Multiplex Bipartite Network Embedding using Dual Hypergraph Convolutional Networks

Hansheng Xue
The Australian National University
Canberra, Australia
hansheng.xue@anu.edu.au

Luwei Yang
Alibaba Group
Hangzhou, China
luwei.ylw@alibaba-inc.com

Vaibhav Rajan
National University of Singapore
Singapore
vaibhav.rajan@nus.edu.sg

Wen Jiang
Alibaba Group
Hangzhou, China
wen.jiangw@alibaba-inc.com

Yi Wei
Alibaba Group
Hangzhou, China
yi.weiy@alibaba-inc.com

Yu Lin*
The Australian National University
Canberra, Australia
yu.lin@anu.edu.au

ABSTRACT

A bipartite network is a graph structure where nodes are from two distinct domains and only inter-domain interactions exist as edges. A large number of network embedding methods exist to learn vectorial node representations from general graphs with both homogeneous and heterogeneous node and edge types, including some that can specifically model the distinct properties of bipartite networks. However, these methods are inadequate to model multiplex bipartite networks (e.g., in e-commerce), that have multiple types of interactions (e.g., click, inquiry, and buy) and node attributes. Most real-world multiplex bipartite networks are also sparse and have imbalanced node distributions that are challenging to model. In this paper, we develop an unsupervised **Dual HyperGraph Convolutional Network** (**DualHGCN**) model that scalably transforms the multiplex bipartite network into two sets of homogeneous hypergraphs and uses spectral hypergraph convolutional operators, along with intra- and inter-message passing strategies to promote information exchange within and across domains, to learn effective node embeddings. We benchmark DualHGCN using four real-world datasets on link prediction and node classification tasks. Our extensive experiments demonstrate that DualHGCN significantly outperforms state-of-the-art methods, and is robust to varying sparsity levels and imbalanced node distributions.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**.

KEYWORDS

Network Embedding, Multiplex Bipartite Network, Hypergraph

ACM Reference Format:
Hansheng Xue, Luwei Yang, Vaibhav Rajan, Wen Jiang, Yi Wei, and Yu Lin. 2021. Multiplex Bipartite Network Embedding using Dual Hypergraph Convolutional Networks. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3449954

*Corresponding author.

1 INTRODUCTION

Network representation learning aims to learn low-dimensional real-valued features of its nodes, also called embeddings, to capture the global structural information of the network [3, 7]. Such vectorial representations enable their direct application in machine learning models for tasks such as link prediction, node classification or community detection, and obviates the need for cumbersome task-specific feature engineering from the input networks. They have been successfully applied in many domains such as recommender systems [17, 26, 43], natural language processing [18, 30, 33] and computational biology [23, 27, 46].

Many network embedding methods have been proposed for homogeneous networks where nodes and edges are both of single type; well-known examples include Node2vec [14], DeepWalk [25], SDNE [37] and LINE [31]. Many real-world interactions are multimodal and multi-typed that give rise to heterogeneous networks where nodes and/or edges can be of different types. Representation learning methods for such networks have also been widely studied, e.g., Metapath2vec [8], HAN [39], HetGNN [44].

The bipartite network has a specific topology, consisting of two node types (see Figure 1) from different domains, containing inter-domain interactions and no intra-domain interactions. Essentially representing matrices, such networks are ubiquitous in a variety of contexts. While general representation learning methods can be applied on such networks, it has been shown that they yield suboptimal representations because many specific characteristics of bipartite networks, such as the two distinct node types and the power-law distribution of node degrees may not be modeled well. As a result, several representation learning methods have been developed specifically for bipartite networks, e.g., BiNE [11], BGNN [16], BiANE [19], FOBE and HOBE [29]. However, these methods do not model heterogeneous interactions or multiple edges types in bipartite networks. Such networks, also called multiplex bipartite networks, model many real-life scenarios. For example, users and items in an e-commerce platform form a multiplex bipartite network where users have different kinds of interactions (click, inquiry, buy) with items.

