running the embedding learning process. The basic idea of the approach is to estimate the updated embedding $\boldsymbol{e}_v$ for each node v in the updated $\mathcal{G}\cup\{s\}$, based on the definition of $d_{\mathcal{G}\cup\{s\}}(v_u, v_a, r_c)$. Note that, since the update of embedding $\boldsymbol{e}_r$ of edge type r can be derived by the update of its head and tail nodes, to simplify the presentation, we ignore the update of $\boldsymbol{e}_r$ in this section. For ease of presentation, we use a simple graph as shown in Figure 4 to describe our estimation method, and we will discuss the general graph later.

Let (v_h, v_t, r) be a triplet that is newly added into the graph. The embeddings of each node v_n in the graph will be updated as the gradient information of (v_h, v_t, r) is propagated through the interlink in Figure 4. For simplicity of the estimation, we approximate the propagation as the first iteration update of the embedding of v_n caused by (v_h, v_t, r), similar approximate method of the existing active learning work [27]. Formally, let $\mathbf{e}^+_{v_n}$ denotes the new embedding of v_n after the completion of triplet (v_h, v_t, r).

Lemma 1: Consider the simple graph in Figure 4 and a newly added triplet (v_h, v_t, r). The updated embedding $\mathbf{e}^+_{v_n}$ can be approximated as:

$$\mathbf{e}^+_{v_n} \leftarrow \mathbf{e}_{v_n} - (2\gamma)^{n+1}(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}), \tag{6}$$

where n is the length of interlink connecting v_n and v_h, γ is the learning rate for back propagation.

Proof 1 (Proof of Lemma 1): Consider the simple graph in Figure 4, let (v_h, v_t, r) be a triplet that is newly added into the graph. The embeddings of each node v in the graph will be updated as the gradient information of (v_h, v_t, r) is propagated through the interlink. We approximate the propagation as

the first iteration update of the embedding of v_n caused by (v_h, v_t, r), similar to approximate method of the existing active learning work [27]. According to gradient descent, TransE updates embedding of each node v as

$$\mathbf{e}_v \leftarrow \mathbf{e}_v - \gamma \cdot \nabla(\mathbf{e}_v), \tag{7}$$

where γ is the learning rate for back propagation. $\nabla(\mathbf{e}_v)$ is the gradient of $\mathbf{e}_v$. When v is the head or tail node of a triplet and o is another node of the triplet, $\nabla(\mathbf{e}_v)$ has the following two different forms:

$$\nabla(\mathbf{e}_v) = \begin{cases} 2(\mathbf{e}_v - \mathbf{e}_o + \mathbf{e}_r) & (v, o, r) \in \mathcal{G} \\ 2(\mathbf{e}_v - \mathbf{e}_o - \mathbf{e}_r) & (o, v, r) \in \mathcal{G}. \end{cases} \tag{8}$$

So, we can rewrite Equation (7) as follows:

$$\mathbf{e}_v \leftarrow \mathbf{e}_v - 2\gamma \cdot (\mathbf{e}_v - \mathbf{e}_o \pm \mathbf{e}_r). \tag{9}$$

Let $\mathbf{e}^+_{v_x}$ denote the updated embedding of node v_x after the completion of triplet (v_h, v_t, r). According to Equation (9), the difference between the updated embeddings of node v_n before and after the triplet (v_h, v_t, r) being completed is

$$\begin{aligned} \mathbf{e}_{v_n} - \mathbf{e}^+_{v_n} &\leftarrow \mathbf{e}_{v_n} - 2\gamma(\mathbf{e}_{v_n} - \mathbf{e}_{v_{n-1}} \pm \mathbf{e}_r) \\ &\quad - \mathbf{e}_{v_n} + 2\gamma(\mathbf{e}_{v_n} - \mathbf{e}^+_{v_{n-1}} \pm \mathbf{e}_r) \\ &= 2\gamma(\mathbf{e}_{v_{n-1}} - \mathbf{e}^+_{v_{n-1}}) \\ &= (2\gamma)^{n-1}(\mathbf{e}_{v_1} - \mathbf{e}^+_{v_1}). \end{aligned} \tag{10}$$

According to Equation (10), in order to obtain the expression of $\mathbf{e}^+_{v_n}$, we need to calculate $\mathbf{e}_{v_1} - \mathbf{e}^+_{v_1}$. To this end, we first write the expression of $\mathbf{e}^+_{v_1}$ as follow:

$$\begin{aligned} \mathbf{e}^+_{v_1} &\leftarrow \mathbf{e}_{v_1} - 2\gamma(\mathbf{e}_{v_1} - \mathbf{e}^+_{v_h} \pm \mathbf{e}_r) \\ &= \mathbf{e}_{v_1} - 2\gamma(\mathbf{e}_{v_1} - \mathbf{e}_{v_h} + 2\gamma(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}) \pm \mathbf{e}_r) \\ &= \mathbf{e}_{v_1} - 2\gamma(\mathbf{e}_{v_1} - \mathbf{e}_{v_h} \pm \mathbf{e}_r) - (2\gamma)^2(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}) \\ &= \mathbf{e}_{v_1} - (2\gamma)^2(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}). \end{aligned} \tag{11}$$

Then we can easily get $\mathbf{e}_{v_1} - \mathbf{e}^+_{v_1} \leftarrow (2\gamma)^2(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t})$. Combined with Equation (10), we can obtain the following expression of $\mathbf{e}^+_{v_n}$:

$$\mathbf{e}^+_{v_n} \leftarrow \mathbf{e}_{v_n} - (2\gamma)^{n+1}(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}), \tag{12}$$

where n is the length of the path connecting v_n and v_h. Hence, we prove the lemma.

We can extend Lemma 1 to the more general case that there are multiple paths between a node v and the newly completed triplet $s = (v_h, v_t, r)$, i.e.,

$$\mathbf{e}^+_v = \mathbf{e}_v - (\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t}) \left[\sum_{p_v^{v_h} \in P_v^{v_h}} (2\gamma)^{|p_v^{v_h}|+1} - \sum_{p_v^{v_t} \in P_v^{v_t}} (2\gamma)^{|p_v^{v_t}|+1} \right], \tag{13}$$

where $P_v^{v_h}$ is the set of paths between v_h and v, and $P_v^{v_t}$ is the set of paths between v_t and v. For simplicity, we use ρ_v to denote $\sum_{p_v^{v_h} \in P_v^{v_h}} (2\gamma)^{|p_v^{v_h}|+1} - \sum_{p_v^{v_t} \in P_v^{v_t}} (2\gamma)^{|p_v^{v_t}|+1}$.

