Metapath-guided Heterogeneous Graph Neural Network for Intent Recommendation

Shaohua Fan
Beijing University of Posts and Telecommunications
Beijing, China
fanshaohua92@163.com

Junxiong Zhu
Alibaba Group
Hangzhou, China
xike.zjx@taobao.com

Xiaotian Han
Beijing University of Posts and Telecommunications
Beijing, China
hanxiaotian.h@gmail.com

Chuan Shi*, Linmei Hu
Beijing University of Posts and Telecommunications
Beijing, China
shichuan@bupt.edu.cn,
hulinmei1991@gmail.com

Biyu Ma
Alibaba Group
Hangzhou, China
biyu.mby@alibaba-inc.com

Yongliang Li
Alibaba Group
Hangzhou, China
anthonylee.liyl@tmall.com

ABSTRACT

With the prevalence of mobile e-commerce nowadays, a new type of recommendation services, called intent recommendation, is widely used in many mobile e-commerce Apps, such as Taobao and Amazon. Different from traditional query recommendation and item recommendation, intent recommendation is to automatically recommend user intent according to user historical behaviors without any input when users open the App. Intent recommendation becomes very popular in the past two years, because of revealing user latent intents and avoiding tedious input in mobile phones. Existing methods used in industry usually need laboring feature engineering. Moreover, they only utilize attribute and statistic information of users and queries, and fail to take full advantage of rich interaction information in intent recommendation, which may result in limited performances. In this paper, we propose to model the complex objects and rich interactions in intent recommendation as a Heterogeneous Information Network. Furthermore, we present a novel **M**etapath-guided **E**mbedding method for **I**ntent **Rec**ommendation (called **MEIRec**). In order to fully utilize rich structural information, we design a metapath-guided heterogeneous Graph Neural Network to learn the embeddings of objects in intent recommendation. In addition, in order to alleviate huge learning parameters in embeddings, we propose a uniform term embedding mechanism, in which embeddings of objects are made up with the same term embedding space. Offline experiments on real large-scale data show the superior performance of the proposed MEIRec, compared to representative methods. Moreover, the results of online experiments on Taobao e-commerce platform show that MEIRec not only gains a performance improvement of 1.54% on CTR metric, but also attracts up to 2.66% of new users to search queries.

KEYWORDS

Recommender Systems, Intent Recommendation, Heterogeneous Information Network, Graph Neural Network

ACM Reference Format:
Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi*, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided Heterogeneous Graph Neural Network for Intent Recommendation. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330673

1 INTRODUCTION

With the development of mobile Internet, the focus of e-commerce has moved from personal computers to smart phones, and various mobile e-commerce platforms have emerged. The benefits of recommendation systems are well recognized as a basic service of e-commerce platforms, which provide personalized recommendation sticking to user's preference. In the past two years, a novel recommendation service (named as intent recommendation in this paper), in many e-commerce Apps (e.g., Taobao and Amazon) have emerged, which automatically recommends user intent (presented as several words) in a search box according to users' historical behaviors when users open an e-commerce App. There are several reasons for the boom of intent recommendation in the era of mobile Internet. First, since typing words on mobile devices is more difficult than on desktop computers, intent recommendation can save user time without any input, which will raise user activity and stickiness. Second, users may have no apparent intent or do not know how to describe their intent, a personalized intent recommendation can help users find what they really need.

Figure 1 illustrates an intent recommendation example on the Taobao mobile App. According to user historic information, an intent (e.g., presented as "air jordan") will be automatically recommended in the search box when a user opens the App. If the user clicks the search button, he/she will jump to the corresponding

*Corresponding author

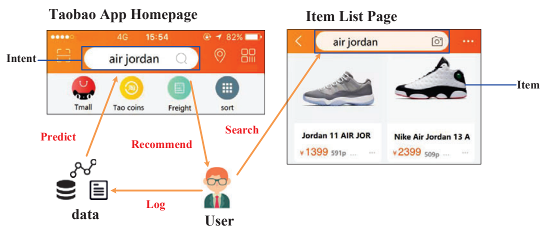
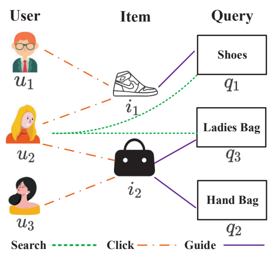
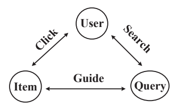


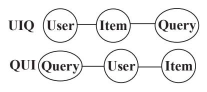
Figure 1: Intent recommendation example on Taobao mobile application.



(a) Toy example



(b) Network schema



(c) Metapaths

Figure 2: Heterogeneous information network and network schema.

item list page. In intent recommendation system, the historic information can be roughly categorized into two types. The first type is attribute data, containing attribute information of objects, such as user profiles and item attributes. The other type is interaction data, containing triple interaction among users, items, and queries, such as user click (item) logs, user search (query) logs, and query guide (item) logs.

In this paper, we define the intent recommendation as follows: automatically recommend a personalized intent for a user according to his/her historical behaviors without query input. Here, in our application scenario, intent is presented as query, consisting of several words or terms simply and directly reflecting user intent. However, intent recommendation is different from traditional query recommendation/suggestion [3, 14, 22] in the following two aspects. (1) It recommends queries according to user behaviors (i.e., interactions), rather than similar previous queries. (2) It also does not need users to input partial query. Also unlike previous study on mobile query recommendation [24] which considers only user-location-query relation, intent recommendation provides a flexible framework to consider complex interactions on heterogeneous objects in real systems. It is also different from item recommendation in several ways. (1) Our intent recommendation needs to consider the interactions among triple-objects (i.e., users, items and queries), rather than binary interactions between users and items in item recommendation. (2) Different from atomic and static items in item recommendation, intent (i.e., query constituted by words) always changes dynamically.

Existing methods for intent recommendation used in industry, such as Taobao and Amazon, usually extract handcrafted features, and then feed these features to a classifier, e.g., GBDT [7] and XGBoost [4]. These methods heavily rely on domain knowledge and need laboring feature engineering. They only utilize attribute and statistic information of users and queries, and fail to take full advantage of the rich interaction information among objects. However, the interaction information is very abundant in real systems, and it is really critical to capture user intent.