The fundamental challenge in any network representation learning method is to learn the similarities or correlations between nodes,

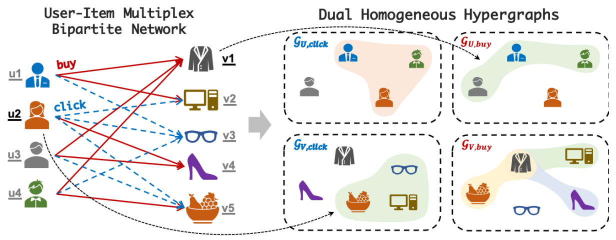

Figure 1: (left) User-item multiplex bipartite network and (right) dual homogeneous hypergraphs. E.g., the user u_2 in the user-item multiplex bipartite network corresponds to a hyperedge that connects v_2, v_3 and v_5 in the homogeneous hypergraph $\mathcal{G}_{V,click}$ because these three items have been clicked by the same user. Similarly, an item v_1 corresponds to a hyperedge that connects u_1, u_3 and u_4 in the homogeneous hypergraph $\mathcal{G}_{U,buy}$ because these three users buy the same item.

from all the given information about the topology, multiple node and edge types and, if provided, the attributes; and preserve the correlations at the latent level in the embeddings [6]. In fact, various network embedding techniques are equivalent to factorization of a node similarity matrix with suitable definitions of similarities [22] or tensor factorization [42]. In bipartite networks, edges provide information about inter-domain node correlations only, while one has to learn intra-domain correlations indirectly. When node attributes are given, attribute and topology based correlations, representing two different modalities, have to be learnt jointly [19]. With the addition of multiple edge types in a multiplex bipartite network, we have more information to model the correlations but generalizing the node similarities using heterogeneous edges with potentially distinct distributional and structural properties can be challenging.

The problem is exacerbated by sparsity of edges and imbalance of distributions of node and edge types in most real-world data. For instance, consider the Alibaba dataset containing user behavior logs from Alibaba.com (more data details are in Section 5.1). There are 6,054 users and 16,595 items and the average degree of users and items are 7.55 and 2.76 respectively. Each user, on average, interacts with less than 0.1% items. Figure 10 shows the clearly distinguishable degree distributions of users and items, with the item distribution having a steeper decline. Further, each edge type can be present in different proportions and sparsity levels, e.g. varying from 25,180 ‘click’ edges to 4,429 ‘contact’ edges (Figure 11.a).

In this paper, we address these challenges by designing a new representation learning model for multiplex bipartite networks. A key step in our approach is to transform the input into two sets of hypergraphs, a set each for a domain in the bipartite network and a hypergraph for each edge type within a set. The transformation is scalable since the total number of edges in the hypergraphs is proportional to the number of nodes in the input and to the number of edge types. A hypergraph generalizes the notion of an edge in simple graphs to a hyperedge which can connect more than two nodes. This transformation effectively serves many purposes. It naturally models sparse and heterogeneous interactions in the input, e.g., in the e-commerce network, multiple items naturally form a hyperedge with a user if they are bought (clicked, or inquired) by the same user, and similarly, multiple users can be connected by a hyperedge to an item (see Figure 1). The homogeneity in these hypergraphs allows us to leverage hypergraph convolutional operators to learn rich representations, capturing local and higher-order structural relationships. However, this alone is not sufficient to capture inter- and intra-domain correlations in the bipartite network. To model these correlations and tackle the imbalance problem in both edge and node types, we design additional intra- and inter-message passing strategies that enable information exchange within and across domains. Further, our method can also incorporate information from node attributes when provided as inputs.

Our model, called **Dual HyperGraph Convolutional Networks** (**DualHGCN**), is evaluated through extensive experiments. On node classification and link prediction tasks, DualHGCN significantly outperforms fourteen state-of-the-art network embedding methods on four real datasets. Our experiments also demonstrate the superiority of our model with respect to robustness to varying sparsity levels, node attribute initialization strategies and handling of imbalanced classes.