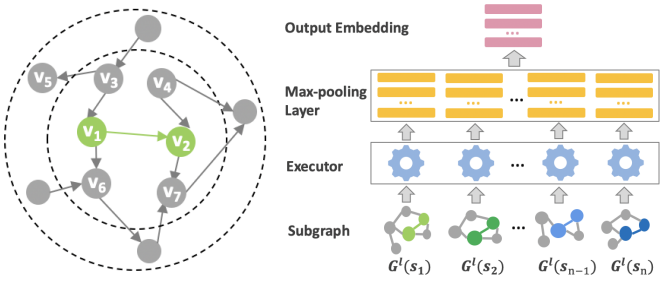
Now we are ready to obtain the informativeness $\mathcal{I}(s)$ of triplet s by applying Equation (13) in Equation (5), i.e,

$$\begin{aligned} \mathcal{I}(s) &= \sum_{(u,a) \in \mathcal{T}} \left(||\mathbf{e}_{v_u} + \mathbf{e}_{r_c} - \mathbf{e}_{v_a}||^2 - ||\mathbf{e}^+_{v_u} + \mathbf{e}_{r_c} - \mathbf{e}^+_{v_a}||^2 \right)^2 \\ &= \sum_{(u,a) \in \mathcal{T}} \Big(2(\mathbf{e}_{v_u} + \mathbf{e}_{r_c} - \mathbf{e}_{v_a})(\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t})(\rho_{v_u} - \rho_a) \\ &\quad - (\mathbf{e}_{v_h} + \mathbf{e}_r - \mathbf{e}_{v_t})^2(\rho_{v_u} - \rho_{v_a})^2 \Big)^2 \end{aligned} \tag{14}$$

Algorithm 1: Active ADS-HIN Completion

Input: $\mathcal{G}$: current ADS-HIN; Ω: completion tasks; B: a budget; b: a task batch
Output: $\mathcal{G}$: completed ADS-HIN

1 Train a TransE model on $\mathcal{G}$;
2 **for** *each iteration* **do**
3 Estimate $\mathcal{U}(\omega)$ for each task ω in Ω;
4 Select b tasks Ω^* from Ω with the largest $\mathcal{U}(\omega)$;
5 Ask the human to complete the tasks in Ω^* ;
6 Add the triplets completed by the human in $\mathcal{G}$;
7 Retrain the TransE model on $\mathcal{G}$;
8 $B \leftarrow B - b$;
9 $\Omega \leftarrow \Omega - \Omega^*$;
10 **if** $B = 0$ *or* $\Omega = \emptyset$ **then** break;
11 Return $\mathcal{G}$;



(a) Subgraph Example. **(b) Parallel Training.**

Fig. 4: Subgraph Parallel Incremental Training.

B. Active Utility-Based Completion Algorithm

The pseudo-code of our utility-based completion algorithm is shown in Algorithm 1. It takes as input a current ADS-HIN $\mathcal{G}$, a set Ω of completion tasks, a budget constraint B on the number of tasks and a batch size b. The output of the algorithm is an updated $\mathcal{G}$ completed by the human. Initially, the algorithm first trains a TransE model [22] on $\mathcal{G}$ (line 1). Then it repeats a task selection iteration until the budget B is exhausted or there is no remaining tasks (lines 2-10). In each iteration, it first estimates the utility $\mathcal{U}(\omega)$ of each task ω in Ω based on Equation (3) (line 3). Then, it delivers b tasks with the highest utility to human, collects the answers and adds the newly completed triplets into $\mathcal{G}$ (lines 4-6). Finally, it retrains the TransE model on the updated $\mathcal{G}$ and continues to the next iteration (line 7).

In the Algorithm 1, the training of TransE has a space complexity of $O((|\mathcal{V}|+|\mathcal{E}_{\texttt{type}}|)*m)$, where m is the dimension of embedding, $|\mathcal{V}|$ is the number of nodes and $|\mathcal{E}_{\texttt{type}}|$ is the number of edge types in $\mathcal{G}$. And in the phase of task utility estimation, at each iteration we use the heap sort algorithm to select top b tasks with the highest utility score, which has the space complexity of $O(b)$.

V. INCREMENTAL REPRESENTATION TRAINING

One obstacle in Algorithm 1 is the efficiency of retraining the ADS-HIN representation. According to Equation (2), the

training time complexity of TransE is $O(mn)$, where m is the dimension of embedding and n is the number of triplets that need to be trained for updating. Obviously, it is highly time-consuming to combine newly completed triplets with the existing ADS-HIN which contains tens of millions of triplets to retrain the TransE model from scratch. To address the obstacle, we introduce an incremental training algorithm that consists of two strategies to improve the training efficiency while ensuring the training quality.

The first strategy is to update embeddings of a subgraph of $\mathcal{G}$, instead of the entire $\mathcal{G}$. As the example shown in Figure 5a, the triplets in an HIN are connected by paths with different lengths. We use l_v^s to denote the length of the shortest path connecting a node v and a triplet s, that is, the length of the shortest path from v to the head node or tail node of s. For example, $l_{v_3}^{(v_1,v_2,r)} = l_{v_4}^{(v_1,v_2,r)} = 1$ because the length of the shortest paths from v_3 to v_1 and v_4 to v_2 are 1. Then we define the l-subgraph of a triplet s, denoted by $\mathcal{G}^l(s)$, as a set of triplets that do not exceed l hops from the triplet s, which can be formalize as:

$$\mathcal{G}^l(s) = \{(v_h, v_t, r) | l_{v_h}^s \leq l \text{ or } l_{v_t}^s \leq l\}, (v_h, v_t, r) \in \mathcal{G}.$$

Note that, the l-subgraph of a triplet contains the triplet itself. For example, $\mathcal{G}^0(v_1, v_2, r) = \{(v_1, v_2, r), (v_3, v_1, r), (v_1, v_6, r), (v_4, v_2, r), (v_2, v_7, r)\}$ because the first three triplets contain the node v_1 and the last two contain the node v_2, which leads to the length of the shortest paths from these triplets to (v_1, v_2, r) being 0. Observing from the training result of TransE, we find that the effect of a new triplet s on the embedding of a node decreases exponentially with the increase of the length of the path from s to the node. Based on this observation, after adding a new triplet s, we can use TransE model to only update the embeddings of triplets in $\mathcal{G}^l(s)$, where l is a hyper-parameter that can be tuned.

The second strategy is to train the embeddings in the l-subgraph of each newly added triplet in parallel for further improving the training efficiency, which is shown as Figure 5b. Given a set of newly added triplets $S = \{s_1, s_2, \cdots, s_n\}$, according to the previous strategy, we extract their l-subgraphs $\mathcal{G}^l(S) = \{\mathcal{G}^l(s_1), \mathcal{G}^l(s_2), \cdots, \mathcal{G}^l(s_n)\}$. Since each l-subgraph can retain the graph structure information related to the newly added triplets independently, we evenly assign the l-subgraphs of the newly added triplets to different executors for parallel training. In particular, for a node or edge type that is assigned to multiple executors, we input the embeddings obtained from these executors into a max-pooling layer to obtain a unified embedding, for keeping the most prominent features [28].

