Hierarchical Bipartite Graph Neural Networks: Towards Large-Scale E-commerce Applications

Zhao Li*, Xin Shen†§, Yuhang Jiao‡, Xuming Pan*, Pengcheng Zou*, Xianling Meng†, Chengwei Yao† and Jiajun Bu†
*Alibaba Group, China
Email: {lizhao.lz,xuming.panxm, xuanwei.zpc}@alibaba-inc.com
†Zhejiang University, Hangzhou, Zhejiang, China
Email: {sxstar,xgyxmxl,yaochw,bjj}@zju.edu.cn
‡School of Information, Central University of Finance and Economics, Beijing, China
Email: jiaoyuhang@email.cufe.edu.cn

***Abstract*—The e-commerce appeals to a multitude of online shoppers by providing personalized experiences and becomes indispensable in our daily life. Accurately predicting user preference and making a recommendation of favorable items plays a crucial role in improving several key tasks such as Click Through Rate (CTR) and Conversion Rate (CVR) in order to increase commercial value. Some state-of-the-art collaborative filtering methods exploiting non-linear interactions on a user-item bipartite graph are able to learn better user and item representations with Graph Neural Networks (GNNs), which do not learn hierarchical representations of graphs because they are inherently flat. Hierarchical representation is reportedly favorable in making more personalized item recommendations in terms of behaviorally similar users in the same community and a context of topic-driven taxonomy. However, some advanced approaches, in this regard, are either only considering linear interactions, or adopting single-level community, or computationally expensive. To address these problems, we propose a novel method with Hierarchical bipartite Graph Neural Network (HiGNN) to handle large-scale e-commerce tasks. By stacking multiple GNN modules and using a deterministic clustering algorithm alternately, HiGNN is able to efficiently obtain hierarchical user and item embeddings simultaneously, and effectively predict user preferences on a larger scale. Extensive experiments on some real-world e-commerce datasets demonstrate that HiGNN achieves a significant improvement compared to several popular methods. Moreover, we deploy HiGNN in Taobao, one of the largest e-commerces with hundreds of million users and items, for a series of large-scale prediction tasks of item recommendations. The results also illustrate that HiGNN is arguably promising and scalable in real-world applications.**

***Index Terms*—Hierarchical Representation, Graph Neural Network, Bipartite Graph, E-commerce Recommendations**

I. Introduction

The e-commerce era is witnessing a rapid development of online retailers who have attracted increasing people favoring online shopping, which conveniently provides items of interest with personalized experiences in our daily life. Leading e-commerce companies nowadays generate hundreds of millions of interactions (e.g. browsers, clicks, add-to-favorites, purchases, and comments) between tens of millions of users and a huge amount of items every day for a series of prediction tasks including Click Through Rate (CTR), Conversion Rate (CVR), personalized recommendation list, and so on [1]–[5]. Precisely predicting a user preference in such a complex environment of e-commerce is very vital for improving user experience, and quite challenging for increasing business volumes. However, unexploited information in these numerous interactions are valuable user preferences and item attractiveness in a collaborative manner, which is arguably beneficial to improving the performance of top-K recommendation and preference ranking [6]. More specific, collaborative filtering assumes that behaviorally similar users would exhibit similar preference on items, and vice versa [7]. As a result, users and items are vectorized as embeddings to reconstruct historical interactions for efficiently predicting user preference. Recently, graph neural networks (GNNs) have gained a high reputation of obtaining state-of-the-art results through effectively learned node embeddings of non-linear interactions in tasks such as node classification and link prediction [8]–[17]. In particular, [18] proposes a neural graph collaborative filtering method to explicitly integrate the user-item interactions into

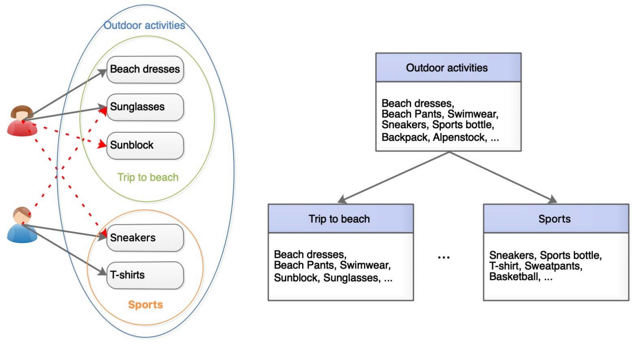

Outdoor activities
Beach dresses
Sunglasses
Sunblock
Trip to beach
Sneakers
T-shirts
Sports
Outdoor activities
Beach dresses, Beach Pants, Swimwear, Sneakers, Sports bottle, Backpack, Alpenstock, ...
Trip to beach
Beach dresses, Beach Pants, Swimwear, Sunblock, Sunglasses, ...
...
Sports
Sneakers, Sports bottle, T-shirt, Sweatpants, Basketball, ...

Fig. 1: An example of topic-driven taxonomy in E-commerce. The virtual arrows of a bipartite graph (*on the left*) indicate a user's potential preference on items, based on which the hierarchical topic tree (*on the right*) is constructed to represent conceptual shopping scenarios.

[§] is the corresponding author.

the embedding process, which is able to encode collaborative signal in the interaction graph structure by exploiting the high-order connectivity from user-item interactions and lead to a better performance of recommendation systems. Intuitively, collaborative filtering is an indication of the effect of community generalization on individual preference. However, in this regard, most state-of-the-art methods including neural graph collaborative filtering do not consider underlying user-community interactions or user hierarchy which have shown an advantageous performance over paradigms using user-item interactions alone [19]–[21].

Generally speaking, GNN methods are inherently flat and do not learn hierarchical representations of graphs. On one hand, it demonstrates in [20] that hierarchical representations of graphs can be combined with various graph neural network architectures in an end-to-end fashion to achieve prevailing results on graph classification benchmarks. Nonetheless, generating a hierarchical representation involves extensive and unscalable computation with the adjacent matrix of the graph. [19] learns a hierarchical representation of graphs by decomposing user information into two orthogonal spaces, each of which represents information captured by community level and individualized user preference respectively, which improves the prediction accuracy with promising scalability. But it neglects multi-level effects of hierarchical clustering and item hierarchy information which limits its application in unsupervised learning for computing meaningful and interpretable clusters on input graphs. On the other hand, [22] proposes an approach that automatically constructs an easy-to-interpret taxonomy on a large-scale bi-partite graph in a unsupervised manner, facilitating an efficient browsing navigation that enhances user search experiences with inherent high-order connections, and the resulting descriptive hierarchical tree is also reported in favor of making more personalized item recommendations to users within the same cluster (community) they belong to. For example, a user, who has bought beach dresses and sunglasses, may prefer the topic of "trip to beach", in which an item under this topic such as sunblock may be clicked. Also, dress and sunglasses may indicate a more extensive topic like "outdoor activities" at a higher level, in which an item such as sneakers may be clicked. Likewise, an item, such as sneakers, may attract some users who prefer "sports", in which a user such as a sports enthusiast may click. Also, sneakers may attract more extensive users who prefer "outdoor activities" at a higher level, in which a user such as an outdoor enthusiast may click on. In spite of flexible topic-driven taxonomy capturing user's intention in many scenarios, by performing parallel hierarchical agglomerative clustering, it is not sufficient to yield satisfactory embeddings for exploiting the user-item non-linear interactions.