2 RELATED WORKS

Homogeneous Network Embedding. Homogeneous networks contain a single type of nodes and edges, and thus the sum of node types and edge types is equal to 2. Many approaches have been proposed for homogeneous network embedding methods such as DeepWalk [25], Node2vec [14], LINE [31], SDNE [37], GCN [20], GraphSAGE [15] and GAT [36]. However, these methods do not explicitly model bipartite structure and multiple edge types.

Heterogeneous Network Embedding. A network is called heterogeneous if the sum of node types and edge types is larger than 2. Although multiplex bipartite networks can be viewed as special cases of heterogeneous networks, existing heterogeneous network embedding methods (e.g., Metapath2vec [8], HAN [39], HetGNN [44], and DyHATR [40]) are not tailored to make use of the bipartite topology information and may result in sub-optimal embedding for multiplex bipartite networks. For example, Metapath2vec [8] uses the meta-path-guided random walk strategy but ignores the difference between explicit and implicit relations and thus becomes suboptimal for network embedding for bipartite networks [11]. Similarly, the node-level and edge-level attention models in DyHATR [40] neglect the unique characteristics of the bipartite network, and also do not work well with increasing sparsity.

Bipartite Network Embedding. Different from multiplex bipartite networks, simple bipartite networks contain two types of nodes and a single type of edges. Several bipartite network embedding methods have been proposed, including BiNE [11, 12], BGNN [16], BiANE [19], FOBE and HOBE [29]. BiNE first performs biased random walks to generate node sequences and then uses a joint optimization strategy to preserve both explicit and implicit information within bipartite networks simultaneously. As a random walk-based approach, the performance of BiNE deteriorates when the bipartite network becomes sparse. Moreover, BiNE neglects the inherent difference between two types of nodes and models all nodes in the same way. BGNN respects the distinction between two types of nodes and proposes a cascaded and unsupervised learning method, which contains inter-domain message passing and intra-domain distribution alignment, to model both same-domain information and cross-domain correlations simultaneously. FOBE and HOBE also distinguish two types of nodes and fit embeddings by optimizing nodes of each type separately. They adopt two sampling strategies to generate indirect node-pair sets, including sampling direct and observed pairs (FOBE) and sampling higher-order pairs using algebraic distance (HOBE). BiANE is an attributed bipartite network embedding method that differs from the previous three models. It models structural information of the bipartite network through intra- and inter-partition proximity, and integrates attributes and topological structure of networks by a latent correlation model.

All these existing methods have been designed for bipartite networks where all edges are of the same type and their performance on multiplex bipartite networks suffer without explicit modeling of heterogeneous edge types (as seen in our experiments). Besides, BiNE, FOBE and HOBE cannot capture the inherent attributes of nodes in the bipartite network.

Hypergraph Embedding. Hypergraph embedding is gaining popularity because of its effectiveness in modeling complex structures within networks. A hypergraph is a generalization of a simple graph in which an hyperedge can connect more than two nodes. HyperGCN [41] decomposes each hyperedge into a collection of node pairs and translates the hypergraph learning tasks into the embedding problem on simple graphs. Several homogeneous hypergraph embedding methods have been proposed. HGNN [10], HyperRec [38] and HCHA [1] introduce a spectral convolution operator into the hypergraph learning model and capture higher-order structures in hypergraphs. MGCN [5] considers both local and hypergraph level graph convolutions and is able to capture wider and richer network information for network embedding. Different from previous approaches, HNHN [9] introduces a flexible normalization scheme and a hypergraph convolutional model with nonlinear activation neurons on both hypernodes and hyperedges. Three recent approaches have been proposed to model heterogeneous hypergraphs, e.g., DHNE [34], Hyper-SAGNN [45], and HWNN [28]. However, most previous hypergraph embedding methods have been designed for supervised tasks, and cannot be directly used for obtaining network embeddings. Besides, hypergraph convolution networks do not specifically model multiplex edges and imbalanced degrees.