VI. EXPERIMENTS

A. Experiment Setup

Datasets. We construct a large-scale dataset based on data collected from the real social advertising platform in WeChat. The dataset consists of an ADS-HIN and the training sets for two representative behavioral targeting tasks. First, we build an ADS-HIN $\mathcal{G}$ based on the information from different views in WeChat to capture the potential interest of users on ads. Note that all data is anonymized for protecting user privacy. Specifically, We consider 8 types of nodes and 11 types of edges (more details can be found in our technical report [12]).

TABLE I: Statistics of ADS-HIN in WeChat Platform

# of nodes with various types			
user	1,167,481	ad	541,118
advertiser	43,188	official_account	255,221
word	243,048	category	236
brand	1,929	tag	256
# of edges with various types			
click	554,691	interact	47,425
un-interested	248,850	advertise	525,422
follow	4,101,158	interested_in_tag	999,200
interested_in_word	998,882	mention	2,737,389
belong_to_topic	792,377	belong_to_brand	1,192,886
advertising_to_tag	1,371,282		

The statistics of the ADS-HIN are shown in Table II. Four of the edge types, namely `belong_to_topic`, `belong_to_brand`, `mention` and `advertising_to_tag`, need to be completed by the human in the real-world application, as the automatic process cannot provide accurate results for these types, which results in 2,674,914 of completion tasks in total. In order to conduct experiments that fairly compare various approaches, we use the pre-accumulated completion results from data labeling employees and advertisers in WeChat platform to simulate humans' answers to these tasks. Edges with other types, such as `click` and `follow`, can be automatically constructed based on the data in WeChat.

Moreover, we construct datasets for two representative behavioral targeting tasks in social advertising. The first task is *ad matching* that retrieves the top N related ads for a user from an enormous ad corpus, and the second one is *CTR prediction* that predicts whether a user will click the ad that is delivered to the user. Following the existing works [20], [21], both tasks take the user-ad instances with click records as the positive samples, while they have different ways of generating negative samples. Specifically, *ad matching* takes all unobserved user-ad instances as the negative samples, while CTR prediction takes the user-ad instances in which the ads that are delivered to the users but not clicked as the negative ones. For each task, we prepare a dataset based on the real data log from WeChat, where each data instance consists of the intrinsic features of users and ads, which are presented previously, and an output for prediction, e.g., matched or clicked. We take the log data of one week as the training set, and take the data of the day next to the week as the test set. Based on this, we obtain a dataset with $5,380,531$ data instances for CTR prediction where the ratio between positive and negative instances is $1:5$, and a data set for ad matching with a positive/negative ratio $1:500$.

Prediction Models. To evaluate the effect of the ADS-HIN as auxiliary data to enhance ad matching and CTR prediction tasks, we concatenate the HIN features and the intrinsic features and use the following state-of-the-art predict models for evaluation. (1) `DSSM` [21] is one of the most widely used ad matching models, which uses two deep neural networks to convert the one-hot encoded user features and ad features

TABLE II: Evaluating HIN Feature Enhancement.

(a) Evaluating on Ad Matching Task.

Method	HR@0.1%		HR@5%	
	DSSM	CDSSM	DSSM	CDSSM
NonHIN	0.016	0.017	0.262	0.247
PartHIN	0.064	0.068	0.435	0.412
CompHIN	0.116	0.117	0.557	0.550

(b) Evaluating on CTR Prediction Task.

Method	AUC		LogLoss	
	DeepFM	Fibinet	DeepFM	Fibinet
NonHIN	0.692	0.694	0.437	0.435
PartHIN	0.706	0.709	0.434	0.432
CompHIN	0.727	0.728	0.423	0.423

into low-dimensional vectors separately. (2) CDSSM [29] is a variant of DSSM, which incorporates a convolutional pooling structure over user features and ad features to learn their low-dimensional vectors. (3) DeepFM [20] is one of the most widely used CTR prediction model, which combines the power of factorization machines and deep learning. (4) Fibinet [30] is a state-of-the-art CTR prediction model, which combines feature importance and bilinear feature interaction.

Evaluation metrics. For the ad matching task, we use Hit Ratio (HR@n%) as the evaluation metric, which measures the ratio of matched ads in the top n% prediction results returned by the prediction model. For the CTR prediction task, we measure the performance using the following two metrics: 1) area under the ROC curve, i.e., Area Under Curve (AUC), and 2) binary cross entropy loss (LogLoss), which are widely used in the existing works for CTR [20], [30].

Parameter settings. For triplet completion, we pick up 10,000 triplet tasks at each iteration. For TransE model training, we set the dimension of the embeddings as 100 and margin as 1.0. For incremental graph training, at each iteration of the ADS-HIN completion, we set the subgraph path threshold l as 2 and use the multi-process method to set up 4 GPU executors for parallel training. For the prediction models, we use the hyper-parameters which are reported effective in their original papers. More specifically, for DSSM, we use two three-layer neural networks whose number of units in the first layer is 300 and the number of units in other layers is 128. For CDSSM, on the basis of DSSM, we use 32 filters in the convolution and a kernel size of 3. For DeepFM, we use a two-layer neural network with 64 hidden units and an embedding size of 32. For Fibinet, we use a three-layer neural network with 400 hidden units and an embedding size of 50. For the training of all the models, we adopt early stopping [31] to avoid over-fitting.

Experiments are conducted on a Ubuntu 16.04 server with 2TB disk, 40 CPU cores (Intel Xeon CPU E5-2630 v4 @ 2.20GHz), one GPU (NVIDIA TITAN V) and 512GB memory.

B. Evaluation on HIN Features.

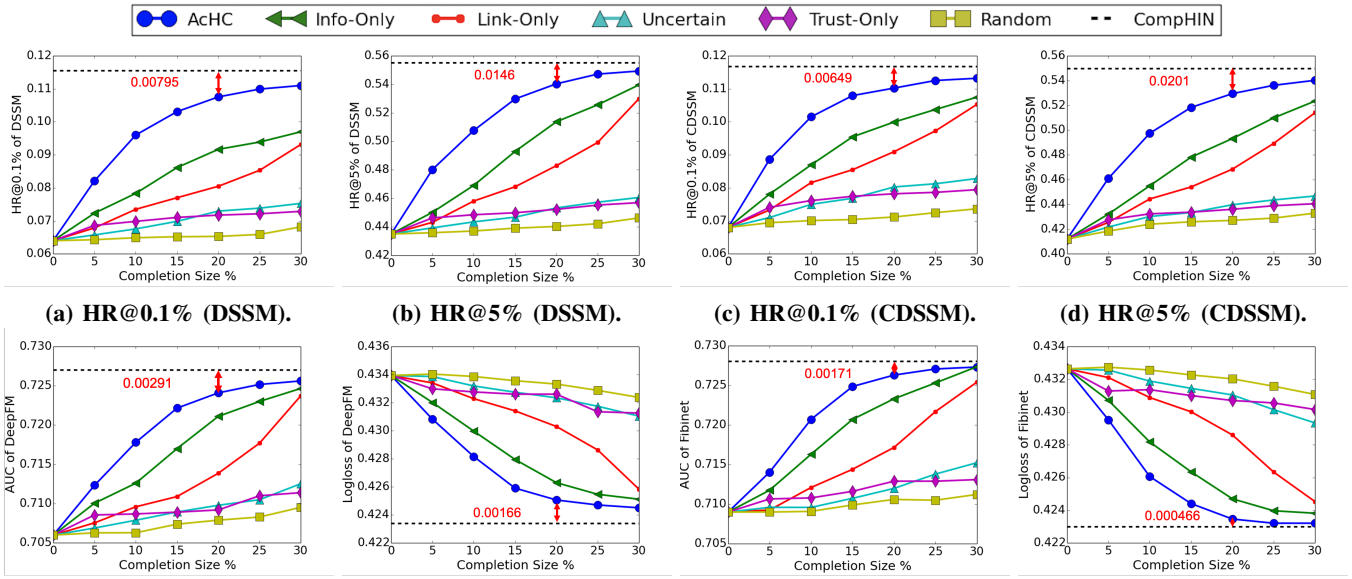
We first evaluate the effect of HIN features mined from our ADS-HIN on enhancing the prediction model. To this end, we consider three settings of the features that are fed into the prediction model. (1) NonHIN only feeds the intrinsic features of users and ads. (2) PartHIN only considers the *partial* ADS-HIN without human completion, and concatenates the HIN features learned from the graph with the intrinsic features to the prediction model. (3) CompHIN is similar to PartHIN where the only difference is that all the completion tasks regarding ADS-HIN have been completed by the human.