In this paper, motivated by the above pioneering work, we aim to learn hierarchical representations on bi-partite graphs to not only exploit the hierarchical high-order connections but also capture non-linear interactions for the purpose of applying to a series of tasks, such as, user preference prediction, and personalized browsing navigation, in large-scale e-commerce scenarios. Therefore, we propose a **H**ierarchical b**i**partite **G**raph **N**eural **N**etwork (HiGNN) which allows one to stack multiple GNN modules in a hierarchical fashion. It builds a coarsened graph as input to the next GNN layer by performing general clustering algorithms on embeddings obtained from the previous GNN layer. The whole process repeats several times until a stopping criteria, i.e., a specified number of levels, are satisfied. Although HiGNN is a two-stage hierarchical representation learning by combining GNN with deterministic clustering algorithms, which is able to preserve non-linear interactions and hierarchical high-order connections as well. More importantly, it is easy to scale to a very large real-world application without involving computationally expensive matrix operations. We summarize the major contributions of this paper as follows:

1) First, we introduce a large-scale **H**ierarchical b**i**partite **G**raph **N**eural **N**etwork (HiGNN), which effectively and efficiently addresses the problem of utilizing high-order connections and non-linear interactions through hierarchical representation learning on bi-partite graphs. Moreover, it is scalable to large-scale sparse graph data related applications.
2) Second, from perspective of supervised learning, by stacking multiple GNN in a hierarchical fashion, HiGNN is able to obtain hierarchical user preferences and the hierarchical item attractiveness through the learned hierarchical structure for depicting users and items precisely. Extensive experiments on large-scale e-commerce datasets, both online and offline, show its prevailing performances in a couple of prediction tasks including Click Through Rate (CTR) and Conversion Rate (CVR).
3) Third, from perspective of unsupervised learning, we apply HiGNN to automatically generate a topic-driven taxonomy of a large-scale real-world e-commerce for providing browsing navigation with personalized recommendation lists to enhance user experiences. It not only demonstrates the superiority of HiGNN compared to existing methods for placing favorable items into right topics, but also boosts the user preference prediction in terms of precision, which sheds a light on its promising applications in large-scale real-world e-commerce.

The remainder of this paper is organized as follows: in Section II, we investigate most related works. Section III gives a detailed description of the proposed HiGNN approach. Experimental results on the real world e-commerce applications for supervised learning are shown in Section IV, while the demonstrations of building a concept-driven taxonomy from unsupervised learning perspective are displayed in Section V. At last, Section VI concludes the paper.

II. Related Works

Our work builds upon a rich line of recent research on graph neural networks (GNNs) with the aim of applying to a serials of e-commerce prediction tasks such as click through rate, conversion Rate, and topic-driven taxonomy. In particular, the GNNs with collaborative filtering and GNNs with hierarchical representation are most relevant literature regarding e-commerce applications and investigated elaborately here.

A. Graph Neural Networks

In recent years, graph neural networks (GNNs) have exerted a tremendous fascination on research community dedicating to effectively learn node embeddings over graph structured data, such as social network data or graph-based representations. GNNs treat the underlying graph as a computation graph and generate individual node embeddings by passing, transforming, and aggregating node feature information across the graph [23]–[26]. The generated node embeddings are widely used as input to any prediction tasks, e.g., for node classification, link prediction, and item recommendation.

B. Graph-based Collaborative Filtering

Another line of research [27], [28] exploits the user-item interaction graph to infer user preference in a collaborative fashion, assuming behaviorally similar users would exhibit similar preference on items. Intuitively, integrating user-item interactions into the embedding function could contribute to making better user preference prediction. An approach named HOP-Rec in [29] performs random walks to enrich the interactions of a user with multi-hop connected items, which is beneficial to obtain better embeddings by partially capturing the collaborative effect of user-item interactions. The recently proposed neural graph collaborative filtering method is designed to propagate embeddings recursively on the graph for modeling the high-order connectivity information in the embedding function, a natural way that encodes collaborative signal in the interaction graph structure. Most of graph-based collaborative filtering methods are highly depended on matrix operations, such as matrix factorization or matrix multiplication, which makes it less scalable on large-scale graphs [30].

C. Hierarchical Graph Representation

General GNN based methods are inherently flat as they only propagate information across edges of a graph and generate individual node embeddings, which is problematic or inefficient for predicting the label associate with the entire graph. However, learning hierarchical representations of graph enjoys its outstanding features in graph classification and clustering, and becomes prevailing in several scenarios such as link prediction, e-commerce recommendation, etc, [19], [22]. There are some recent works that learn hierarchical graph representations by combining GNNs with different clustering processes. In particular, the recently proposed approach DIFFPOOL [20], a differentiable graph pooling module that can generate hierarchical representations of graphs and can be combined with various graph neural network architectures in an end-to-end fashion. It hierarchically learns a differentiable soft assignment at each layer of a deep GNN, mapping nodes to a set of clusters based on their learned embeddings. DIFFPOOL obtains favorable hierarchical representation and computes meaningful and interpretable clusters on the input graphs, while requiring explicitly expressing with the adjacent matrix of the graph. Consequently, it is computationally expensive that make it less popular in handling large-scale graphs [30]. On the other hand, some researchers [31], [32] illustrate a user's community-level embedding to be effective in graph classification tasks, in addition to a user's individual embedding. In [20], authors make some efforts in effectively co-training two embeddings by decomposing user information into two orthogonal spaces, each of which represents information captured by community level and individualized user preference respectively. Another intriguing application of hierarchical graph representation is e-commerce taxonomy for offering a personalized dynamic shopping navigation. [22] illstrates a topic-driven hierarchical taxonomy based on user-item bi-partite graph in presence of query interactions effectively expressing user intention. It establishes correlation between categories of ontology-driven taxonomy, and offers an explainable recommendation with a noticeable prediction accuracy. While it is shown advantageous in some applications, it is not sufficient to capture the user-item non-linear interactions as it perform a traditional hierarchical agglomerative clustering to explore the hierarchical structure.

III. Hierarchical Graph Neural Networks

In this section, we present the proposed HiGNN framework. First, we introduce bipartite GraphSAGE on a user-item graph to project user vertices and item vertices into two different feature spaces, i.e., user embedding and item embedding. Then, we elaborate on HiGNN implementation regarding constructing the hierarchical structure.

A. Preliminaries

The user-item graph is a quadruple $G = (U, I, E, S)$, where users $U = \{u_1, u_2, \dots, u_M\}$ and items $I = \{i_1, i_2, \dots, i_N\}$ are two sets of vertices, E is the set of edges and each edge $\{e = (u_m, i_n) | u_m \in U, i_n \in I\}$ is associated with a weight $S(e)$ to denote the connection strength. An edge (u_m, i_n) exists if user u_m clicks item i_n in the behavior history. In this graph, there are no edges between users or between items. The CTR prediction task is to learn a function between a user u_m and an item i_n, which can be used to predict the probability i_n is clicked by u_m.

B. Bipartite GraphSAGE

The intuition behind bipartite GraphSAGE is that at each iteration, a user aggregates information from local neighbor items, and an item aggregates information from local neighbor users. As this process iterates, vertices incrementally gain more and more information from further reaches of the graph.

We first denote $\text{AGGREGATE}_u^p, \forall p \in \{1, 2, \dots, P\}$ and $\text{AGGREGATE}_i^p, \forall p \in \{1, 2, \dots, P\}$ as the aggregator functions for users and items, which aggregate information from neighbors respectively. Denote sets of weight matrices $\boldsymbol{W}_u^p, \forall p \in \{1, 2, \dots, P\}$ and $\boldsymbol{W}_i^p, \forall p \in \{1, 2, \dots, P\}$ for users and items, which are used to propagate information between different steps of the model.

In the bipartite GraphSAGE method, the entire bipartite graph $G = (U, I, E, S)$ and features for all users $\boldsymbol{X}_u = \{\boldsymbol{x}_u, \forall u \in U\}$ and all items $\boldsymbol{X}_i = \{\boldsymbol{x}_i, \forall i \in I\}$ are provided as input, where $\boldsymbol{x}_u \in \mathbb{R}^{d_u}$ and $\boldsymbol{x}_i \in \mathbb{R}^{d_i}$. Denote $\boldsymbol{h}_u^p$ and $\boldsymbol{h}_i^p$ as the embedding of user and item at step p, and $\boldsymbol{h}_u^0 = \boldsymbol{x}_u$ and $\boldsymbol{h}_i^0 = \boldsymbol{x}_i$ for all users and items. In step p, each user u aggregates item embeddings in its immediate neighborhood, $\boldsymbol{h}_i^{p-1}, \forall i \in N(u)$, and transforms item embeddings into the corresponding user embedding $\boldsymbol{h}_{N(u)}^p$ by multiplying a transformation matrix $\boldsymbol{M}_i^u$. This process is described as

$$\boldsymbol{h}_{N(u)}^p \leftarrow \boldsymbol{M}_i^u \cdot \mathrm{AGGREGATE}_u^p(\{\boldsymbol{h}_i^{p-1}, \forall i \in N(u)\}). \quad (1)$$

With the same consideration, the aggregated item embedding is derived from

$$\boldsymbol{h}_{N(i)}^p \leftarrow \boldsymbol{M}_u^i \cdot \mathrm{AGGREGATE}_i^p(\{\boldsymbol{h}_u^{p-1}, \forall u \in N(i)\}), \quad (2)$$

where $\boldsymbol{M}_u^i$ is the transformation matrix from user to item. Any type of aggregator is available and we adopt mean aggregator in our demonstration.