As a general information modeling method, Heterogeneous Information Network (HIN) [18], consisting of multiple types of objects and links, has been widely applied to many data mining tasks [10, 17, 18]. In this paper, we propose to model the intent recommendation system with a HIN, through which we can flexibly exploit its rich interaction information. As shown in Figure 2(a), obviously, HIN clearly demonstrates objects in intent recommendation (e.g., users, items and queries) and their interaction relations, such as "user click item", "user search query" and "query guide item". Although, some HIN based recommendation methods have been proposed [8, 19, 23], they mainly employ metapath based features through exploiting the interaction relations between users and items, which makes them hardly handle the triple-object interactions in intent recommendation.

In this paper, we present a novel **M**etapath-guided **E**mbedding method for **I**ntent **Rec**ommendation (called MEIRec). In order to fully utilize rich interaction information in intent recommendation, we propose to learn structural feature representations of users and queries with heterogeneous Graph Neural Network (GNN). Concretely, we present the metapath-guided neighbours to aggregate rich neighbour information, where different aggregation functions are designed according to the characteristics of different types of neighboring information. In addition, in order to handle large-scale data involved in intent recommendation, considering that both queries and titles of items consist of a limited number of terms, we design a uniform term embedding mechanism, in which embeddings of users and queries are made up with the same term embedding space. With the static features used in existing systems, as well as the embeddings of users and queries learned from interaction information, we build a prediction model for intent recommendation.

The major contributions of this paper are summarized as follows:

- We propose an important, but seldom exploited, intent recommendation problem, which automatically recommends a personalized intent according to user's historical behaviors.
- We present a novel MEIRec model with GNN. Through modeling intent recommendation system as a HIN, MEIRec utilizes metapath-guided neighbours to exploit rich interaction information in HIN. Moreover, a uniform term embedding mechanism is designed to greatly reduce the parameter space.
- Extensive offline experiments on a large-scale real data show that our MEIRec outperforms the representative baselines. We also conduct online experiments on Taobao e-commerce platform. The results show that our model significantly improves key metrics considered by the platform. Particularly, the platform attracts the 2.66% of new users to search the recommended query with the help of our method.

2 PRELIMINARIES

In this section, we define some basic concepts used in our model.

DEFINITION 1. **Intent Recommendation.** Given a set <$\mathcal{U}$, $\mathcal{I}$, $\mathcal{Q}$, $\mathcal{W}$, $\mathcal{A}$, $\mathcal{B}$>, where $\mathcal{U} = \{u_1, \cdots, u_p\}$ denotes the set of p users, $\mathcal{I} = \{i_1, \cdots, i_q\}$ denotes the set of q items, $\mathcal{Q} = \{q_1, \cdots, q_r\}$ denotes the set of r queries, $\mathcal{W} = \{w_1, \cdots, w_n\}$ denotes the set of n terms, $\mathcal{A}$ denotes the attributes associated with objects, and $\mathcal{B}$ denotes the interaction behaviors between different types of objects. In our application, a query $q \in Q$ or an item $i \in I$, is constituted by several terms $w \in \mathcal{W}$. The purpose of intent recommendation is to recommend the most related intent (i.e., query) $q \in \mathcal{Q}$ to a user $u \in \mathcal{U}$.

Taking Figure 1 for example, for a user $u \in \mathcal{U}$, when he refreshes the App, we can utilize information from $\mathcal{A}$ and $\mathcal{B}$ to calculate the preference score of u for a candidate query $q \in \mathcal{Q}$, and recommend the query with the highest score as user intent to the user u. It is worth noting that the recommended query reflects user intent through exploiting user historical interaction information. Moreover, the recommended query may be not previous queries, but new phrases generated by the combination of existing terms.

We model our recommendation task in the setting of **Heterogeneous Information Network** (HIN) [20]. A HIN is defined as a graph $G = (V, E)$, which has more than one node type or link type. In HIN, **network schema** is proposed to describe the meta structure of a network, which describes the object types and their interaction relations. The **metapath** [20], a relation sequence connecting two objects, is proposed to capture the structural and semantic relation between objects.

Figure 2(a) shows a toy example of HIN and Figure 2(b) is the corresponding network schema. We can see that the network consists of multiple types of objects (e.g, User (U), Item (I), Query (Q)) and their rich interaction relations. We are particularly interested in the metapaths that start from users and queries in our application, which can reveal semantic relations for users and queries. For example in Figure 2(c), "$User - Item - Query$ (UIQ)" metapath indicates user clicks items, and these items are guided by some queries. And "$Query - User - Item$ (QUI)" indicates a query is searched by some users, and these users also click some items recently.

DEFINITION 2. **Metapath-guided Neighbors**. Given an object o and a metapath ρ (start form o) in a HIN, the metapath-guided neighbors is defined as the set of all visited objects when the object o walks along the given metapath ρ. In addition, we denote the i-th step neighbors of object o as $\mathcal{N}_\rho^i(o)$. Specifically, $\mathcal{N}_\rho^0(o)$ is o itself.

Taking Figure 2(a) as an example, given the metapath "$User - Item - Query$ (UIQ)" and a user u_2, we can get metapath-guided neighbors as $\mathcal{N}_{\text{UIQ}}^1(u_2) = \{i_1, i_2\}$, $\mathcal{N}_{\text{UIQ}}^2(u_2) = \{q_1, q_2, q_3\}$. Then, all the metapath-guided neighbors of u_2 are denoted as $\mathcal{N}_{\text{UIQ}}(u_2) = \{\mathcal{N}_{\text{UIQ}}^0(u_2), \mathcal{N}_{\text{UIQ}}^1(u_2), \mathcal{N}_{\text{UIQ}}^2(u_2)\} = \{u_2, i_1, i_2, q_1, q_2, q_3\}$.

3 THE MEIREC MODEL

In this section, we present the proposed model **M**etapath-guided **E**mbedding for **I**ntent **Rec**ommendation (**MEIRec**).

3.1 Overview

The basic idea of the proposed model MEIRec is to design a heterogeneous GNN for enriching the representations of users and queries. With the help of HIN built from intent recommendation system, MEIRec leverages metapaths to guide the selection of different-step neighbors and designs a heterogeneous GNN to obtain the rich embeddings of users and queries. Moreover, we represent different types of objects with uniform term embedding for less parameters learning, since queries and titles of items are constituted by a small number of terms.