Table III shows the experimental results. We can see that the consideration of HIN features can improve the performance of the two tasks significantly. For example, PartHIN has up to 300% improvements on HR@0.1% and 2.16% improvements on AUC, and CompHIN has up to 625% improvements on HR@0.1% and 5.06% improvements on AUC. Moreover, for the two tasks, CompHIN outperforms PartHIN, e.g., 28%-81% improvements on HR@n%, up to 2.97% improvement on AUC, and up to 2.6% reductions on LogLoss. These observations, on the one hand, validate our claim that the HIN features are useful to enhance the prediction model for behavioral targeting, and, on the other hand, show the necessity of human-in-the-loop ADS-HIN completion.

C. Evaluation on Active HIN Completion.

This section evaluates the effectiveness of our active utility-based task selection algorithm in Section IV-B. We consider the following baseline methods for selecting tasks. (1) *Random* randomly selects tasks for completion. (2) *Uncertain* prefers to select tasks with uncertain triplets, which is commonly used in active learning methods. Specifically, a triplet is uncertain if its truthfulness is close to 0.5. Formally, we use the information entropy to measure the uncertainty. (3) *Link-Only* utilizes the reciprocal of the length of the shortest path between each triplet and each user-ad pair in the ADS-HIN. (4) *Trust-Only* is a variant of our triplet utility score in Equation (3) that only considers the truthfulness while ignoring the informativeness. (5) *Info-Only* is a variant of our triplet utility score in Equation (3) that only considers the informativeness while ignoring the truthfulness. For ease of presentation, we denote our utility-based strategy as ACHC. For more comprehensive comparisons, we vary the budget for HIN completion, which is the percentage of tasks (from zero to 30%) published for completion in all candidate tasks.

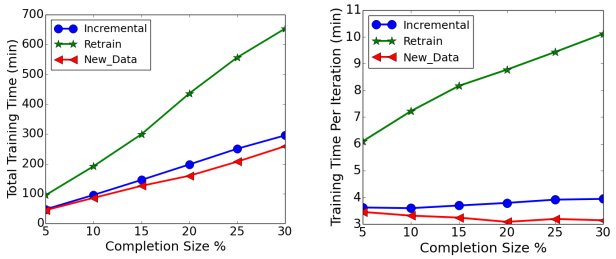
Figure 6 reports the experimental results. We can see that *Random* performs the worst, which shows that active HIN completion is necessary. The two baselines *Uncertain* and *Trust-Only* also achieve inferior performance, because they ignore if the newly added triplets are informative to improve the prediction model. *Uncertain* only considers if the newly added triplets are helpful to reduce the generalization error of TransE model and *Trust-Only* only considers whether more triplets can be added in ADS-HIN. Moreover, *Link-Only* achieves better performance than the above two baselines. This is because *Link-Only* prefers the triplets that are closed to the user-ad pairs in our training set $\mathcal{T}$. Recall our previous analytics that the effect of triplets on the embeddings of a node decreases exponentially with the increase of the length of the paths from the triplets to the node. Due to this reason, *Link-Only* gives more chances for users and ads to update the embeddings



(a) HR@0.1% (DSSM). **(b) HR@5% (DSSM).** **(c) HR@0.1% (CDSSM).** **(d) HR@5% (CDSSM).**

(e) AUC (DeepFM). **(f) LogLoss (DeepFM).** **(g) AUC (Fibinet).** **(h) LogLoss (Fibinet).**

Fig. 5: Evaluating Completion Strategies on Ad Matching and CTR Prediction Tasks.



(a) Total Training Time of ADS-HIN Representation. **(b) Training Time per Human-Completion Iteration.**

Fig. 6: Evaluating Efficiency of Re-training Methods.

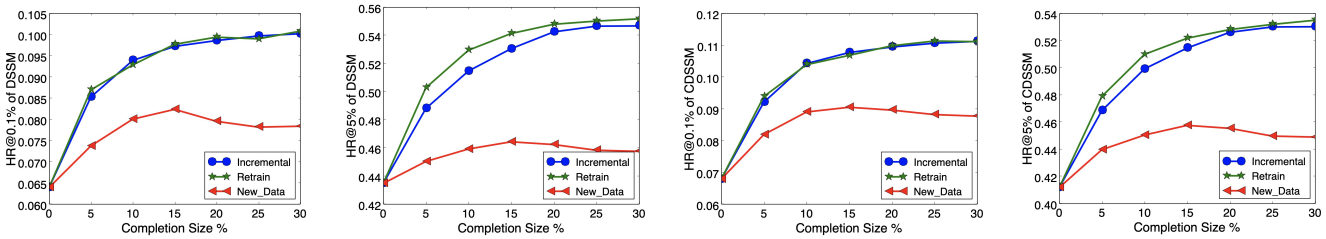
after the triplet completion, which may further improve the prediction model. However, *Info-Only* performs better than *Link-Only*, because it considers the distances in translational distances between users and ads, which would further improve the informativeness. Moreover, ACHC further improves *Info-Only*, as it considers both truthfulness and informativeness in measuring the task utility. We can also see that ACHC achieves nearly the results of `CompHIN` with only 20% of the triplet tasks. These results demonstrate the effectiveness of our completion method for real large-scale advertising that reduces the effort of preparing the ADS-HIN for BT.

D. Evaluation on Incremental Training.

This section demonstrates the efficiency and effectiveness of the incremental training strategy introduced in Section V, we consider the following baseline training methods. (1) `Retrain` adds newly completed triplets into our ADS-HIN and retrains our ADS-HIN representation model from scratch. (2) `NewData` only considers the newly completed triplets to fine-tune the representation model, while keeping embeddings of other nodes trained on the previous iterations unchanged.