After aggregating the neighboring vertex embedding, bipartite GraphSAGE concatenates the vertex (both users and items) current embedding with the aggregated neighborhood embedding, and feeds the concatenated embedding through a full connection layer with nonlinear activation function σ, in order to transform the embedding to be used at the next step of the method. The method is expressed as

$$\boldsymbol{h}_u^p \leftarrow \sigma\big(\boldsymbol{W}_u^p \cdot \mathrm{CONCAT}(\boldsymbol{h}_u^{p-1}, \boldsymbol{h}_{N(u)}^p)\big) \quad (3)$$

for users and

$$\boldsymbol{h}_i^p \leftarrow \sigma\big(\boldsymbol{W}_i^p \cdot \mathrm{CONCAT}(\boldsymbol{h}_i^{p-1}, \boldsymbol{h}_{N(i)}^p)\big) \quad (4)$$

for items. Denote $\boldsymbol{z}_u \equiv \boldsymbol{h}_u^P, \forall u \in U$ and $\boldsymbol{z}_i \equiv \boldsymbol{h}_i^P, \forall i \in I$ as the final embedding output at step P.

In order to learn useful, predictive embeddings in a fully unsupervised setting for bipartite graphs, we apply a bipartite graph-based loss function to the output embeddings, $\boldsymbol{z}_u, \forall u \in U$ and $\boldsymbol{z}_i, \forall i \in I$, and tune the weight matrices, $\boldsymbol{W}_u^p, \forall p \in \{1, 2, \ldots, P\}, u \in U$ and $\boldsymbol{W}_i^p, \forall p \in \{1, 2, \ldots, P\}, i \in I$, the transformation matrices, $\boldsymbol{M}_i^u$ and $\boldsymbol{M}_u^i$, and parameters of the aggregator functions via stochastic gradient descent. The bipartite graph-based loss function encourages nearby users and items have similar embeddings, while enforcing that embeddings of disparate users and items are highly distinct:

$$\begin{aligned} J_{BG} = &- \log\big[\sigma\big(f[\mathrm{CONCAT}(\boldsymbol{z}_u, \boldsymbol{z}_i), S((u,i))]\big)\big] \\ &- Q_u \cdot \mathbb{E}_{u_n \sim P_n(u)} \log\big[\sigma\big(f[\mathrm{CONCAT}(\boldsymbol{z}_{u_n}, \boldsymbol{z}_i), \gamma]\big)\big] \\ &- Q_i \cdot \mathbb{E}_{i_n \sim P_n(i)} \log\big[\sigma\big(f[\mathrm{CONCAT}(\boldsymbol{z}_u, \boldsymbol{z}_{i_n}), \gamma]\big)\big], \end{aligned} \quad (5)$$

where (u, i) is a pair of user and item if an edge exists between them. f is a full connection network for generating similarity based on the concatenation of user embedding and item embedding, and the corresponding edge weight. σ is the sigmoid function. P_n is a negative sampling distribution. Q_u and Q_i are defined as the number of negative samples for users and items, respectively. γ is a hyper-parameter for denoting the weight of negative samples.

C. HiGNN implementation

In this subsection, we present HiGNN, a network that allows one to stack multiple GNN modules, i.e., bipartite GraphSAGE, in order to construct a hierarchical structure in an end-to-end fashion.

Denote $\boldsymbol{Z}_u = \{\boldsymbol{z}_u, \forall u \in U\}$ and $\boldsymbol{Z}_i = \{\boldsymbol{z}_i, \forall i \in I\}$ as the sets of user embedding and item embedding, respectively. For brevity, we denote $(\boldsymbol{Z}_u, \boldsymbol{Z}_i) \leftarrow \mathcal{BG}(G, \boldsymbol{X}_u, \boldsymbol{X}_i)$ as the implementation of bipartite GraphSAGE with input bipartite graph G, user features $\boldsymbol{X}_u$ and item features $\boldsymbol{X}_i$ in the following article.

Given $(\boldsymbol{Z}_u, \boldsymbol{Z}_i)$ and the origin user-item graph, we adopt some clustering approach, i.e., K-means, to cluster similar users and similar items together in their own feature spaces, respectively. We consider user clusters C_u and item clusters C_i clustered by K-means as new users and items in a new coarsened user-item graph. The user cluster feature $\boldsymbol{X}_{C_u}$ is able to be expressed as the average user embedding of users who belong to the cluster. With a similar method, the item cluster feature $\boldsymbol{X}_{C_i}$ can be expressed. The edge weight of (C_u, C_i) in the new coarsened graph is calculated as

$$S(C_u, C_i) = \sum_e S(e), \forall e = (u, i) \in G, u \in C_u, i \in C_i, \quad (6)$$

where $u \in C_u$ means user u belongs to user cluster C_u, and $i \in C_i$ has the similar meaning. An edge is existed between C_u and C_i if and only if $S(C_u, C_i) > 0$.

Based on all information above (vertices, edges and edge weights), we are able to construct a new coarsened user-item graph. This graph is able to be used as input to the bipartite GraphSAGE at the next level. For clearly, denote $K_u(\boldsymbol{Z}_u^l)$, $K_i(\boldsymbol{Z}_i^l)$ as the K-means process at level l for users and items respectively. Denote $F(C_u^l, C_i^l, G^{l-1})$ as the coarsened graph construction process at level l. The hierarchical structure is able to be constructed by repeating bipartite GraphSAGE L times. We summarize the implementation of HiGNN in Algorithm 1. Then, the learned hierarchical user preference and hierarchical item attractiveness from the hierarchical structure can be utilized for the subsequent prediction tasks.

D. Algorithm Complexity Analysis

As we can see, the user/item aggregator in bipartite GraphSAGE and clusting are the main operations. For the two kind of operations, the computational complexity of the first layer is dominant. For the first layer of GNN, the computational complexity of aggregator is $O((M + N)(K_1 * K_2))$, where M is the number of users, N is the number of items and K_1 and K_2 is the number of neighbors sampled at the depth of 1 and 2 respectively. For the first layer of Kmeans, we use the single-pass version which estimates the cluster centers with a single pass over all data and is appropriate for large-scale clustering. Thus, the computational complexity is $O(M * K_u + N * K_i)$, where K_u and K_i is the specified cluster number of user and item respectively.

```
Algorithm 1: HiGNN implementation
  Input: User-item graph $G(U, I, E, S)$, user features
         $\boldsymbol{X}_u$ and item features $\boldsymbol{X}_i$
  Output: Hierarchical structure
          $\mathcal{G} \leftarrow \{G^0, G^1, \ldots, G^L\}$,
          $\mathcal{Z}_u \leftarrow \{\boldsymbol{Z}_u^1, \ldots, \boldsymbol{Z}_u^L\}$ and
          $\mathcal{Z}_i \leftarrow \{\boldsymbol{Z}_i^1, \ldots, \boldsymbol{Z}_i^L\}$
1 $\mathcal{G}, \mathcal{Z}_u, \mathcal{Z}_i \leftarrow \{G^0\}, \{\}, \{\}$;
2 $l \leftarrow 1$;
3 while $l \leq L$ do
4     $(\boldsymbol{Z}_u^l, \boldsymbol{Z}_i^l) \leftarrow \mathcal{BG}(G^{l-1}, \boldsymbol{X}_u^{l-1}, \boldsymbol{X}_i^{l-1})$;
5     $C_u^l, C_i^l \leftarrow K_u(\boldsymbol{Z}_u^l), K_i(\boldsymbol{Z}_i^l)$;
6     $(G^l, \boldsymbol{X}_u^l, \boldsymbol{X}_i^l) \leftarrow F(C_u^l, C_i^l, G^{l-1})$;
7     $\mathcal{G}, \mathcal{Z}_u, \mathcal{Z}_i \leftarrow \mathcal{G} \cup G^l, \mathcal{Z}_u \cup Z_u^l, \mathcal{Z}_i \cup Z_i^l$;
8     $l \leftarrow l + 1$;
9 end
```

IV. Hierarchical Bipartite Graph Neural Network for E-commerce Prediction

HiGNN obtains hierarchical user preferences and hierarchical item attractiveness by stacking multiple GNNs in a hierarchical fashion. To utilise the learned user embeddings and item embeddings to precisely predict e-commerce tasks, we develop a deep neural network with HiGNN. Extensive experiments on large-scale e-commerce datasets, including offline and online, show our method outperforms other popular compared algorithms.