3 PRELIMINARIES

In this section, we define a multiplex bipartite network and its vertex embedding. Table 1 provides a summary of frequently used symbols in the paper.

Definition 2.1. Multiplex Bipartite Network. A Multiplex Bipartite Network $G = (U, V, E, X)$ consists of node sets U and V of different types, and edge sets $E = E^1 \cup E^2 \ldots \cup E^k$ where E^i denotes the i-th type of edge. $X = \{X_u, X_v\}$ denotes the features of node sets U and V.

For example, logs of user behavior in Alibaba.com can be represented as a multiplex bipartite network containing two types of nodes (users and items) and several types of edges (e.g., click, enquiry, contact). For each user and item, the logs also contain unique attributes with different dimensions. For instance, attributes of users contain country, gender, search logs, etc. In contrast, Items usually have attributes, such as category, price, search counts, visit counts, buy logs, etc.

Table 1: Summary of notations and descriptions.

Notations	Descriptions
$G = (U, V, E, X)$	Multiplex Bipartite Network (MBN)
U, V	Two types of node sets in MBN
$E = E^1 \cup \ldots \cup E^k$	A set of edges in MBN
$X = \{X_u, X_v\}$	Two feature sets of U and V in MBN
k	The number of edge types
$\mathbf{G}_U, \mathbf{G}_V$	Two homogeneous hypergraph sets
$\mathcal{G}_{U,i}, \mathcal{G}_{V,j}$	A homo-hypergraph from U, V
$\mathbf{H}$	The incidence matrix of hypergraph
$\mathbf{D}$	The diagonal matrices of node degree
$\mathbf{B}$	The diagonal matrices of hyperedge degree
$\mathbf{W}$	The diagonal identity matrix
$\mathbf{X}_{U,i}, \mathbf{X}_{V,j}$	The learned features of $\mathcal{G}_{U,i}, \mathcal{G}_{V,j}$
$\mathbf{P}, \mathbf{Q}$	Learnable weight matrix
$\sigma(\cdot)$	Activation function
$\mathbf{Z} = \{\overline{\mathbf{X}}_U, \overline{\mathbf{X}}_V\}$	Final embeddings of $\mathbf{G}_U, \mathbf{G}_V$
n	Negative sample parameter
d	Dimension of the embedding

Definition 2.2. Multiplex Bipartite Network Embedding. Given a multiplex bipartite network $G = (U, V, E, X)$, its embedding is a d-dimensional feature $\overline{\mathbf{X}}_U \in \mathbb{R}^{|U|\times d}$, $\overline{\mathbf{X}}_V \in \mathbb{R}^{|V|\times d}$ for each node in U and V, where $d \ll |U|$ and $d \ll |V|$, that captures information of both the global topological structure and node attributes.

We define the sparsity of a multiplex bipartite network as $\langle \mathcal{S} \rangle = 1 - \frac{|E|}{|U|\times|V|}$. Many real-world multiplex bipartite networks are extremely sparse, i.e., $|E| \ll |U| \times |V|$. For instance, in the Alibaba dataset $\langle \mathcal{S} \rangle = 99.95\%$.

4 METHODOLOGY

Given an input multiplex bipartite network, we first transform it into two sets of homogeneous hypergraphs. Our model architecture comprises a hypergraph convolutional network that assumes these dual homogenous hypergraphs as inputs, with additional inter- and intra-message passing layers to enable information sharing across the networks. Finally, the entire model is trained using a gradient descent based optimizer. The next four subsections provide more details. Figure 2 shows an overview of the entire method.