We first compare the efficiency of the three training strategies. For more comprehensive comparisons, we vary the percentage of tasks (from 5% to 30%) published for completion in all candidate tasks. Figure 7a reports the total time for all the iterations of the representation training. We can see that `Incremental` significantly reduces the total training time over `Retrain`, because it only considers the path within length threshold of 2 and reduces the training data by nearly 82% in each iteration of the ADS-HIN updating. On the other hand, although `NewData` can reduce 97% of the training data in each iteration, it does not significantly outperform `Incremental` as the parallel training of `Incremental` further saves time. We also report the training time for each iteration of the ADS-HIN completion in Figure 7b. We can see that as the number of completed triplets increases, the training time of `Retrain` per iteration increases significantly due to the increase in the number of triplets, including the original triplets and the newly completed triplets, that need to be updated. However, the training time of both `Incremental` and `NewData` at each iteration is almost unchanged. For `NewData`, because the number of triplets that need to be updated at each iteration, i.e., the number of completion tasks published at each iteration, is almost the same. For `Incremental`, as the ADS-HIN is gradually completed, the number of triplets in each l-subgraph will increase. Fortunately, the parallel training can further alleviate that effect and reduce the training time.

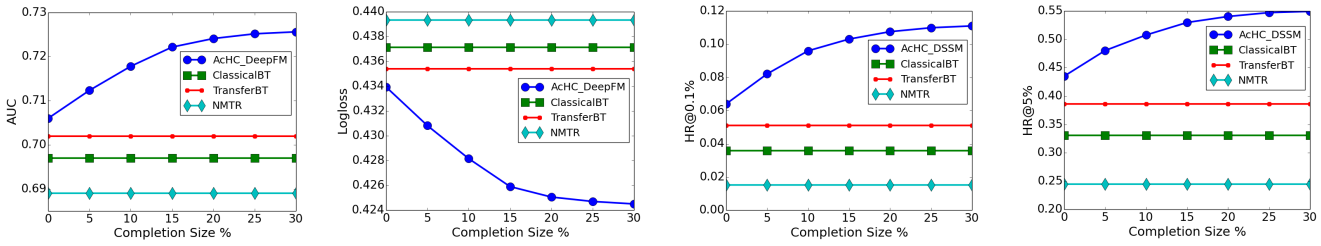
Next, we compare the performance of the three training methods in Figure 8. We observe that `NewData` performs the worst, because it fine-tunes the TransE model trained on the previous iterations by only the newly completed triplets, which may lose much information in the graph. Second, `Retrain` uses all the triplets, including the original triplets and the newly completed triplets to retain the graph, and it achieves the best performance. However, the differences in performance between `Incremental` and `Retrain` is insignificant. In particular, as the number of completed triplets increases, the difference becomes even smaller.



(a) HR@0.1% (DSSM). (b) HR@5% (DSSM). (c) HR@0.1% (CDSSM). (d) HR@5% (CDSSM).

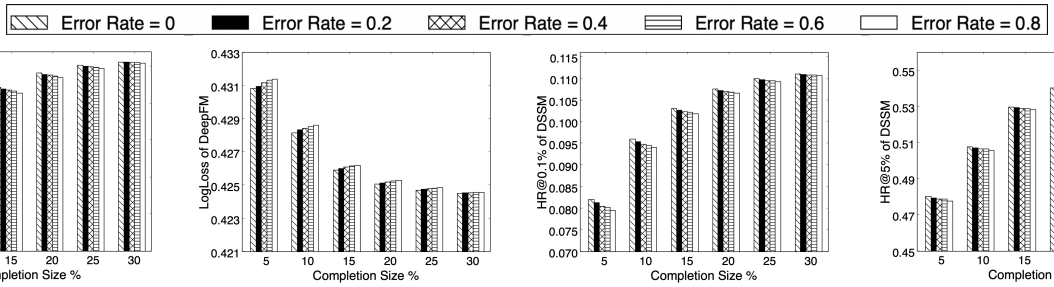
(e) AUC (DeepFM). (f) LogLoss (DeepFM). (g) AUC (Fibinet). (h) LogLoss (Fibinet).

Fig. 7: Evaluating Training Methods on Ad Matching and CTR Prediction Tasks.



(a) AUC. (b) LogLoss. (c) HR@0.1%. (d) HR@5%.

Fig. 8: Comparison with Behavioral Targeting Methods.



(a) AUC (DeepFM). (b) LogLoss (DeepFM). (c) HR@0.1% (DSSM). (d) HR@5% (DSSM).

Fig. 9: Robustness against Human Errors on Ad Matching and CTR Prediction Tasks.

E. Comparisons for Behavioral Targeting.

We compare the performance of our proposed HIN enhanced behavioral targeting with the following 3 representative and state-of-the-art classical behavioral targeting methods which are based on the behavior modeling. (1) `ClassicalBT` [13] builds the behavior matrices based on the classical Term Frequency & Inverse Document Frequency (TFIDF) indexing [32] for different behaviors to obtain the behavior feature vector of each user. (2) `TransferBT` [14] improves `ClassicalBT` by utilizing the transfer learning strategy across different behavioral domains. (3) `TemporalBT` [33] models the temporal interest change patterns of users to obtain the time-based behavior features. For a fair comparison, we concatenate the intrinsic features and behavior features and use the same predict models for evaluation (i.e., DeepFM for CTR prediction and DSSM for ad matching). We also compare our method with a state-of-the-art end-to-end behavioral targeting method, `NMTR` [17], a neural multi-task learning method, which predicts the target behavior from multiple types of behaviors.

Figure 9 reports the experimental results. We can see that `NMTR` performs the worst, because the multi-task learning method of `NMTR` relies heavily on the ordinal relations among the multiple types of behaviors. Just like the example in the original paper of `NMTR`, in the e-commerce platform, a user must view a product (i.e., clicking the product page) before she can purchase it. However, in the social media advertising platform, multiple types of behaviors generally do not exist such ordinal relations. The two baselines `ClassicalBT` and

TransferBT also achieve inferior performance and they perform worse than PartHIN (i.e., the size of human completion tasks for AcHC is 0), because they model the behaviors based on the TFIDF indexing by considering each user as a document and considering each item (i.e., ad, word, tag and official_account) as a term for user representation, in which the term frequency is an important feature for capturing user's interests, such as the frequency that a user browses a web page in web advertising. However, in the social media advertising platform, most of the user behaviors are one-off. For example, a user's behavior on following an official account will only occur once. By considering the timestamp of the user's behaviors, TemporalBT performs better than PartHIN. But our human-in-the-loop framework AcHC outperforms TemporalBT with only a few completion tasks. Moreover, as the number of completed triplets increases, the performance of AcHC is significantly better than all baseline methods. These results demonstrate the effectiveness of our human-in-the-loop approach to social media advertising.