A. Supervised Deep Neural Network with HiGNN for E-commerce Predictions

As shown in Figure 2, a supervised deep neural network with HiGNN for e-commerce predictions is proposed to solve a series of prediction tasks, including CTR, CVR, and personalized recommendation. In this section, we extract the hierarchical user preference and hierarchical item attractiveness from the hierarchical structure and utilize it for CVR prediction.

Considering user and item embeddings obtained from the user-item graph, $\boldsymbol{z}_u$ describes the user preference on different items and $\boldsymbol{z}_i$ describes the item attractiveness on different kinds of users. In a CVR prediction task, for a user u and a candidate item i, if the user preference of u and the item attractiveness of i are matched, we predict that user i will purchase the item i.

In our proposed hierarchical structure, the user cluster embedding $\boldsymbol{z}_{C_u}^l$ at level l presents the level l user preference for all users in this cluster. Thus, we derive the hierarchical user preference of user u by concatenating the user cluster embedding (user embedding at the first level) $\boldsymbol{z}_u^H = \mathrm{CONCAT}(\boldsymbol{z}_u^1, \boldsymbol{z}_u^2, \ldots, \boldsymbol{z}_u^L)$, which synthesizes different-grained user preferences together. With the similar consideration, we obtain hierarchical item attractiveness of item i as $\boldsymbol{z}_i^H = \mathrm{CONCAT}(\boldsymbol{z}_i^1, \boldsymbol{z}_i^2, \ldots, \boldsymbol{z}_i^L)$.

Given the hierarchical user preference, hierarchical item attractiveness, user profile (gender, purchasing power, etc.) and item statistic (click count, purchase count, etc.) as input, full connection layers are used to learn the combination of features automatically. The loss function of CVR prediction problem is defined as

$$J_{CVR} = -\frac{1}{N_T} \sum_{(x,y)} \left[y \log p(x) + (1-y) \log(1-p(x)) \right], \quad (7)$$

where N_T is the size of the training set, with x as the input of the network and $y \in \{0, 1\}$ as the label, $p(x)$ is the output of the network after sigmoid function, indicating the predicted probability of sample x be purchased.

B. Offline Experiments and Results

We design our offline experiments to demonstrate whether the hierarchical user preference and hierarchical item attractiveness help to improve CVR prediction accuracy, and discuss the sensitivity of hyper-parameters in HiGNN.

1) Datasets and Metrics: We use Taobao dataset, a real-world industry dataset, to evaluate the performances of different methods. Table I summarizes the statistics of our experimental datasets.

TABLE I: Statistical Information of Datasets

Dataset	Users	Items	User-Item Clicks	Density
Taobao #1	34,519,150	13,296,702	280,522,717	6.11e-7
Taobao #2	11,727,217	3,053,149	1,109,274	3.10e-8

Taobao datasets contain user-item click behaviors and transactions on Taobao, one of the largest online e-commerce platforms in the world. One week's logs are used for training and logs of the following day for testing. Specifically, Taobao #1 utilizes click and transaction logs in one week as the training set, and click and transaction logs on the next day as the testing set. Taobao #2 utilizes click and transaction logs about new arrival products in one week as the training set, and click and transaction logs about new arrival products on the next day as the testing set. Taobao #2 dataset concerns cold-start scenario, which focuses on the new items published within 2 months. Thus, Taobao #2's data density is relatively smaller than Taobao #1's.

We consider purchase behaviors as positive samples, and click behaviors without purchasing as negative samples. Because the number of positive samples is relatively small, to achieve better performance, we adopt a replicate sampling strategy to make the ratio of positive samples to negative samples as 1:3 in Taobao #1 dataset. However, to keep the real cold-start scenario in the e-commerce system, and test different algorithms in relatively sparse and unbalanced data, we utilize original records in Taobao #2 dataset. Table VI summarizes the statistics of samples in datasets.

We adopt the area under the receiver operator curve (AUC) to evaluate the performance of all the methods [33]. AUC is the most popular evaluation metric on prediction tasks in both research and industry area. Larger AUC means better performance.

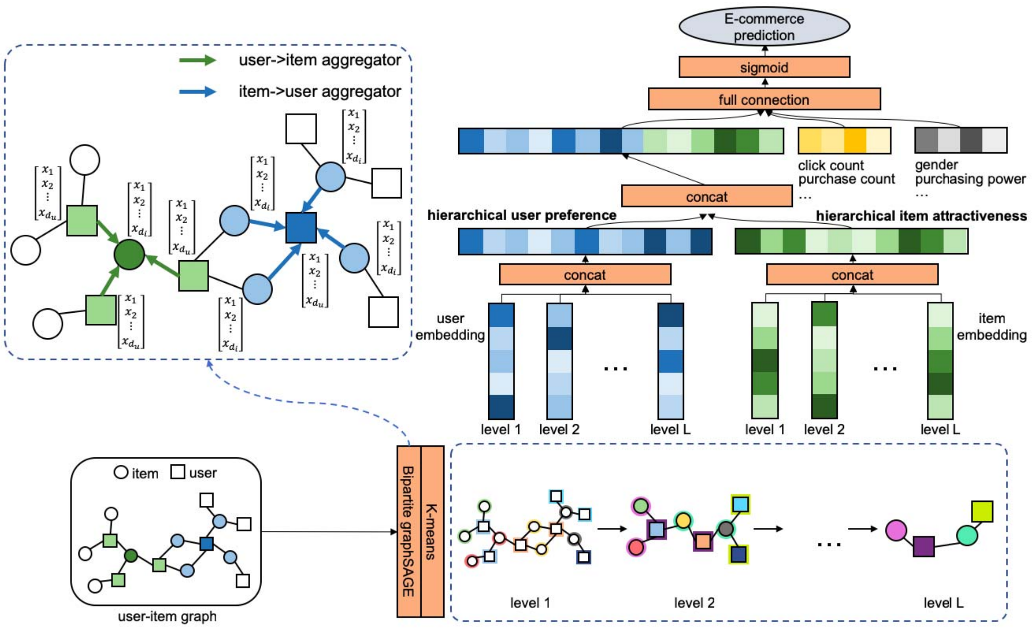


Fig. 2: Supervised Deep Neural Network with HiGNN for E-commerce Predictions

TABLE II: Samples Information of Datasets

Dataset	Training Set			Testing Set
	Positive	Negative	Total	Total
Taobao #1	78,988,312	223,612,179	302,600,491	40,824,588
Taobao #2	2,074,792	28,689,261	30,764,053	3,986,179

2) Compared Algorithms and Settings: To the best of our knowledge, no existing algorithm can deal efficiently with hierarchical user preferences and hierarchical item attractiveness to predict real-world e-commerce tasks of such large scale, including [30] and [20]. Our baseline algorithms are as follows:

- CGNN: A graph neural network method learns two user embeddings for prediction by decomposing user information into two orthogonal spaces, each of which represents information captured by community level and individualized user preference respectively. CGNN can be considered as a special case of our proposed method, which fixes the number of user levels to 2. The parameter of CGNN refers to [19].
- DIN: A popular deep neural network method without graph structure information and hierarchical information in the e-commerce system. DIN can be regarded as a special case of our proposed method at level 0 ($L = 0$). The parameter of DIN refers to [1].
- GE: Single level Graph Embedding-based (GE) method, which is our proposed method using only one level, without hierarchical information.
- HUP-only: Submodel of our proposed method, which considers Hierarchical User Preference only, without item attractiveness.
- HIA-only: Submodel of our proposed method, which considers Hierarchical Item Attractiveness only, without user preference.