4.1 Dual Homo-Hypergraphs Construction

We now show how to transform a multiplex bipartite network into two sets of homogeneous hypergraphs (dual homo-hypergraphs). We construct two sets of homogeneous hypergraphs $\mathbf{G}_U, \mathbf{G}_V$, from node sets U, V, respectively, as follows:

$$\mathbf{G}_U = \{\mathcal{G}_{U,base}, \mathcal{G}_{U,1}, ..., \mathcal{G}_{U,k}\}, \mathbf{G}_V = \{\mathcal{G}_{V,base}, \mathcal{G}_{V,1}, ..., \mathcal{G}_{V,k})\}, \tag{1}$$

where $\mathcal{G}_{U,i} = \{U, \mathcal{E}_{U,i}\}$, $\mathcal{G}_{V,j} = \{V, \mathcal{E}_{V,j}\}$, and $\mathcal{E}_{U,i}$ and $\mathcal{E}_{V,j}$ denote hyperedges in $\mathcal{G}_{U,i}$ and $\mathcal{G}_{V,j}$ respectively. Note that all

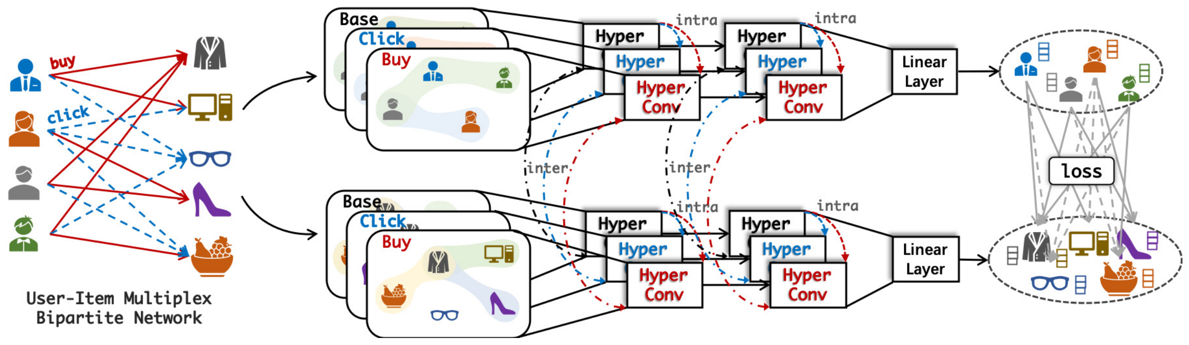

Figure 2: The Overall framework of the proposed DualHGCN method.

the homogeneous hypergraphs in $\mathbf{G}_U$ share the same node set U while all the homogeneous hypergraphs in $\mathbf{G}_V$ share the same node set V. For a node $v \in V$, a hyperedge is introduced in $\mathcal{E}_{U,i}$ of $\mathcal{G}_{U,i}$ which connects to $\{u | u \in U, (u,v) \in E^i\}$, i.e., the vertices in U that are directly connected to v by E^i. Similarly, for a node $u \in U$, a hyperedge is introduced in $\mathcal{E}_{V,j}$ of $\mathcal{G}_{V,j}$ which connects to $\{v | v \in V, (u,v) \in E^j\}$, i.e., the vertices in V that are directly connected to u by E^j.

Refer to Figure 1 for an example. In the user-item multiplex bipartite network, the user u_2 clicks three items (v_2, v_3 and v_5), which corresponds to a hyperedge that connects these three items in the homogeneous hypergraph $\mathcal{G}_{V,click}$. Similarly, the item v_1 is bought by three users (u_1, u_3 and u_4) which corresponds to a hyperedge that connects these three users in the homogeneous hypergraph $\mathcal{G}_{U,buy}$.