Figure 3 shows the overall framework of MEIRec. First, we use the triple-object HIN containing <user, item, query> as input. Second, we use the uniform term embedding to generate the initial embeddings of items and queries. Third, we aggregate the information of metapath-guided neighbors to learn the embeddings of users and queries via heterogeneous GNN. After that, we fuse the embeddings of users and queries based on different metapaths, respectively. Finally, with the fused embeddings of users and queries, accompanying with static features of users and queries, we predict the probability that a user will search a specific query. We illustrate these steps in detail in the following subsections.

3.2 Uniform Term Embedding

In previous neural-network based recommendation, every user or query should have an unique embedding. In the intent recommendation scenario, there are billions of users and queries. If we employ traditional collaborative filtering or neural-network based methods to represent all users and queries, it will make the number of parameters tremendous. Note that queries and titles of items are constituted by terms and the number of terms is not many. So we propose to represent the queries and items with a small number of term embeddings. And thus we only need to learn the term embeddings, rather than all object embeddings. This method is able to significantly reduce the number of parameters.

Specifically, we extract terms from the queries and items' titles , and build a term lexicon $\mathcal{W} = \{w_1, w_2, \cdots, w_{n-1}, w_n\}$. Note that queries and items (i.e., their titles) are the combination of several terms. For example, as shown in Figure 3(a)-(b), query "Hand Bag" is constituted by terms "Hand" and "Bag", and item "LV Hand Bag" is constituted by terms "LV", "Hand" and "Bag". Since the number of the lexicon $\mathcal{W}$ is far less than the number of the queries and users, the uniform term embedding can significantly reduce the number of learned parameters. More importantly, the new queries that have never been searched before can be represented by these terms.

We will illustrate how the uniform term embedding works with two examples query q_2 and item i_2 shown in Figure 3(b), the query q_2 is constituted by the term set $\{w_1, w_n\}$ and the item i_2 has the term set $\{w_1, w_{n-1}, w_n\}$. We use multi-hot encoding to represent the query q_2 and item i_2 as following:

$$\begin{array}{rllllll} & \{w_1, & w_2, & \cdots, & w_{n-1}, & w_n\} \\ q_2 = & (1, & 0, & \cdots, & 0, & 1) \\ i_2 = & (1, & 0, & \cdots, & 1, & 1). \end{array} \tag{1}$$

Terms are important words or phrases. We use the AliWS (Alibaba Word Segmenter) to segment the queries and items' titles and select important words or phrases which contains rich meanings

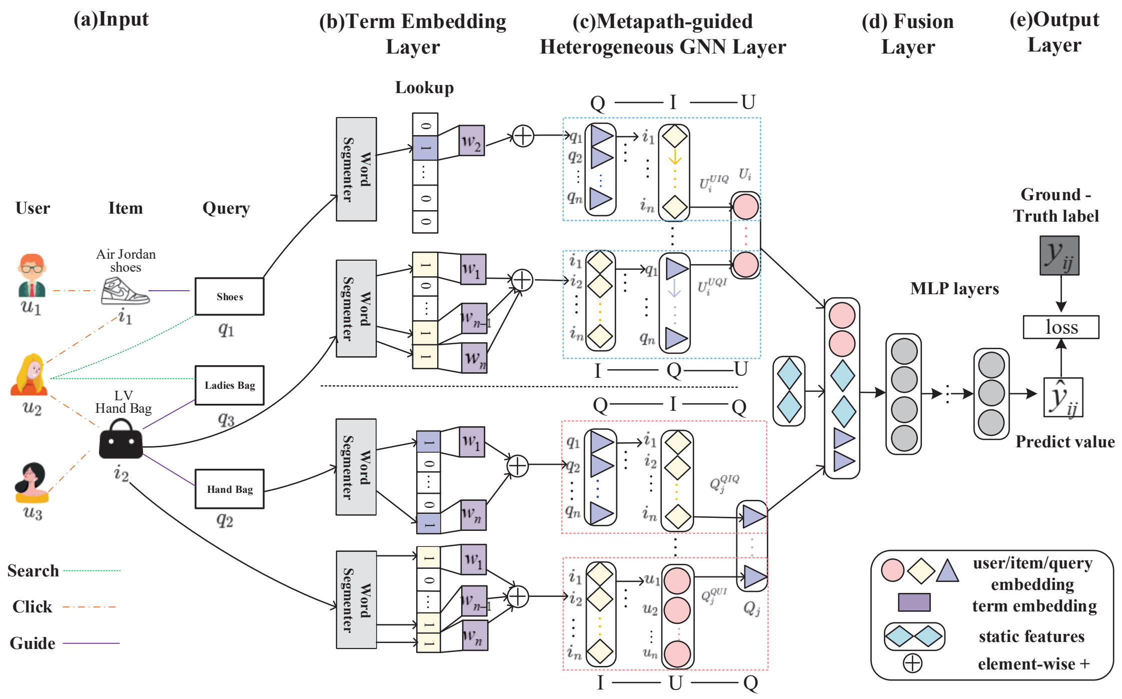


Figure 3: The framework of MEIRec.

A term embedding $f : M \rightarrow \mathcal{R}^d$, where M represents the dictionary of term words, is a parameterized function mapping all the terms to d-dimensional distributional vectors. In the look-up layer, the queries or items are represented as combination of term embeddings to extract their semantic information. Then we aggregate the term embeddings to get the embeddings of queries or items as following:

$$E_{q_2} = g(\boldsymbol{e}_{w_1}, \boldsymbol{e}_{w_n}), E_{i_2} = g(\boldsymbol{e}_{w_1}, \boldsymbol{e}_{w_{n-1}}, \boldsymbol{e}_{w_n}), \tag{2}$$

where e_{w_i} is the embedding of term w_i and the $g(\cdot)$ means the operation function applied to the terms. In our experiments, we adopt the average function.