We deploy our algorithm on Alibaba's server clusters comprising of 300 computing workers with 3000 CPUs. Empirically, both the dimension of user embedding d_u and item embedding d_i in bipartite GraphSAGE equal to 32. To keep consistency and simplicity, in this paper, we set the level number of a hierarchical structure $L = 3$ and the K-means parameter K^l at level l satisfies $K^l = K^{l-1}/\alpha$, $\alpha = 5$ in Taobao datasets for achieving better performance. Furthermore, we will discuss the parameters' sensitiveness in this section later. We also set sizes of fully connected layers as 256, 128 and 64, learning rate as 0.001, batch size as 1024. L2-norm is used for regularization. Leaky ReLu is utilized as the activation function, and Sigmoid is utilized as the loss function.

3) Performance Comparison: Table III lists the performance results of all compared methods. We evaluate our proposed method, HiGNN, with five popular and state-of-the-art methods in two different Taobao datasets.

TABLE III: Performance Evaluation (AUC).

Dataset	CGNN	DIN	GE	HUP-o	HIA-o	HiGNN
Taobao #1	0.829	0.844	0.863	0.853	0.855	**0.870**
Taobao #2	0.875	0.870	0.893	0.881	0.881	**0.899**

It can be observed that our proposed method significantly

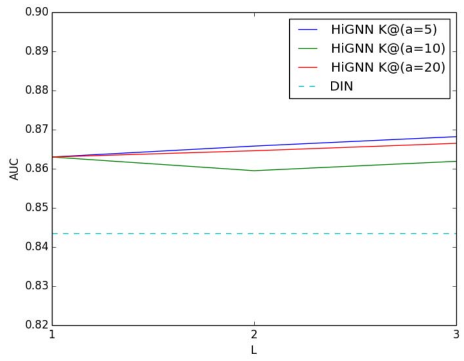


Fig. 3: AUC comparisons with different K and L strategies.

outperforms the baselines in all two datasets in terms of AUC. In particular, GE is better than DIN, which indicates that the graph embedding method is able to represent the user preference and the item attractiveness more precisely. Both HUP-only and CGNN consider user hierarchical embedding without item hierarchical embedding. Because CGNN fixes the level to 2, it is relatively worse than HUP-only. Our proposed method performs better than HUP-only and HIA-only, which indicates that either the user hierarchical embedding or item hierarchical embedding is effective for the CVR prediction model. Moreover, the proposed method performs best among all compared methods, which indicates that the combination of hierarchical embedding of user and item can further improve the CVR prediction accuracy. A more important thing we observed is that HiGNN still works on very sparse dataset, i.e. the cold-start scenario. Notice that HiGNN outperforms DIN by 3.08% and 3.33% in Taobao #1 and #2 respectively, hierarchical information works more effectively when the graph is sparse.

4) Sensitivity Analysis: There are two key hyper-parameters having the most influence on HiGNN, i.e., the level number L and the K-means parameter K. We conduct experiments for investigating the influence of different L and different K update strategy. The results are shown in Fig. 3.

In this figure, we also plot one compared algorithm, DIN, which can be regarded as a special case of our proposed method at level 0 ($L = 0$). It can be observed that adding hierarchical information can achieve better performance. In most cases, AUC increases with an increase of L when L is less than or equal to 3, which demonstrates the hierarchical information is able to improve AUC performance. Obtaining more levels of user embeddings and item embeddings can better learn user preferences and item attractiveness.

Different K strategies lead to different clustering results, and the influence of clustering is determined as the distribution and scale of datasets. In particular, we set $K^l = \frac{K^{l-1}}{\alpha}$ from $\alpha = 5$ to $\alpha = 20$ on Taobao #1 dataset, which are shown in Fig. 3. Larger α can decrease the scale of hierarchical bi-partite graphs quickly, which reduces the running time. However, Larger α also means more information loss. So smaller α can achieve better performance, specifically, $\alpha = 5$ in the best in this parameter sensitiveness experiment.

C. Online Experiments and Results

This subsection shows the results of the online evaluation in the large-scale real-world e-commerce platform, Taobao system, with a standard A/B testing configuration. We mainly concern crucial e-commerce evaluation metrics as follows:

- Unique Visitor (UV): the number of different clicked visitors, indicating whether our recommendation can attract users to click.
- transaction CouNT (CNT): the number of transactions, indicating whether our recommendation can increase sales volume.
- CTR: the ratio of click number to visit number, one popular evaluation metric in an e-commerce system, especially in search and recommendation scenarios.
- CVR: the ratio of transaction number to click number, which directly shows the recommendation effectiveness on sales volume.

We apply our model on the real Taobao e-commerce online system for new arrival products to solve the cold-start problem. Table IV reports the results of improvement of UV, CNT, CTR, and CVR on two testing days.

TABLE IV: Online A/B Testing of Performance Evaluation

Date	Day 1	Day 2
UV	43,514 $\rightarrow$ 44,341 (+1.90%)	48,531 $\rightarrow$ 49,522 (+2.04%)
CNT	54,438 $\rightarrow$ 55,940 (+2.76%)	60,717 $\rightarrow$ 62,001 (+2.11%)
CTR	0.3569 $\rightarrow$ 0.3581 (+0.34%)	0.3469 $\rightarrow$ 0.3492 (+0.66%)
CVR	0.1226 $\rightarrow$ 0.1253 (+2.25%)	0.1206 $\rightarrow$ 0.1231 (+2.09%)

The results show that our proposed method increases commercial volumes under all e-commerce evaluation metrics. Both CNT and CVR, two crucial commercial metrics reflecting the sales volume, are improved by more than 2% on two A/B testing days. Moreover, HiGNN also increases UV and CTR, which indicates our proposed method can achieve user preferences and item attractiveness more precisely, and attract more visitors to click their interested items.

V. Hierarchical Bipartite Graph Neural Networks for E-commerce Taxonomy

In this section, we first briefly revisit the background about taxonomy and the basic definition of topic-driven taxonomy based on query-item graphs. Then we introduce the Hierarchical Bi-partite Graph Neural Network (HiGNN) on a query-item graph, which is an indication of users' search intention and somewhat different from the method on a user-item graph. Next we compare our proposed HiGNN method with the current taxonomy solution of Taobao, giving the quantitative experiments and analysis. Finally, we demonstrate the cases of constructing a topic-driven taxonomy structure from large-scale real-world e-commerce data using our proposed approach.

A. Brief Background

In addition to the specific e-commerce prediction applications, taxonomy construction is another crucial task in e-commerce scenarios. Dictionary-based ontology taxonomy [34] is a widely used method to organize items into categorical structures in most existing e-commerce platforms, because the hierarchical conceptual knowledge behind the items can be naturally distilled into the ontology dictionary. Take Fig. 4 for example, "Beach Dress" is a leaf category and belongs to a parent category "Dress" and a grandparent category "Women's Clothing". The ontology taxonomy follows the pre-defined dictionary rules to manage items, which ignores the correlations between items in the shopping history and results in low coverage of the taxonomy. What's more, the terms in the dictionary are likely to be highly redundant since the concept of one item can be expressed in many different ways. Thus the ontology taxonomy in many cases can't capture the user's search intention. However, helping users explore items with categories sharing the same topic, in other words, the same search intention, in massive number of data is one of the most important characteristics required for e-commerce systems.

Noticeably, user's search queries can effectively reflect the topics of interest to the user, to bridge the gap between item taxonomy and user search intention, we apply our HiGNN on the query-item graph to automatically generate a new topic-driven taxonomy structure.

1) ***Query-Item Graph****:* Similar to the user-item graph, a query-item graph can be represented by the quadruple $G = (Q, I, E, S)$, where queries $Q = \{q_1, q_2, \dots, q_M\}$ and items $I = \{i_1, i_2, \dots, i_N\}$ are two sets of vertices, $E = \{e = (q_m, i_n) \mid q_m \in Q, i_n \in I\}$ is the set of edges and S is the weights set where $S(e)$ denotes the connection strength of edge e. Note that an edge (q_m, i_n) exists if and only if a user clicked item i_n from the resulting items of the query q_m. In addition, there is no edge between query-query pair or item-item pair in this graph, hence it is a bi-partite graph.

2) ***Topic-Driven Taxonomy****:* The taxonomy construction task in e-commerce scenarios is to organize hundreds of millions of items into hierarchical topics-based structures, in which each topic consists of a group of items sharing the same user's search intention and a set of descriptions extracted from search queries. In particular, the taxonomy structure in this paper is constructed from the query-item graph, which not only contains the conceptual information, but also considers the conceptual shopping relationship on the query-item graph.