Two special homogeneous hypergraphs $\mathcal{G}_{U,base} \in \mathbf{G}_U$ and $\mathcal{G}_{V,base} \in \mathbf{G}_V$ are defined as $\mathcal{G}(U, \bigcup_{i=1}^{k} \mathcal{E}_{U,i})$ and $\mathcal{G}(V, \bigcup_{j=1}^{k} \mathcal{E}_{V,j})$, respectively. Note that the cardinalities of hyperedge sets in the constructed hypergraphs are: $|\mathcal{E}_{U,i}| \leq |V|$, $|\mathcal{E}_{V,j}| \leq |U|$, $|\mathcal{E}_{U,base}| \leq k|V|$ and $|\mathcal{E}_{V,base}| \leq k|U|$ for $1 \leq i,j \leq k$. The total number of hyperedges in the dual homo-hypergraphs is proportional to the number of nodes and edge types in the input network: $O(k(|U| + |V|))$. Thus, the transformation easily scales to large inputs.

4.2 Hypergraph Convolutional Networks

Note that the hypergraphs that we constructed from a multiplex bipartite network are homogeneous and now we can apply hypergraph convolutions on them to learn representations. Graph convolutional network [20] has been widely used in modeling simple networks. Recent hypergraph convolutional operators have borrowed ideas from the spectral theory on simple graphs and achieved good performance in hypergraph embedding (e.g., HGNN, HCHA, HyperGCN and MGCN). We briefly describe two classical hypergraph convolutional operators used in our model.

Simple graphs use the adjacency matrix A to represent edges, whereas hypergraphs introduce the incidence matrix H to describe the relationship between nodes and hyperedges. Given a homo-hypergraph $\mathcal{G}_{U,i} = (U, \mathcal{E}_{U,i})$ where $i \in \{base, 1, \ldots, k\}$, k is the number of edge types, the incidence matrix of $\mathcal{G}_{U,i}$ is defined as:

$$\mathbf{H}_{U,i}(u,e) = \begin{cases} 1, & \text{if } u \text{ is incident to } e, e \in \mathcal{E}_{U,i} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $\mathcal{E}_{U,i}$ denotes the set of hyperedges in $\mathcal{G}_{U,i}$, $\mathbf{H}_{U,i} \in \mathbb{R}^{|U| \times |\mathcal{E}_{U,i}|}$, and $i \in \{base, 1, \ldots, k\}$ denotes the constructed homo-hypergraph i. Similarly we define the incidence matrix $\mathbf{H}_{V,j}$ for $\mathcal{G}_{V,j}$. Let $\mathbf{D}_{U,i} \in \mathbb{R}^{|U| \times |U|}$ and $\mathbf{B}_{U,i} \in \mathbb{R}^{|\mathcal{E}_{U,i}| \times |\mathcal{E}_{U,i}|}$ denote diagonal matrices of the node degree and the hyperedge degree respectively, where $\mathbf{D}_{U,i}(u,u) = \sum_{e \in \mathcal{E}_{U,i}} \mathbf{H}_{U,i}(u,e)$ and $\mathbf{B}_{U,i}(e,e) = \sum_{u \in U} \mathbf{H}_{U,i}(u,e)$.

Two hypergraph spectral convolutional operators, symmetric hypergraph convolution (sym) and the asymmetric hypergraph convolution (asym), are used to learn embeddings of each hypergraph in our model. For a simple graph, the convolutional operator can be formulated as $X^{l+1} = \sigma(A \cdot X^l P^l)$, where X is the feature matrix, A is the adjacency matrix and P is the learnable weight matrix. Because the incidence matrix $\mathbf{H}$ denotes the relationship between nodes and hyperedges, we use $\mathbf{HWH}^\top$ to measure the pairwise relationships between nodes in the same homogeneous hypergraph, where $\mathbf{W}$ is the weight matrix that assigns weights for all hyperedges. Usually, we initialize the weight matrix $\mathbf{W}$ with the identity matrix yielding equal weights for all hyperedges. Thus, the intuitive hypergraph convolutional operator can be formulated as:

$$\mathbf{X}^{l+1} = \sigma(\mathbf{HWH}^\top \cdot \mathbf{X}^l \mathbf{P}^l) \tag{3}$$