By using the terms from the same lexicon, we obtain the embeddings of queries and items in a uniform term embedding space. Thus, term embeddings can be optimized by the embeddings of all objects simultaneously. This leads to that term embeddings contain the user-query and user-item interaction information in the embedding layer. Moreover, we only need to learn term embeddings with a small size (it is about 280000, compared to near ten millions of objects, in our experiments), which significantly reduces the complexity of our model.

3.3 Metapath-guided Heterogeneous Graph Neural Network

Inspired by the basic idea of the GCNs which generates object embeddings based on local neighbors [12, 21], we first propose a metapath-guided heterogeneous GNN. That is, we leverage metapaths to obtain different-step neighbors of an object, and the embeddings of users and queries are the aggregation of their neighbors under different metapaths.

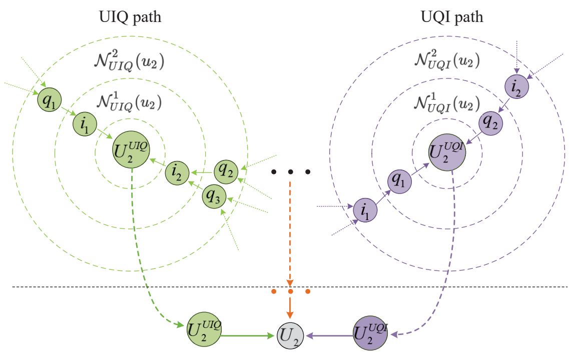


Figure 4: A toy example of metapath-guided information aggregation. Objects in this example are from Figure 2.

We present a toy example in Figure 4 to illustrate this process. Here we describe how to obtain the embedding U_2 of user u_2 based on multiple metapaths, such as UIQ and UQI. We first illustrate how we aggregate neighbor information along path UIQ. We use the uniform term embedding to obtain the initial embeddings of queries. And then we aggregate the metapath-guided neighbors to get the metapath-guided embedding of user u_2. According to the network structure in Figure 2(a), we get the 1-st step neighbors set of u_2 , $\mathcal{N}^1_{\text{UIQ}}(u_2) = \{i_1, i_2\}$. For each node i_k in the neighbors set $\mathcal{N}^1_{\text{UIQ}}(u_2)$, we extract the 2-nd step neighbor set $\mathcal{N}^2_{\text{UIQ}}(u_2) = \{q_1, q_2, q_3\}$. After we obtain the 1-st step and 2-nd step neighbors set of u_2, we aggregate the embeddings of 2-nd step neighbors to obtain the 1-st step neighbors' embeddings. In this example, we aggregate the embedding of query q_1 to obtain the item i_1's embedding, and aggregate the embeddings of queries q_2 and q_3 to obtain the item i_2's embedding. Finally, we aggregate the embeddings of 1-st step neighbors $\{i_1, i_2\}$ to obtain embedding $\boldsymbol{U}^{\text{UIQ}}_2$ of user u_2. Following this process, we can get different metapath-guided embeddings of u_2, such as $\boldsymbol{U}^{\text{UQI}}_2$. Then we aggregate all the metapath-guided embeddings to get final embedding of u_2 (i.e., $\boldsymbol{U}_2$).

3.4 User Modeling

In our model, we aggregate the information of different-step neighbors to obtain the representation $\boldsymbol{U}_i$ of user u_i via metapath-guided heterogeneous GNN. In this subsection, we show the how MEIRec models user embedding in detail.

As shown in the upper box in Figure 3(c), in order to get the embedding $\boldsymbol{U}_i$ of user u_i, we select metapaths starting from target user. We first search different-step neighbors along the metapath, and then aggregate the embeddings of neighbors step by step. Taking the metapath UIQ (meaning user clicks the items which had been guided by queries) for example, we can obtain different-step neighbors of a user u_i. After we get the 1-st step and 2-nd step neighbors set, we aggregate the embeddings of 2-nd step neighbors (query) to obtain the 1-st step neighbors' (item) embeddings and the embedding I^{UIQ}_j of item i_j in $\mathcal{N}^1_{\text{UIQ}}(u_i)$ based on the metapath UIQ is:

$$\boldsymbol{I}^{\text{UIQ}}_j = g(\boldsymbol{E}_{q_1}, \boldsymbol{E}_{q_2}, \cdots), \tag{3}$$

where $g(\cdot)$ is the aggregation function. According to characteristics of neighbors, we design different functions and in our model we adopt the average function which gains an impressive performance in our experiments. And the queries $\{q_1, q_2, \cdots\}$ are the neighbors of item i_j.

Next, we aggregate 1-st step neighbors' (item) embeddings to obtain the embedding $\boldsymbol{U}^{\text{UIQ}}_i$ of user u_i:

$$\boldsymbol{U}^{\text{UIQ}}_i = g(\boldsymbol{I}^{\text{UIQ}}_1, \boldsymbol{I}^{\text{UIQ}}_2, \cdots), \tag{4}$$

where the items $\{i_1, i_2, \cdots\}$ are the neighbors of user u_i. Since users click queries or items with timestamp, we model the neighbors of users (i.e., items or queries) as a sequence data. Recurrent Neural Network (RNN), especially the Long Short Term Memory (LSTM) [2, 5] has been proved to perform well for sequential data. So MEIRec leverages the LSTM to dynamically model the neighbors of users. That is, the aggregation function $g(\cdot)$ is LSTM for the neighbors of users.

Then we obtain the fused user embedding by aggregating embeddings based on different metapaths $\{\rho_1, \rho_2, \cdots, \rho_k\}$:

$$\boldsymbol{U}_i = g(\boldsymbol{U}^{\rho_1}_i, \boldsymbol{U}^{\rho_2}_i, \cdots, \boldsymbol{U}^{\rho_k}_i), \tag{5}$$

where the ρ is metapath starting from user.

3.5 Query Modeling

Similar to user information aggregation, we also obtain the fused query embedding $\boldsymbol{Q}_i$ based on metapaths $\{\rho_1, \rho_2, \cdots, \rho_k\}$:

$$\boldsymbol{Q}_i = g(\boldsymbol{Q}^{\rho_1}_i, \boldsymbol{Q}^{\rho_2}_i, \cdots, \boldsymbol{Q}^{\rho_k}_i), \tag{6}$$

where the ρ is the metapath starting from query.