B. HiGNN on Query-Item Graphs

Different from the user-item graph in e-commerce prediction task, the query-item graph has features embedded into the same space.

Specifically, the e-commerce prediction task is to measure the user's preference for different items and the attractiveness of an item to different users, which requires a lot of user information, such as age, gender, education level etc., and

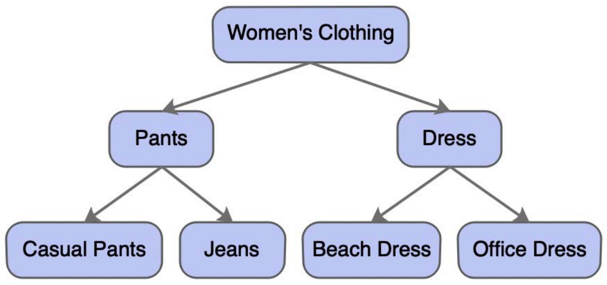


Fig. 4: An example of ontology-driven taxonomy in E-commerce, each node of which represents a category.

item information, e.g., price, sales, brand, etc., those are orthogonally distributed in different feature spaces. However, taxonomy task is to construct a hierarchical topics-driven structure from the historical interactions between queries and items, especially from those keywords and titles, first capturing the hierarchical topics preserved in items and then assigning descriptions extracted from search queries to each topic. Note that the original keywords and titles of both queries and items in taxonomy task are composed of texts, which allows us to exploit the widely used natural language processing technique, word2vec [35], to embed the original features of queries and items into the same latent space. Thus, the Bi-partite Graph Neural Network applied on query-item graphs is slightly different from the method on the user-item graphs.

First, we denote AGGREGATE as the aggregate operation for both queries and items, and $\boldsymbol{W}$ as the weight matrix used to propagate information between query-item vertices and their local neighbors. For each aggregate operation step $p \in \{1, 2, \dots, P\}$, the aggregate operations and the weight matrices can be represented by AGGREGATE^p and $\boldsymbol{W}^p$. The inputs of this Bi-partite Graph Neural Network are the entire bipartite query-item graph $G = (Q, I, E, S)$ and features for all queries $\boldsymbol{X}_Q = \{\boldsymbol{x}_q, \forall q \in Q\}$ and all items $\boldsymbol{X}_I = \{\boldsymbol{x}_i, \forall i \in I\}$, where $\boldsymbol{x}_q, \boldsymbol{x}_i \in \mathbb{R}^{d_w}$ since the features are in the same word-embedding latent space.

1) ***Aggregator****:* We denote $\boldsymbol{h}_q^p$ and $\boldsymbol{h}_i^p$ as the embedding of query and item at the aggregation step p, and $\boldsymbol{h}_q^0 = \boldsymbol{x}_q$ and $\boldsymbol{h}_i^0 = \boldsymbol{x}_i$. In step p, each query q aggregates information from its one-hop neighborhood, denotes as $\boldsymbol{h}_i^{p-1}, \forall i \in N(q)$, and transforms the features into the corresponding query features $\boldsymbol{h}_{N(q)}^p$ by multiplying a transformation matrix $\boldsymbol{M}^p$. This process is described as

$$\boldsymbol{h}_{N(q)}^p \leftarrow \boldsymbol{M}^p \cdot \text{AGGREGATE}^p(\{\boldsymbol{h}_i^{p-1}, \forall i \in N(q)\}). \quad (8)$$

Note that the item and query features are embedded into the same space, with the shared transformation matrix $\boldsymbol{M}^p$, the aggregated item embedding is derived from

$$\boldsymbol{h}_{N(i)}^p \leftarrow \boldsymbol{M}^p \cdot \text{AGGREGATE}^p(\{\boldsymbol{h}_q^{p-1}, \forall q \in N(i)\}), \quad (9)$$

similar to the supervised part, we adopt mean aggregator in this paper.

*2) **Dense Layer**:* After aggregating the neighboring information, we concatenate the target vertex embedding with the aggregated neighborhood embedding, then feed the concatenated embedding through dense layer with a nonlinear activation function σ. The method is expressed as

$$\boldsymbol{h}_q^p \leftarrow \sigma\big(\boldsymbol{W}^p \cdot \text{CONCAT}(\boldsymbol{h}_q^{p-1}, \boldsymbol{h}_{N(q)}^p)\big) \tag{10}$$

for queries and

$$\boldsymbol{h}_i^p \leftarrow \sigma\big(\boldsymbol{W}^p \cdot \text{CONCAT}(\boldsymbol{h}_i^{p-1}, \boldsymbol{h}_{N(i)}^p)\big) \tag{11}$$

for items. Denote $\boldsymbol{z}_q = \boldsymbol{h}_q^P$ and $\boldsymbol{z}_i = \boldsymbol{h}_i^P$ as the final embedding output at step P.

*3) **Unsupervised Loss**:* As we mentioned above, the input embeddings of queries and items are in the same latent space, and the differentiable weight matrices for queries and items are shared, which suggests that the output embeddings of both queries and items are distributed in the same feature space. Following the same rules, the unsupervised query-item graph-based loss function can be rewritten as:

$$\begin{aligned} J_{BG} = &- \log\left[\sigma\big(f[\text{CONCAT}(\boldsymbol{z(e)}), S(e)]\big)\right] \\ &- N_n \cdot \mathbb{E}_{e_n \sim P_n(e)} \log\left[\sigma\big(f[\text{CONCAT}(\boldsymbol{z(e_n)}), \gamma]\big)\right], \end{aligned} \tag{12}$$

where $e = (q, i)$ is a pair of query and item if e exists in set E. f is Multi-Layer Perceptron (MLP) for generating similarity based on the concatenation of query-item embeddings and edge weight $S(e)$. σ is the activation function. P_n is a negative sampling distribution. N_n is defined as the number of negative samples for edges. γ is a hyper-parameter for denoting the weight of negative samples.

C. Topic-Driven Taxonomy Construction

In this subsection, we show how to generate topic-driven taxonomy with the unsupervised HiGNN, and to obtain a set of meaningful topic descriptions.

1) Hierarchical Taxonomy: The intuition behind HiGNN for taxonomy is to enhance the connections between queries and items that share the same search intention, i.e., the same topic, by clustering vertices which have the similar learned embeddings. The new coarsened graph consisting of the centroid of each cluster will be considered as the input of HiGNN at the next level. After repeating this coarsening procedure a few times, we can construct a hierarchical topic-based structures from the query-item graph. Obviously, the taxonomy results are very sensitive to the number of clusters that we set to be coarsened at each level. In order to generate a better clustering result, we exploit the Calinski-Harabasz Index [36] to maximize the between-cluster variance and minimize the within-cluster variance, the objective function can be formulated as

$$\max \quad CH = \frac{D_B(k)}{D_W(k)} \times \frac{N-k}{k-1}, \tag{13}$$

where k is the number of clusters, $D_B(k)$ denotes the between-cluster variance, $D_W(k)$ denotes the within-cluster variance and N is the number of point data.

2) Topic Description Matching: Once the topic-driven taxonomy structure is generated, the topic is probably associated with a set of items and each item will be connected with a number of queries. To make the topic more interpretive, we follow the similar strategy described in [37] to find the most representative query as the description for a specific topic. The topic description matching method mainly considers two factors for calculating the representativeness of a query q for a topic t_k, which are **popularity** and **concentration**. Specifically, the **popularity** stands for the frequency at which a query q appears in the topic t_k, and the **concentration** represents the relevance of the topic t_k and the query q compared with other topics. The representativeness of a query q for a topic t_k can be derived from

$$r(q, t_k) = \sqrt{pop(q, t_k) \cdot con(q, t_k)}, \tag{14}$$

in which $pop(q, t_k)$ and $con(q, t_k)$ are the popularity and concentration scores of q for t_k. Denote $\mathcal{I}_k$ as the items in the cluster of topic t_k, $pop(q, t_k)$ can be calculated as

$$pop(q, t_k) = \frac{\log tf(q, \mathcal{I}_k) + 1}{\log tf(\mathcal{I}_k)} \tag{15}$$

where $tf(\mathcal{I}_k)$ is the number of tokens in $\mathcal{I}_k$ and $tf(q, \mathcal{I}_k)$ denotes the number of tokens from the items that in the same cluster of topic t_k with query q.