However, the previous hypergraph convolutional operator may change the scale of the feature vectors $\mathbf{X}$ by adding layers of convolutional operators (multiplication with $\mathbf{HWH}^\top$). To constrain the number of parameters and decrease the number of matrix multiplications, and thereby avoid the overfitting problem, GCN introduces a renormalization trick, $X^{l+1} = \sigma((I + D^{-1/2}AD^{-1/2}) \cdot X^l P^l) = \sigma(\widetilde{D}^{-1/2}\widetilde{A}\widetilde{D}^{-1/2} \cdot X^l P^l)$, where $\widetilde{A} = A + I$, $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$, I is the identity matrix and D is the node degree matrix of a simple graph. Similarly, the symmetric normalization version of hypergraph convolutional operator for $\mathcal{G}_{U,i}$ can be defined as:

$$\mathbf{X}_{U,i}^{l+1} = \sigma(\mathbf{D}_{U,i}^{-1/2}\mathbf{H}_{U,i}\mathbf{W}_U\mathbf{B}_{U,i}^{-1}\mathbf{H}_{U,i}^\top\mathbf{D}_{U,i}^{-1/2} \cdot \mathbf{X}_{U,i}^l\mathbf{P}_{U,i}^l), \tag{4}$$

and, the asymmetric hypergraph convolutional operator for $\mathcal{G}_{U,i}$ can be defined as:

$$\mathbf{X}_{U,i}^{l+1} = \sigma(\mathbf{D}_{U,i}^{-1}\mathbf{H}_{U,i}\mathbf{W}_U\mathbf{B}_{U,i}^{-1}\mathbf{H}_{U,i}^\top \cdot \mathbf{X}_{U,i}^l\mathbf{P}_{U,i}^l), \tag{5}$$

where σ denotes the nonlinear activation function (i.e., ReLU function in our model), $\mathbf{X}_{U,i}^{l} \in \mathbb{R}^{|U| \times d_l}$ is the feature of the l-th layer, $\mathbf{W}_U \in \mathbb{R}^{|V| \times |V|}$ is the identity matrix, and $\mathbf{P}_{U,i}^{l} \in \mathbb{R}^{d_l \times d_{l+1}}$ denotes the learnable filter matrix, d_l and d_{l+1} are the dimensions of the l-th and $(l+1)$-th layers respectively.

Similar hypergraph convolutional operators are also applied to learn features from $\mathcal{G}_{V,i}$. Therefore, for dual homo-hypergraphs $\mathcal{G}$, we can learn features from each homo-hypergraph ($\mathcal{G}_{U,i}$ or $\mathcal{G}_{V,j}$) independently through the above hypergraph convolutional operators (Eqs. 4 and 5). Thus, we obtain the low-dimensional node representations $\{\mathbf{X}_{U,base}, \mathbf{X}_{U,1}, ..., \mathbf{X}_{U,k}\}$ and $\{\mathbf{X}_{V,base}, \mathbf{X}_{V,1}, ..., \mathbf{X}_{V,k}\}$.

These hypergraph convolutional operators can model each homo-hypergraph, but cannot handle the problem of multiplex edges and topological imbalance. Thus, as described in the following section, we add new layers to enable intra- and inter- message-passing.

4.3 Message-passing Strategies

In the previous section, the multiplex bipartite network is transformed into independent hypergraphs ($\mathcal{G}_{U,i}$ or $\mathcal{G}_{V,j}$) that correspond to each edge type. There may be information loss with respect to each node in the hypergraphs because correlations between different edge types have not been modeled. Take the e-commerce platform for an example, a user is more likely to 'buy' an item after this user 'inquiries' this item or similar items, but the current embeddings consider 'buy' and 'inquiry' independently as they are two different edge types between user and item. Therefore, we introduce the following intra-message passing strategy to promote information sharing among $\{\mathbf{X}_{U,base}, \mathbf{X}_{U,1}, ..., \mathbf{X}_{U,k}\}$ and among $\{\mathbf{X}_{V,base}, \mathbf{X}_{V,1}, ..., \mathbf{X}_{V,k}\}$.