Note that the neighbors of queries (i.e., items and users) are not presented in the time order. So in our model, we leverage the Convolutional Neural Network (CNN) to dynamically model the neighbors of queries. That is, the aggregation function $g(\cdot)$ is CNN for the neighbors of queries.

3.6 Optimization Objective

In our model, we predict the probability $\hat{y}_{ij}$ of user u_i search the query q_j which is in the range of [0,1] to ensure that the output value is a probability. Through aggregating the neighbors of user and query, we obtain the fused user embedding $\boldsymbol{U}_i$ for user u_i and fused query embedding $\boldsymbol{Q}_j$ for query q_j. In addition, there are raw static features used in traditional methods, include attributes of users (queries) and static features from interaction information. We feed these static features to a Multi-Layer Perceptron for obtaining the representation of the static features $\boldsymbol{S}_{ij}$. Then, we concatenate the embeddings of user, query and static features to fuse them. Finally, we feed the fused embeddings into MLP layers to get the predict score $\hat{y}_{ij}$. Then we have:

$$\hat{y}_{ij} = sigmoid(f(\boldsymbol{U}_i \oplus \boldsymbol{Q}_j \oplus \boldsymbol{S}_{ij})), \tag{7}$$

where the $f(\cdot)$ is the MLP layers with only one output, $sigmoid(\cdot)$ is the sigmoid layer, and $\oplus$ is the embedding concatenate operation.

The loss function of our model is a point-wise loss function in Equation 8.

$$J = \sum_{i,j \in \mathcal{Y} \cup \mathcal{Y}^-} \left(y_{ij} log \hat{y}_{ij} + (1 - y_{ij}) log(1 - \hat{y}_{ij}) \right), \tag{8}$$

where y_{ij} is the label of the instance (i.e. 1 or 0) and the $\mathcal{Y}$ and the $\mathcal{Y}^-$ are the positive and negative instances set, respectively.

3.7 Model Analysis

Here we analyze the parameter space complexity of MEIRec. For simplicity, we assume that there are M (billion-level) objects and N (100K-level) terms. In addition, suppose that there are h hidden layers with n neurons in our model and we set the dimension of the embedding to d. We compare the parameter space complexity between traditional latent factor and MEIRec. We first analyze the parameter space complexity of the traditional latent factor model which learns the embeddings of users and queries. The parameters to learn consists of two parts: embedding matrix and weight matrix of the hidden layer. The parameter space complexity of embedding matrix is $O(d * M)$, while the weight matrix of the hidden layer is $O(n * h)$. So the whole parameter space complexity is $O(d * M + n * h)$. In the real applications, because of the fact that $d * M \gg n * h$, the parameter space complexity of traditional methods is $O(d * M + n * h) \approx O(d * M)$. However, it is not the case for our MEIRec. MEIRec uses the uniform term embeddings instead of embeddings of the users and items in the embedding layer, so the parameter space complexity is $O(d * N + n * h) \approx O(d * N)$. Because $N \ll M$, the parameter space complexity of MEIRec is

Table 1: The statistics of the datasets.

Dataset	1-day	3-day	5-day
Training size (positive)	2,000,000	6,000,000	9,999,999
Training size (all)	8,000,000	23,999,998	39,999,997
Validation size (positive)	2,000,000	2,000,000	1,949,143
Validation size (all)	7,999,997	8,000,000	7,949,142
Train users	4,792,621	11,489,531	16,419,735
Train queries	871,133	1,653,865	2,163,574
Validation users	4,819,489	4,809,497	4,790,912
Validation queries	876,636	859,488	787,672
New users in validation set	3,666,692	2,613,695	2,064,564
Density	4.8×10^{-7}	3.1×10^{-7}	2.8×10^{-7}

much smaller than traditional methods. The smaller parameter space complexity of MEIRec means small computational space and high learning efficiency, which makes it suitable for large-scale data for real applications.

4 OFFLINE EXPERIMENTS

4.1 Dataset

We collect a real-world large-scale dataset from Taobao mobile application from Android and IOS online. We first extract 42 static features for user, including gender, age, purchasing power, etc and 39 static features for query, including length, term size, CTR, etc. And we construct a HIN based on interaction data collected during 10 days, consisting of about 100 million queries, 400 million users and 400 million items. In addition, the HIN contains about 4 billion search relations between users and queries, 20 billion click relations between users and items, and 4 billion guide relations between items and queries. The network constitutes structural information for the training and validation samples.

Next, we introduce how to construct training and validation samples. We utilize the interaction data during 5 days. Specifically, each raw interaction record in the collected dataset contains <user, recommended query, timestamp, label> representing that the recommended query has been shown to user at timestamp. And the label indicates whether the user clicks the recommended query. To better understand the performance of our proposed model, we validate our model on different scales of data. In our offline experiments, we use training data for different time periods (from 1 to 5 days) to predict the next one-day. Therefore, we have three datasets with different scales marked as **1-day**, **3-day** and **5-day**. To get more robust results, we vary the size of the each training set from 40% to 100%. The detailed statistics of the data are shown in Table 1. Besides, in order to get term lexicon, we first use the AliWS to segment the context of queries and titles of items to obtain a term lexicon, and select 280,000 terms to meet the needs of Alibaba e-commerce.

Our datasets have the following unique characteristics. First, the datasets are large enough, and contain millions of users and queries in both of training and validation set. Second, our datasets contain about half to three quarters new users in validation set. Third, as the density (i.e., (#interactions of users and queries)/(#users∗#queries)) shown in Table 1, the datesets are extremely sparse. These characteristics of data bring great challenges to our model design.

4.2 Baselines and Evaluation Metrics

To validate the effectiveness of our proposed model, we use the popular models used in industry (e.g., LR, DNN, and GBDT) with different feature settings and a popular neural network based model NeuMF. Note that those query recommendation models are not included because of not suitable for our problem setting, and some recent fancy models are also excluded due to not handling large-scale data.