The $con(q, t_k)$ is defined as

$$con(q, t_k) = \frac{\exp(rel(q, D_k))}{1 + \sum_{1 \le j \le K} \exp(rel(q, D_j))} \tag{16}$$

$rel(:, :)$ is the BM25 relevance of two elements, D_k denotes the concatenation of the titles of all items belonging to the same topic t_k.

D. Experiments and Results

SHOAL [22] is Alibaba's current topic-driven taxonomy solution deployed on Taobao platform, which also considers a hierarchical graph-based strategy but only uses a well-defined metric to calculate the query-item embeddings. SHOAL doesn't apply a trainable graph neural network to learn the non-linear interactions between queries and items, which results in the inability to generate more meaningful query-item embeddings. We compare our proposed method with SHOAL in both online and offfline environments.

1) Offline Datasets and Metrics: We design our quantitative experiments to demonstrate whether the hierarchical embeddings which contain the non-linear query-item interactions will help to construct the topic-driven taxonomy that captures users search intention as much as possible. We use a real-world large-scale Taobao dataset, which contains hundreds of millions of items and query-click history in the last seven days, to evaluate the performances of different methods. Table V shows the information of our experimental dataset. Taobao #3 dataset contains query-item click behaviors on Taobao. Over seven days' logs are used for training the neural network and generating the topic-driven taxonomy. Specifically, each existing query-item click log indicates that a user clicked the item from the result of the search query, a query-item edge exists if and only if the corresponding click appears in the logs, and the weight of the edge is the number of the specific query-item clicks.

TABLE V: Statistical Information of Taxonomy Dataset

Dataset	Queries	Items	Q-I Edges	Density
Taobao #3	76,218,663	138,514,439	1,000,947,908	9.481e-8

The purpose of the unsupervised loss function of this graph neural network is to encourage the nearby queries and items to share the similar embeddings, while enforcing the embeddings of disparate users and items are highly distinct. We consider all the existing query-item edges as positive samples, and items without being clicked in the search query results as negative query-item samples. We follow the same ratio of positive samples and negative samples in the e-commerce prediction experiments, which is set to 1:3. Table VI summarizes the statistics of samples in dataset.

TABLE VI: Sample Information of Taxonomy Dataset

Dataset	Positive	Negative	Total
Taobao #3	1,000,947,908	3,002,843,724	4,003,791,632

We report the results of taxonomy to the domain experts to evaluate the performance of all methods. For each taxonomy results, the experts pick 100 topics from the taxonomy and randomly select 100 items under each topic to calculate the accuracy. The larger accuracy, the better result. Furthermore, to demonstrate the hierarchical separating capacity of methods, we use a new metric, the diversity. To define the diversity, we first introduce the term, "qualified topic". Items belonging to a qualified topic should cover more than two different categories. We define diversity as the ratio of the number of qualified topics to the number of all topics discovered by the algorithm. Then larger diversity means better hierarchical separating capacity.

2) Offline Experimental Results: To investigate the model effectiveness, we compare our proposed method with Alibaba's current topic-driven taxonomy solution SHOAL [22]. In the parameter setting, we set the level number of the hierarchical structure $L = 4$ according to the observation of natural ontology level of items in the e-commerce platform. The dimension of all embeddings for both query and item are set to 32, which is the same as the previous experiments. To keep consistency, we set SHOAL's number of clusters as same as HiGNN's.

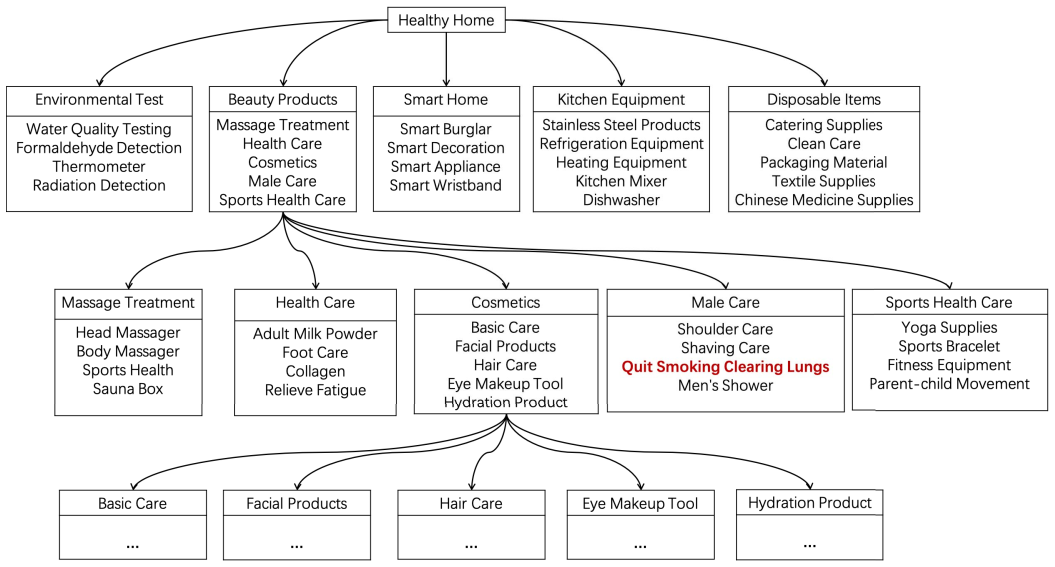
Table VII lists the performance results of SHOAL and HiGNN, in which the number of clusters are user specified and set to be the same for fair comparisons. It is noted that SHOAL's average numbers of final levels is 4.31, which is similar to ours. It is also observed that our proposed method outperforms the Alibaba's current topic-driven taxonomy solution, which is improved by more than 4% of the accuracy. That's because our method not only considers the graph-based non-linear interactions between queries and items but also applies a trainable neural network architecture to learn a hierarchical features which embeds the user search intention successfully. The diversity, our proposed new metric, is also used to evaluate SHOAL and HiGNN. As Table VII shown, HiGNN outperforms Alibaba's existing method by 6% in diversity. Under the same cluster number, HiGNN can discover more qualified topics, which demonstrates its strong hierarchical separating capacity.

TABLE VII: Taxonomy Quality Evaluation

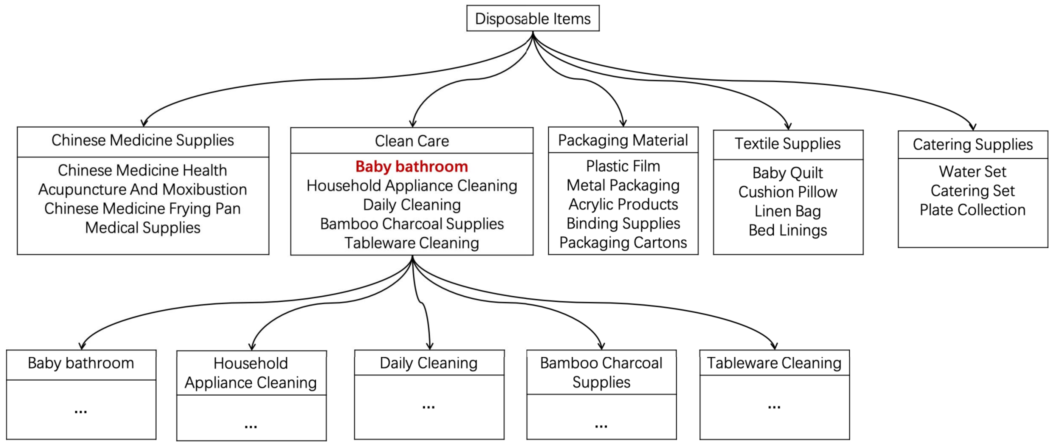
Algorithm	#Level	Accuracy	Diversity
SHOAL	4.31 (on average)	85%	66%
HiGNN	4	89%	70%