F. Robustness against Human Errors

This section evaluates robustness of our framework again human errors. We first estimate human errors in our crowdsourcing platform deployed inside WeChat. We sample 5% completion tasks and manually provide the ground-truth. Then, we assign each of the tasks to 3 workers and obtain the result by majority voting. Finally, we find the average error rates of human after majority voting is 4.39%. This implies us to conduct the experiment by varying error rate from 0% to 8%. We evaluate the models, DeepFM and DSSM for CTR prediction and ad matching respectively. The experimental results are shown as the Figure 10. With varying the human error rates, the performance of our method does not change significantly. The main reason is that, as also verified by the existing works [8], when error rate is within a reasonable range, the effect of human mistakes on results is negligible.

G. Illustrative Case Study

In this section, we present a case study from WeChat, to interpret the Ads-HIN enhanced prediction. More specifically, we divide the ads in different industries and consider the CTR prediction task over these ads. We respectively evaluate the AUC scores of CTR prediction before and after using the HIN features mined from our Ads-HIN, and compute the relative improvement on AUC. Figure 11 reports the top industries with the highest AUC improvement because of the HIN features. We can see that our method achieves the highest improvement on industries including Game, Electronics, Software, and Entertainment. First, in comparison with other industries (e.g., Car), some industries such as Electronics and Entertainment contain a *diverse* set of ads in many subcategories. For example, the Electronics industry may contain ads of high-end products, such as HDTV and low-end ones, such as desk lamp. Thus, the intrinsic features about user profile and ad attributes are too "coarse-grained" to capture user's interest in the ads. In this case, our method can enhance the prediction by incorporating the fine-grained information mined from our Ads-HIN. Second, customers of the industries such as Game and Entertainment have shown more flexible and dynamic interest. Many game players are found to be always willing to try new games that are released. The intrinsic user features, such as age, gender, game playing history, could hardly describe their changing interest, nor can they predict their behavioral targeting results. Our method can capture the changing interest of the users by using their behavioral data, such as following official accounts or visiting game forums.

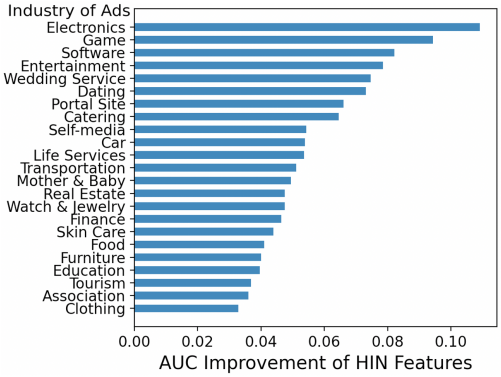


Fig. 10: An Illustrative Case Study.

VII. Related Work

Behavioral targeting in online advertising. Existing approaches to behavioral targeting can be divided into two categories. The classical methods [13], [14] segment users based on their online behaviors and then deliver the related ads to each user segment. More recently, some behavioral targeting methods [15]–[17], [33], [34] are proposed to model the user behavior sequences as the behavior vectors, which is used to directly predict the click behavior. [15], [33] explore impact of the change in user behavior as indication of interests in ad. [34] extends matrix factorization to jointly factorize multiple behavior matrices. [16] jointly captures observed multiple types of behaviors and unobserved clicks by a margin-based learning framework. [17] proposes a neural multi-task learning method to predict the target behavior from multiple types of behaviors. Our approach and these works focus on different aspects of behavioral targeting: they focus on specific user behavior modeling strategies, while we pay more attention to incorporate heterogeneous information of user behaviors and ad properties in an HIN to enhance the prediction model.

HIN enhanced recommendation. There are many works to leverage the HIN to improve the recommendation performance. Among them, some methods [23], [35]–[37] use the information from the HIN directly to enrich the representation of items or users by applying representation learning techniques to encode the HIN into low-rank embedding. These methods are generally called as embedding-based methods. [23], [35] use the HIN constructed with items and their related attributes, while [37] builds user-item HINs by introducing users into the network. Other methods [38]–[40] leverage the connectivity patterns of the objects in the HIN for recommendation, which

are generally called as path-based methods. Among them, [38] integrates matrix factorization with extracted meta-paths in HINs. [39], [40] use deep learning models to encode the path embedding explicitly. Our approach adopts the idea of embedding-based methods studied in the graph representation learning.

VIII. Conclusion

In this paper, we have studied the problem of human-in-the-loop social behavioral targeting. We introduced an ADS-HIN that models different relationships between users and ads from a variety of heterogeneous sources in an advertising platform in WeChat. Then, we extracted "high-quality" features from the HIN and fed the features to a prediction model for behavioral targeting. We devised a human-in-the-loop framework that judiciously leverages human intelligence to construct the ADS-HIN given budget on the number of human involvements. We conducted experiments on large-scale and real advertising datasets from WeChat to show the performance superiority of our approach.

References

[1] C. Shi, Y. Li, J. Zhang, Y. Sun, and S. Y. Philip, "A survey of heterogeneous information network analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 17–37, 2016.

[2] G. Li, J. Wang, Y. Zheng, and M. J. Franklin, "Crowdsourced data management: A survey," *IEEE TKDE*, vol. 28, no. 9, pp. 2296–2319, 2016.

[3] S. E. Whang, P. Lofgren, and H. Garcia-Molina, "Question selection for crowd entity resolution," in *PVLDB*. Stanford InfoLab, August 2013. [Online]. Available: http://ilpubs.stanford.edu:8090/1064/

[4] X. Chen, P. N. Bennett, K. Collins-Thompson, and E. Horvitz, "Pairwise ranking aggregation in a crowdsourced setting," in *WSDM*, 2013, pp. 193–202.

[5] Z. Xu, H. Zhang, C. Hu, L. Mei, J. Xuan, K.-K. R. Choo, V. Sugumaran, and Y. Zhu, "Building knowledge base of urban emergency events based on crowdsourcing of social media," *CCPE*, vol. 28, no. 15, pp. 4038–4052, 2016.

[6] A. B. McCoy, A. Wright, A. Laxmisan, M. J. Ottosen, J. A. McCoy, D. Butten, and D. F. Sittig, "Development and evaluation of a crowdsourcing methodology for knowledge base construction: identifying relationships between clinical problems and medications," *JAMIA*, vol. 19, no. 5, pp. 713–718, 2012.

[7] F. Daniel, P. Kucherbaev, C. Cappiello, B. Benatallah, and M. Allahbakhsh, "Quality control in crowdsourcing: A survey of quality attributes, assessment techniques, and assurance actions," *CSUR*, vol. 51, no. 1, pp. 1–40, 2018.

[8] Y. Tao, Y. Li, and G. Li, "Interactive graph search," in *SIGMOD*, 2019, pp. 1393–1410.

[9] S. Jia, Y. Xiang, X. Chen, and K. Wang, "Triple trustworthiness measurement for knowledge graph," in *WWW*, 2019, pp. 2865–2871.