- **LR** [15]: It is a linear model with static features.
- **DNN**: With the same input setting as LR, we implement the deep neural network with 3 layers MLP.
- **GBDT** [7]: It is a scalable tree-based model for feature learning and classification task. We feed static features into GBDT.
- **LR/DNN/GBDT+ DW**: We feed the static features of users and queries, as well as the pre-training embeddings learned by DeepWalk (DW) [16] from structural information, into LR/DNN/GBDT model.
- **LR/DNN/GBDT+ MP**: We feed the static features of users and queries, as well as the pre-training embeddings learned by MetaPath2vec (MP) [6] from structural information, into LR/DNN/GBDT model.
- **NeuMF [9]**: It is the state-of-art neural network method for top-N recommendation. Here we feed it with the structural information (interactions between users and queries), since it cannot be fed the static features.
- **MEIRec**: It's our model with the input of the static features and structural information.

In our experiments, we use **A**rea **U**nder receiver operator characteristic **C**urve (AUC) [13] to evaluate the performance of different models for comparison. The large AUC value means better performance.

4.3 Detailed Implementation

We implement the proposed method based on Tensorflow [1]. For our method, we set the dimension of term embedding as 64. We use a single-layer LSTM with 64 hidden neurons to model the user-query-sequence and user-item-sequence and use a single-layer CNN to aggregate queries' neighbors. For GBDT, the tree number is set as 200. For Deepwalk/MetaPath2vec, the dimension of embeddings is set as 32. For all the methods, in the training stage, we randomly initialize the model parameters with a Gaussian distribution, and optimize the model with mini-batch Adam [11]. We set the batch size as 512, and set the learning rate as 0.001. All the experiments are performed in Nvidia Tesla P100 Cluster.

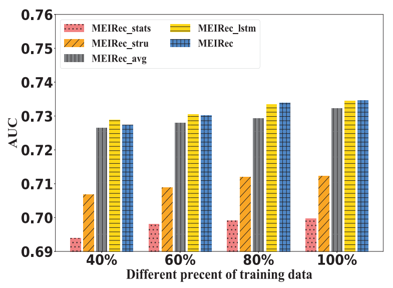
4.4 Performance Evaluation

The performances of MEIRec and the baselines are reported in Table 2. The major findings from the experimental results can be summarized as follows:

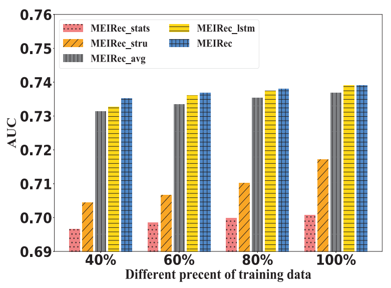
(1) MEIRec significantly outperforms all the compared baselines. Compared to the best performance of baselines (i.e., GBDT + MP or GBDT + DW, indicated with "*" at Table 2), MEIRec offers an improvement of 2.1%~4.3% in the three datasets. The results show that MEIRec achieves best results by using both static and structural features. It indicates that our model adopts a more principled way to

Table 2: The AUC comparisons of different methods. The * indicates the best performance of the baselines. Best results of all methods are indicated in bold. The last row indicates the percentage of improvements gained by the proposed method compared to the best baseline.

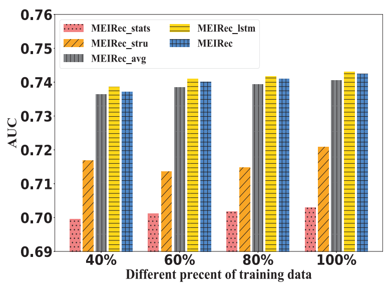
Method	1-day				3-day				5-day			
	40%	60%	80%	100%	40%	60%	80%	100%	40%	60%	80%	100%
NeuMF	0.6014	0.6066	0.6136	0.6143	0.6168	0.6218	0.6249	0.6291	0.6172	0.6224	0.6246	0.6295
LR	0.6854	0.6838	0.6884	0.6889	0.6844	0.6863	0.6857	0.6865	0.6817	0.6831	0.6827	0.6836
LR+DW	0.6878	0.6904	0.6898	0.6930	0.6888	0.6896	0.6898	0.6900	0.6838	0.6842	0.6863	0.6867
LR+MP	0.6918	0.6936	0.6950	0.6969	0.6919	0.6930	0.6933	0.6933	0.6874	0.6890	0.6898	0.6899
DNN	0.6939	0.6981	0.6991	0.6997	0.6966	0.6985	0.6999	0.7008	0.6996	0.7011	0.7017	0.7029
DNN+DW	0.6962	0.6980	0.7003	0.7024	0.7005	0.7017	0.7024	0.7030	0.7017	0.7029	0.7040	0.7047
DNN+MP	0.6984	0.6992	0.7024	0.7057	0.7025	0.7040	0.7051	0.7057	0.7017	0.7044	0.7060	0.7069
GBDT	0.7071	0.7071	0.7067	0.7073	0.7070	0.7071	0.7072	0.7071	0.7067	0.7068	0.7072	0.7066
GBDT+DW	0.7114	0.7119	0.7112*	0.7118*	0.7109	0.7106	0.7106	0.7104	0.7109	0.7112	0.7109	0.7114
GBDT+MP	0.7122*	0.7127*	0.7110	0.7111	0.7123*	0.7122*	0.7122*	0.7124*	0.7118*	0.7114*	0.7114*	0.7120*
MEIRec	**0.7273**	**0.7302**	**0.7339**	**0.7346**	**0.7352**	**0.7369**	**0.7380**	**0.7390**	**0.7372**	**0.7401**	**0.7409**	**0.7425**
Improvement	2.1%	2.5%	3.2%	3.2%	3.2%	3.5%	3.6%	3.7%	3.6%	4.0%	4.1%	4.3%



(a) 1-day

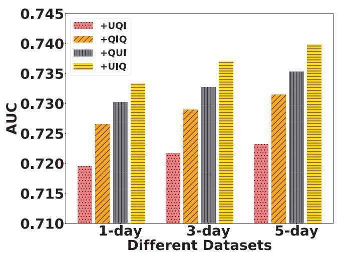


(b) 3-day

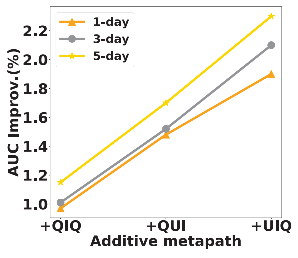


(c) 5-day

Figure 5: The AUC comparisons of MEIRec with different aggregation strategies.