3) Offline Case Study: We show a taxonomy case generated by our method from the real-world Taobao dataset in Figure 5. We apply our method to generate a four-level taxonomy, and each parent topic is split into several child topics. Figure 5 shows parts of the taxonomy generated by HiGNN. As shown in Figure 5(a), HiGNN splits the root topic 'Healthy Home' into five sub-topics: 'Environmental Test', 'Beauty Products', 'Smart Home', 'Kitchen Equipment' and 'Disposable Items'. The description of those topics are generated automatically by selecting the term that is most representative for a topic. We find those topics are of good quality and precisely summarize the major topics in our daily home life. In Figure 5(a) and 5(b), we also show how HiGNN splits level two topics 'Beauty Products' and 'Disposable Items' into more fine-grained topics. Taking 'Beauty Products' as an example: (1) at level three, HiGNN can successfully find five child topics in 'Beauty Products': 'Massage Treatment', 'Health Care', 'Cosmetics', 'Male Care', and 'Sports Health Care'; (2) at level four, HiGNN splits the 'Cosmetics' topic into more fine-grained topics: 'Basic Care', 'Facial Products', 'Hair Care', 'Eye Makeup Tool' and 'Hydration Product'. Similarly for the 'Disposable Items' topic (Figure 5(b)), HiGNN can discover level-three topics like 'Chinese Medicine Supplies' and level-four topic like 'Household Appliance Cleaning'. Moreover, the description of child topics for each topic are of good quality and they are semantically coherent and cover different aspects of the same topic. We also find some interesting topics, such as 'Quit Smoking Clean lungs' in 'Male Care' topic and 'Baby bathroom' in 'Clean Care' topic, which are not existed in the taxonomy generated by the compared method.

4) Online Evaluation: In order to verify the effectiveness of our HiGNN method on taxonomy construction tasks, we design an online A/B test with more than 3 million users in a Taobao's real recommendation application. In the control group, the taxonomy structure with the matched recommendations are generated by SHOAL. While in the experiment group,



(a) The sub-topics under the topics 'Healthy Home', 'Beauty Products', and 'Cosmetics'.



(b) The sub-topics under the topics 'Disposable Items' and 'Clean Care'.

Fig. 5: An example of a topic-driven taxonomy generated by HiGNN.

we apply our HiGNN to generate the topic-driven taxonomy structure and based on that we give matching recommendations. The result of the A/B test shows that HiGNN brings a 3.8% improvement in terms of Click Through Rate (CTR) compared to the recommendations generated by SHOAL.

VI. Conclusions

In this paper, we introduce a large-scale Hierarchical Bipartite Graph Neural Network (HiGNN) with the aim of applying to a series of e-commerce scenarios such as user preference prediction, item recommendations, and so on. Although state-of-the-art methods armed with Graph Neural Network (GNN) considerately utilize high-order connections, non-linear interactions, and hierarchical representation on a user-item bipartite graph to bring collaborative filtering signal into fullplay, they are inadequate to display their abilities in large-scale e-commerce applications. However, by stacking multiple GNN modules and using a deterministic clustering algorithm alternately, HiGNN is able to efficiently obtain user and item embeddings simultaneously in a hierarchical fashion, and scalable to large-scale bipartite graphs. To evaluate the performance of HiGNN, we conduct extensive experiments from two perspectives: supervised learning for predicting on user preference, and unsupervised learning for constructing topic-driven taxonomy. The experimental results demonstrate that HiGNN is able to effectively and efficiently obtain the hierarchical structure, and achieve a significant improvement compared against state-of-the-art baselines. Moreover, HiGNN is also deployed into Taobao, one of the largest real-world e-commerce platforms, and is potential to capture increasing attention in various applications.

Acknowledgment

This work was supported by Zhejiang Lab under No. 2019KE0AB01, the National Key RD Program of China under Grant 2018YFB1403202, and the Alibaba-ZJU Joint Research Institute of Frontier Technologies.

References

[1] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1059–1068.

[2] C. Chu, Z. Li, B. Xin, F. Peng, C. Liu, R. Rohs, Q. Luo, and J. Zhou, "Deep graph embedding for ranking optimization in e-commerce," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 2007–2015.

[3] Z. Liu, V. W. Zheng, Z. Zhao, Z. Li, H. Yang, M. Wu, and J. Ying, "Interactive paths embedding for semantic proximity search on heterogeneous graphs," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &#38; Data Mining*, 2018, pp. 1860–1869.

[4] C. Lei, S. Ji, and Z. Li, "Tissa: A time slice self-attention approach for modeling sequential user behaviors," in *The World Wide Web Conference*, 2019, pp. 2964–2970.

[5] J. Bai, C. Zhou, J. Song, X. Qu, W. An, Z. Li, and J. Gao, "Personalized bundle list recommendation," in *The World Wide Web Conference*, 2019, pp. 60–71.

[6] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *The World Wide Web Conference*, 2019, pp. 151–161.

[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 173–182.

[8] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 1025–1035.

[9] T. N. Kipf and M. Welling, in *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[10] T. Lei, W. Jin, R. Barzilay, and T. Jaakkola, "Deriving neural architectures from sequence and graph kernels," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 2017, pp. 2024–2033.

[11] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, 2016, pp. 2014–2023.

[12] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, and P. C. andPhilip S. Yu, "Heterogeneous graph attention network," in *The World Wide Web Conference*, 2019, pp. 2022–2032.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016, pp. 3844–3852.

[14] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proceedings of the 2nd International Conference on Learning*, 2014.

[15] D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, 2015, pp. 2224–2232.

[16] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[17] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "Addgraph: Anomaly detection in dynamic graph using attention-based temporal gcn," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 7 2019, pp. 4419–4425.

[18] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 165–174.

[19] C. Li, K. Jia, D. Shen, C. R. Shi, and H. Yang, "Hierarchical representation learning for bipartite graphs," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 7 2019, pp. 2873–2879.

[20] R. Ying, J. You, C. Morris, X. Ren, W. L. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, 2018, pp. 4805–4815.

[21] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 29–38.

[22] Z. Li, X. Chen, X. Pan, P. Zou, Y. Li, and G. Yu, "Shoal: Large-scale hierarchical taxonomy via graph-based query coalition in e-commerce," *Proc. VLDB Endow.*, vol. 12, no. 12, pp. 1858–1861, Aug. 2019.

[23] H. Dai, B. Dai, and L. Song, "Discriminative embeddings of latent variable models for structured data," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, 2016, pp. 2702–2711.

[24] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 2017, pp. 1263–1272.

[25] K. T. Schütt, P.-J. Kindermans, H. E. Sauceda, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 992–1002.

[26] Y. Li, R. Zemel, M. Brockschmidt, and D. Tarlow, "Gated graph sequence neural networks," in *Proceedings of the 3rd International Conference on Learning*, 2016.

[27] X. He, M. Gao, M.-Y. Kan, and D. Wang, "Birank: Towards ranking on bipartite graphs," *IEEE Trans. on Knowl. and Data Eng.*, vol. 29, no. 1, pp. 57–71, Jan. 2017.

[28] A. N. Nikolakopoulos and G. Karypis, "Recwalk: Nearly uncoupled random walks for top-n recommendation," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 150–158.

[29] J. Yang, C. Chen, C. Wang, and M. Tsai, "Hop-rec: high-order proximity for implicit recommendation," in *Proceedings of the 12th ACM Conference on Recommender Systems*, 2018, pp. 140–144.

[30] H.-F. Yu, C.-J. Hsieh, S. Si, and I. S. Dhillon, "Parallel matrix factorization for recommender systems," *Knowl. Inf. Syst.*, vol. 41, no. 3, pp. 793–819, Dec. 2014.

[31] H.-J. Xue, X.-Y. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 3203–3209.

[32] F. Ye, C. Chen, and Z. Zheng, "Deep autoencoder-like nonnegative matrix factorization for community detection," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1393–1402.

[33] C. Li, Y. Lu, Q. Mei, D. Wang, and S. Pandey, "Click-through prediction for advertising in twitter timeline," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 1959–1968.

[34] S. Weng, H. Tsai, S. Liu, and C. Hsu, "Ontology construction for information classification," *Expert Syst. Appl.*, vol. 31, no. 1, pp. 1–12, 2006.

[35] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proceedings of the 1st International Conference on Learning, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[36] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[37] C. Zhang, F. Tao, X. Chen, J. Shen, M. Jiang, B. M. Sadler, M. Vanni, and J. Han, "Taxogen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2701–2709.