[10] T. Nguyen, D. Phung *et al.*, "A relational memory-based embedding model for triple classification and search personalization," in *ACL*, 2020, pp. 3429–3435.

[11] X. Han, S. Cao, X. Lv, Y. Lin, Z. Liu, M. Sun, and J. Li, "Openke: An open toolkit for knowledge embedding," in *EMNLP 2018: system demonstrations*, 2018, pp. 139–144.

[12] J. Yang, X. Zhao, J. Fan, G. Chen, C. Peng, S. Yao, and X. Du, "A human-in-the-loop approach to social behavioral targeting," in *Technical Report*, 2020, https://github.com/hinsonver/tr/blob/master/hinbt.pdf.

[13] J. Yan, N. Liu, G. Wang, W. Zhang, Y. Jiang, and Z. Chen, "How much can behavioral targeting help online advertising?" in *WWW*, 2009, pp. 261–270.

[14] T. Chen, J. Yan, G. Xue, and Z. Chen, "Transfer learning for behavioral targeting," in *WWW*, 2010, pp. 1077–1078.

[15] B. Loni, R. Pagano, M. Larson, and A. Hanjalic, "Bayesian personalized ranking with multi-channel user feedback," in *RecSys*, 2016, pp. 361–364.

[16] J. Ding, G. Yu, X. He, Y. Quan, Y. Li, T.-S. Chua, D. Jin, and J. Yu, "Improving implicit recommender systems with view data." in *IJCAI*, 2018, pp. 3343–3349.

[17] C. Gao, X. He, D. Gan, X. Chen, F. Feng, Y. Li, T.-S. Chua, and D. Jin, "Neural multi-task recommendation from multi-behavior data," in *ICDE*. IEEE, 2019, pp. 1554–1557.

[18] S. Fan, Y. Lu, and S. Gupta, "Social media in-feed advertising: the impacts of consistency and sociability on ad avoidance," in *PACIS*, 2017, p. 190.

[19] C. Shi, Y. Ye, and J. Zhang, "Hena 2019: The 3rd workshop of heterogeneous information network analysis and applications," in *CIKM*. ACM, 2019, pp. 2991–2992.

[20] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[21] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *CIKM*, 2013, pp. 2333–2338.

[22] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *NIPS*, 2013, pp. 2787–2795.

[23] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," in *WWW*, 2018, pp. 1835–1844.

[24] H. Wang, Q. Meng, J. Fan, Y. Li, L. Cui, X. Zhao, C. Peng, G. Chen, and X. Du, "Social influence does matter: User action prediction for in-feed advertising," in *AAAI*, vol. 34, no. 01, 2020, pp. 246–253.

[25] Y. Zhu, J. Lin, S. He, B. Wang, Z. Guan, H. Liu, and D. Cai, "Addressing the item cold-start problem by attribute-driven active learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 4, pp. 631–644, 2019.

[26] N. Rubens, R. Tomioka, and M. Sugiyama, "Output divergence criterion for active learning in collaborative settings," *IPSJ Online Transactions*, vol. 2, pp. 240–249, 2009.

[27] Y. Guo and R. Greiner, "Optimistic active-learning using mutual information." in *IJCAI*, vol. 7, 2007, pp. 823–829.

[28] N. Murray and F. Perronnin, "Generalized max pooling," in *CVPR*, 2014, pp. 2473–2480.

[29] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *CIKM*, 2014, pp. 101–110.

[30] T. Huang, Z. Zhang, and J. Zhang, "Fibinet: combining feature importance and bilinear feature interaction for click-through rate prediction," in *RecSys*, 2019, pp. 169–177.

[31] Y. Yao, L. Rosasco, and A. Caponnetto, "On early stopping in gradient descent learning," *Constructive Approximation*, vol. 26, no. 2, pp. 289–315, 2007.

[32] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[33] M. Aly, S. Pandey, V. Josifovski, and K. Punera, "Towards a robust modeling of temporal interest change patterns for behavioral targeting," in *WWW*, 2013, pp. 71–82.

[34] Z. Zhao, Z. Cheng, L. Hong, and E. H. Chi, "Improving user topic interest profiles by behavior factorization," in *WWW*, 2015, pp. 1406–1416.

[35] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *SIGIR*, 2018, pp. 505–514.

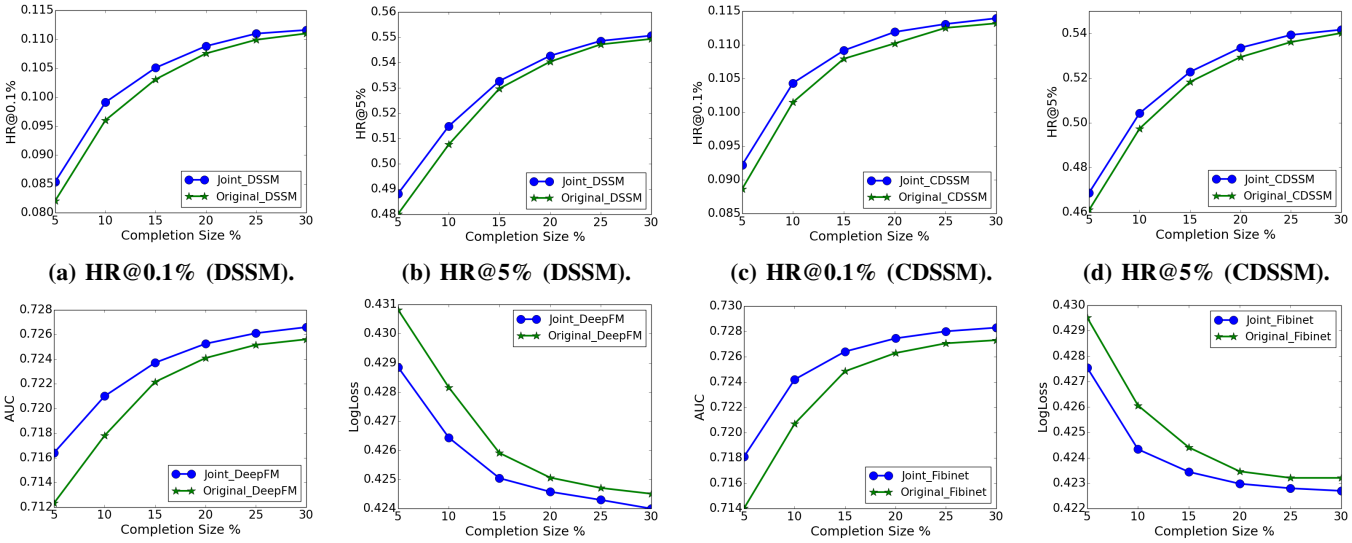
[36] Y. Zhang, Q. Ai, X. Chen, and P. Wang, "Learning over knowledge-base embeddings for recommendation," *arXiv preprint arXiv:1803.06540*, 2018.

[37] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, and Q. Liu, "Shine: Signed heterogeneous information network embedding for sentiment link prediction," in *WSDM*, 2018, pp. 592–600.