(a) The AUC performances of additive metapths.

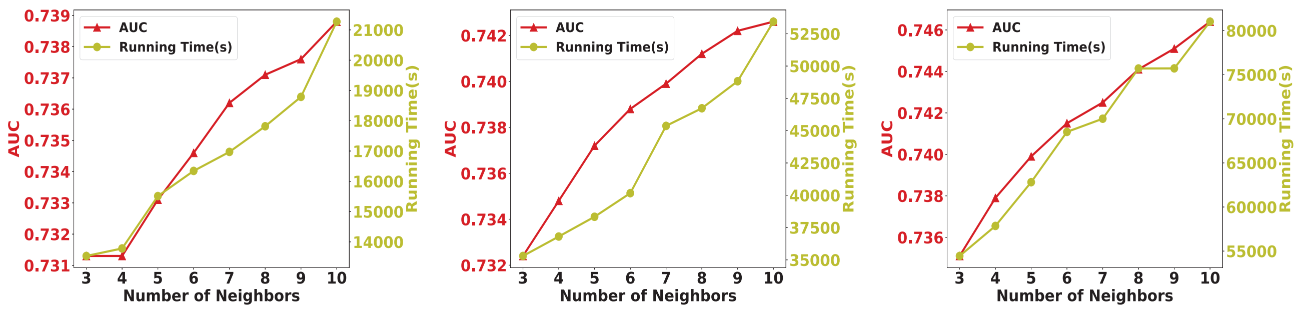


(b) The AUC improvements of additive metapths.

Figure 6: Performances of MEIRec with additive metapths.

leverage static features and interaction relations for improving prediction performance. And compared with pre-training embedding methods (i.e., LR/DNN/GBDT+ DW and LR/DNN/GBDT+ MP), our model learns the embeddings in a task-guided (i.e., classification objective) way, which is more effective to learn embeddings for the query task intent recommendation.

(2) Among these baselines, we find that the order of overall performances is as follows: at method level, GBDT > DNN > LR > NeuMF. Due to that NeuMF cannot learn the embeddings of new users and new queries appeared in the validation set, new objects' embeddings will be random variables, which makes the worst performances of NeuMF. For this problem, GBDT is a good classification model to fuse feature information, which makes it achieve good performances. That is the reason why it is widely used in real systems. And at feature level, (static features + heterogeneous embeddings) based methods > (static features + homogeneous embeddings) based methods > static features based methods. This ranking indicates that fusing more information could usually get better performances. And we can also find that, using heterogeneous network embedding (i.e., MetaPath2vec) can get better performances than homogeneous network embedding (i.e., Deepwalk). It demonstrates that we should consider heterogeneity of objects in HIN for better performances. At both levels, we conclude that choosing a model plays a key role in intent recommendation, and adopting appropriate methods to fuse more information could significantly improve the performance. As a consequence, the MEIRec



(a) 1-day (b) 3-day (c) 5-day

Figure 7: Performances of MEIRec with different number of neighbors.

achieves best performances, due to the heterogeneous GNN model and utilization of rich heterogeneous interactions.

(3) As the scale of data increasing, our model outperforms the best baselines with an increased margin (from 2.1% to 4.3%). The result further confirms that our model is more scalable for large-scale datasets.

4.5 Effect of Aggregation Methods

In our model, we enrich the information of users and queries by capturing their neighbor information along the given metapath. In order to explore the effect of different neighbor aggregation functions, we design different variants of MEIRec as follows.

- **MEIRec**$_{stats}$: It only uses the static features.
- **MEIRec**$_{stru}$: It only uses the structural information.
- **MEIRec**$_{avg}$: Both structural information and static features are used. We use the AVE function (i.e., average operation on aggregated embeddings) to aggregate the neighbors of both users and queries in this model.
- **MEIRec**$_{lstm}$: It uses the structural information and static features. We use LSTM to aggregate the neighbors of users and use AVE to aggregate the neighbors of queries in this model.
- **MEIRec**: It is the proposed model MEIRec.

The AUC comparisons of our proposed method with different aggregation methods are shown in Figure 5. We can see that the methods which only use static or structural information (i.e., MEIRec$_{stats}$, MEIRec$_{stru}$) get worse results than the methods using both information. MEIRec$_{stru}$ outperforms MEIRec$_{stats}$ (same as DNN), indicating structural information is more powerful than static features for this task, and our structural fusing strategy is effective. The performance of MEIRec$_{stats}$ also indicates that static information have limited effect on the results of MEIRec. Moreover, the results of MEIRec$_{avg}$ confirm that use more information of both is significantly better than the methods only using one single kind of features. On the other hand, the model MEIRec$_{lstm}$ and MEIRec gain more improvements than MEIRec$_{avg}$, indicating that it is necessary to leverage different aggregation functions for different types of neighbors. In our model, for user side, the LSTM function capture time-sequence information for user behaviors, such as user click item sequence and user search query sequence. And for query side, the unordered functions (i.e., CNN or AVG) are good enough to aggregate the neighbor information of query.

4.6 Effect of Different Metapaths

In our model, we aggregate different types of neighbors guided by metapaths to improve the recommendation performance. To further investigate the effect of different metapaths on learned embeddings for the intent recommendation task, we observe the performance of MEIRec through adding four informative metapaths one by one. The four metapaths are UQI, QIQ, QUI, and UIQ, and they are added into the model by their order.

Figure 6(a) is the AUC performance of MEIRec with additive metapaths. The results shown in three datasets demonstrate the performance of our proposed model stably increases as we add metapaths one by one. Moreover, Figure 6(b) is the AUC improvements of the variants (i.e., +QIQ, +QUI, +UIQ) against the model only using the first metapath (i.e., +UQI). One can also see that the performance of MEIRec with more metapaths gradually increases as the scale of data increases. The results indicate that when we deal with large-scale data, MEIRec with more metapaths usually yields better performances. This gain demonstrates that adding new informative metapaths plays an important role in learning task-related embeddings. Note that we only employ four representative metapaths in experiments due to the limitation of experimental settings. However, MEIRec provides a flexible framework to utilize more metapaths and integrate more heterogeneous interactions.

4.7 Effect of the Number of Neighbors

In this subsection, we conduct a series of experiments on three datasets to evaluate the effect of the number of neighbors on performance. Specifically, for query side, we set the number of neighbors as a fixed value 5, and for user side, we vary the number of neighbors from 3 to 10.

As illustrated in Figure 7, for different number of neighbors, the red line represents the AUC performance, and the yellow line indicates the running time. One can see that the performance of our model steadily improves as the number of neighbors grows. Note that, with the increasing neighbors, the performances of MEIRec still increase, but tend to be steady. We only set the maximum number of neighbors as 10, due to the limitation of computation resource. This conforms that the information of neighbors can effectively enhance the representations of users. It also demonstrates that the more local neighbor information is more effective in modeling the user's personalized intent. However, we also notice that

Table 3: Online A/B testing experiments results.

Data	Methods	CTR	Unique Click	UCTR
Android	GBDT	1.746%	256,116	13.939%
	MEIRec	1.758%	260,634	14.229%
	Improvement	0.70%	1.76%	2.07%
IOS	GBDT	0.7687%	62,462	5.2579%
	MEIRec	0.8056%	65,895	5.5436%
	Improvement	4.79%	5.50%	5.43%
Total	GBDT	1.4035%	318,578	10.5252%
	MEIRec	1.4252%	326,529	10.8052%
	Improvement	1.54%	2.50%	2.66%

as the number of neighbors grows, the running time of the model also increases. Therefore, in order to tradeoff between accuracy and running time, we usually set the number of neighbors as 5.

5 ONLINE EXPERIMENTS

To furtherly evaluate the proposed model, we conduct online experiments in Taobao mobile App. We conduct a bucket testing (i.e., A/B testing) online to test the users' response to our model against baseline. We select one bucket for baseline, and another bucket for our model. And we select the GBDT model for comparison for that GBDT is used in real system. We use the metric **CTR**, **Unique Click**, and **UCTR** to evaluate the online performance, where **CTR** and **UCTR**=**Unique Click**/**Unique Visitor** indicate change of the click ratio and visit ratio.

The results are shown in Table 3. We can see that, compared to the GBDT, MEIRec achieves performance improvement in all metrics, which indicates that incorporating interaction information can better capture user latent intent. Our model gains the improvement of 0.70%, 4.79% and 1.54% for Android, IOS and Total respectively in CTR. Since the CTR is to measure the ratio of clicks against impressions, the improvement of CTR shows that our model can greatly improve the user's search experience. In addition, the metric UCTR indicates how many unique visitors click the recommended query, and it gains an improvement of 2.07%, 5.43% and 2.66% for Android, IOS and Total. The improvement of UCTR shows that our model have an advantage in attracting new users to search queries.

6 CONCLUSION

In this paper, we study the intent recommendation problem which plays an important role in increasing user activity and stickiness in mobile e-commerce. In order to solve the challenges in intent recommendation, we model objects and interactions in intent recommendation system with a HIN and propose a novel metapath-guided GNN method for intent recommendation, called MEIRec. MEIRec utilizes metapath-guided neighbours to exploit rich structural information in HIN. Moreover, a uniform term embedding is designed in MEIRec, which not only significantly reduces parameter space, but also makes it suitable for new generated users and queries. The extensive results on offline and online experiments demonstrate the effectiveness of our proposed model.

The number of visitors who performed a click

ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China (No. 61772082, 61806020, 61702296), the National Key Research and Development Program of China (2017YFB0803304), the Beijing Municipal Natural Science Foundation (4182043), the Fundamental Research Funds for the Central Universities, and the BUPT Excellent Ph.D. Students Foundation (No. CX2019127).

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.
[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*. 651–665.
[3] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *SIGKDD*. ACM, 875–883.
[4] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *SIGKDD*. 785–794.
[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. [n. d.]. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*. 1724–1734.
[6] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *SIGKDD*. 135–144.
[7] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.
[8] Xiaotian Han, Chuan Shi, Senzhang Wang, S Yu Philip, and Li Song. 2018. Aspect-Level Deep Collaborative Filtering via Heterogeneous Information Networks.. In *IJCAI*. 3393–3399.
[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.
[10] Binbin Hu, Zhiqiang Zhang, Chuan Shi, Jun Zhou, Xiaolong Li, and Yuan Qi. 2019. Cash-out User Detection based on Attributed Heterogeneous Information Network with a Hierarchical Attention Mechanism. In *AAAI*.
[11] Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
[12] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.
[13] Jorge M Lobo, Alberto Jiménez-Valverde, and Raimundo Real. 2008. AUC: a misleading measure of the performance of predictive distribution models. *Global ecology and Biogeography* 17, 2 (2008), 145–151.
[14] Patrick Marcel and Elsa Negre. 2011. A survey of query recommendation techniques for data warehouse exploration.. In *EDA*. 119–134.
[15] Andrew Y Ng and Michael I Jordan. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NIPS*. 841–848.
[16] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*. 701–710.
[17] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S Yu Philip. 2019. Heterogeneous information network embedding for recommendation. *TKDE* 31, 2 (2019), 357–370.
[18] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and P. S. Yu. 2017. A survey of heterogeneous information network analysis. *TKDE* 29, 1 (2017), 17–37.
[19] Chuan Shi, Zhiqiang Zhang, Ping Luo, P. S. Yu, Yading Yue, and Bin Wu. 2015. Semantic path based personalized recommendation on weighted heterogeneous information networks. In *CIKM*. 453–462.
[20] Yizhou Sun, Jiawei Han, Xifeng Yan, P. S. Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *VLDB* 4, 11 (2011), 992–1003.
[21] Wang Xiao, Ji Houye, Shi Chuan, Wang Bai, Cui Peng, Yu P., and Ye Yanfang. 2019. Heterogeneous Graph Attention Network. In *WWW*.
[22] Zhiyong Zhang and Olfa Nasraoui. 2006. Mining search engine query logs for query recommendation. In *WWW*. 1039–1040.
[23] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *SIGKDD*. 635–644.
[24] Zhou Zhao, Ruihua Song, Xing Xie, Xiaofei He, and Yueting Zhuang. 2015. Mobile Query Recommendation via Tensor Function Learning. In *IJCAI*. 4084–4090.