[38] C. Shi, B. Hu, W. X. Zhao, and S. Y. Philip, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 357–370, 2018.

[39] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *AAAI*, vol. 33, 2019, pp. 5329–5336.

[40] Y. Xian, Z. Fu, S. Muthukrishnan, G. De Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," in *SIGIR*, 2019, pp. 285–294.



(a) HR@0.1% (DSSM). **(b) HR@5% (DSSM).** **(c) HR@0.1% (CDSSM).** **(d) HR@5% (CDSSM).**

(e) AUC (DeepFM). **(f) LogLoss (DeepFM).** **(g) AUC (Fibinet).** **(h) LogLoss (Fibinet).**

Fig. 11: Evaluating Joint Model on Ad Matching and CTR Prediction Tasks.

IX. Appendix

A. The Schema of ADS-HIN

We construct a large-scale dataset based on data collected from the real social advertising platform in WeChat. The dataset consists of an ADS-HIN and the training sets for two representative behavioral targeting tasks. We build an ADS-HIN $\mathcal{G}$ based on the information from different views in WeChat to capture the potential interest of users on ads. More specifically, we consider the following 8 types of nodes. (1) `user`: the users of WeChat social platforms; (2) `ad`: the ads placed in WeChat Moments; (3) `official_account`: the media accounts on WeChat; (4) `advertiser`: the person or company that advertises; (5) `topic`: the topics of ads and official accounts, such as game and beauty; (6) `brand`: the brands related to ads and official accounts, such as NIKE and Sony; (7) `tag`: the user's interest tag refined based on the user's basic attributes and behaviors; (8) `word`: the keywords in the ad texts, images and videos.

Also, we consider the following 11 types of edges. (1) `click` : the relation from a user to an ad, meaning that the user has clicked the ad; (2) `interact`: the relation from a user to an ad, meaning that the user has commented or liked the ad; (3) `uninterest`: the relation from a user to an ad, meaning that the user has blocked or reported the ad; (4) `advertisting`: the relation from an advertiser to an ad, meaning that the advertiser has advertised the ad; (5) `follow`: the relation from a user to an official account, meaning the user has subscribed to the official account; (6) `interested_in_tag`: the relation from a user to a tag, meaning that the user is interested in the tag; (7) `interested_in_word`: the relation from a user to a word, meaning the user is interested in the keyword; (8) `mention`: the relation from an ad to a word, meaning that the word is a keyword mentioned in the ad; (9) `belong_to_topic`: the relation from an ad or official account to a topic, meaning that the ad or official account belongs to the topic; (10) `belong_to_brand`: the relation from an ad or official account to a brand, meaning that the ad or official account belongs to the brand; (11) `advertising_to_tag`: the relation from an ad to an interest tag, meaning that the ad is targeted advertising to the users interested in the interest tag.

B. Joint Classification Model

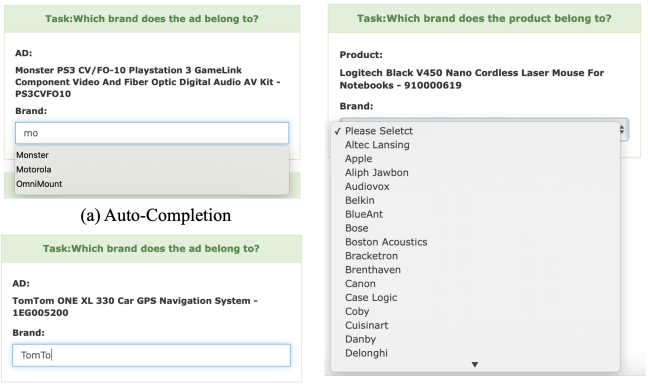
In this section, we introduce an improved classification method by utilizing *joint learning*. We unify the two model of graph representation learning and classification in a joint model to jointly learn the embeddings of users and ads. The benefit of the method is two-fold. On one hand, the prediction information of specific tasks is introduced into the graph representation learning model to guide the selection of triplet task. On the other hand, the graph structure information is introduced into the classification model to improve the performance of the classification model. Specifically, the joint model takes as input the intrinsic features of the users and ads, the data log and the ADS-HIN. We find the alignments between the user-ad instances and the nodes of users and ads in the ADS-HIN. The model jointly learns the BT task and graph representation task by enhancing the intrinsic features of user and ad in the classification model with their corresponding node embeddings in the graph representation model. As the shared parameters, the node embeddings of user and ad are updated by both models. We train the joint model using the overall objective function as follows:

$$L = \lambda L_C + (1 - \lambda) L_G, \tag{15}$$

where λ is a hyperparameter to balance the two tasks and L_C is the loss of the classification model.

We validate the effectiveness of the joint model by comparing it with the original classification model. The experimental results are shown in Figure **??**. We can see that the joint model outperforms all the four classifications original model. Especially when less than 15% of the tasks are completed, this advantage is more obvious. Because the joint model can improve the performance of task selection and classification model, and when the number of completed edges is limited, the advantage of sharing information by the joint training is more significant.

These results demonstrate the effectiveness of the joint model and we will take a more thorough study of the improved classification model in the future work.



(a) Auto-Completion

(b) Text_Box_Only

(c) Drop-down_Box_Only

Fig. 12: The Interface Examples.

TABLE III: Performance of Different Types of Interfaces.

Interface	Average completion time (ms)	Accuracy
$Auto-Completion$	2388.58	1.00
$Drop-down\_Box\_Only$	8871.38	1.00
$Text\_Box\_Only$	5485.28	0.875

C. User Study for Interface Generator

This section we provide a user study to shown the implementation and effectiveness of our interface generator. The study involves 16 users who are required to answers 150 brand completing tasks. In each task, we provide an ad, users need to tell us the brand of the ad. We divide the 150 brand completing tasks into 3 groups equally, each using different types of user interfaces. The first type is our $Auto-Completion$ interface, shown in Figure **??**(a), which automatically prompting and completing users inputs to help human easily find the desired answers. We compare our interface with two alternative user interfaces, $Drop-down\_Box\_Only$ and $Text\_Box\_Only$. Figure **??**(c) shows the example of $Drop-down\_Box\_Only$, which requires users to select a desired answer from a drop-down list of 84 brand candidates. Figure **??**(b) shows the example of $Text\_Box\_Only$, which requires users to freely type the answers they want in the text box without any prompts. First of all, we count the accuracy of users of the three groups of tasks. In addition, we use Python Flask framework with JS to count the time taken by users to complete each task. The result is shown as Table **??**. We can see that the average completion time of $Drop-down\_Box\_Only$ is the longest. Because the users need to pay large efforts to find the right answers from the large number of candidates. $Text\_Box\_Only$ achieves shorter completion time, its accuracy is very low. Because, without the limitation of candidate answers, users are more likely to make mistakes. $Auto-Completion$ outperforms $Drop-down\_Box\_Only$ and $Text\_Box\_Only$ on both accuracy and completion time, which shows its effectiveness.