Intra-message passing. As $\mathcal{G}_{U,base}$ contains information aggregated from all $\mathcal{G}_{U,i}$, we incorporate information from the learned $\mathbf{X}_{U,base}$ into each $\mathbf{X}_{U,i}$. The iterative formula of the intra-message passing strategy (from l-th layer to $(l+1)$-st layer) is defined as:

$$\mathbf{X}_{U,i}^{l+1} = \sigma(\Theta_{U,i} \cdot \mathbf{X}_{U,i}^{l}\mathbf{P}_{U,i}^{l} + \mathbf{X}_{U,base}^{l}\mathbf{Q}_{U,base}^{l}) \tag{6}$$

where $\Theta_{U,i} = \mathbf{D}_{U,i}^{-1/2}\mathbf{H}_{U,i}\mathbf{W}_U\mathbf{B}_{U,i}^{-1}\mathbf{H}_{U,i}^{\top}\mathbf{D}_{U,i}^{-1/2}$ for symmetric convolutional operators or $\Theta_{U,i} = \mathbf{D}_{U,i}^{-1}\mathbf{H}_{U,i}\mathbf{W}_U\mathbf{B}_{U,i}^{-1}\mathbf{H}_{U,i}^{\top}$ for asymmetric convolutional operators (from Eqns 4 and 5) and $\mathbf{Q}_{U,base}^{l}$ denotes the learnable transform matrix.

As discussed in Section 4.1, when a multiplex bipartite network is transformed into homogeneous hypergraphs, a node u in U corresponds to a hyperedge $e \in \mathcal{E}_{V,j}$ in the hypergraph $\mathcal{G}_{V,j}$. However, the hypergraph convolutional operators in the previous section neglects the above correspondence between nodes and hyperedges and thus may result in suboptimal embeddings. Therefore, we introduce the following inter-message passing to reinforce similar properties between $\mathbf{X}_{U,i}$ and $\mathbf{X}_{V,i}$ with respect to the same i-th edge type.

Inter-message passing. We propose an inter-message passing strategy which fuses the features $\mathbf{X}_{U,i}$ and $\mathbf{X}_{V,j}$ (from l-th layer to $l+1$-th layer) and is given by:

$$\mathbf{X}_{U,i}^{l+1} = \sigma(\Theta_{U,i} \cdot \mathbf{X}_{U,i}^{l}\mathbf{P}_{U,i}^{l} + \mathbf{H}_{V,i}^{\top}\mathbf{X}_{V,i}^{l}\mathbf{Q}_{V,i}^{l}) \tag{7}$$

$$\mathbf{X}_{V,j}^{l+1} = \sigma(\Theta_{V,j} \cdot \mathbf{X}_{V,j}^{l}\mathbf{P}_{V,j}^{l} + \mathbf{H}_{U,j}^{\top}\mathbf{X}_{U,j}^{l}\mathbf{Q}_{U,j}^{l}) \tag{8}$$

where $\mathbf{H}_{U,i}$ and $\mathbf{H}_{V,j}$ denote the incidence matrix, $\mathbf{Q}_{U,i}^{l}$ and $\mathbf{Q}_{V,j}^{l}$ denote the learnable transform matrix. After t iterations, the embeddings of nodes U and V can be formulated as $\mathbf{X}_U^t = \{\mathbf{X}_{U,base}^t, \mathbf{X}_{U,1}^t, ..., \mathbf{X}_{U,k}^t\}$, $\mathbf{X}_{U,i}^t \in \mathbb{R}^{|U| \times d_t}$, and $\mathbf{X}_V^t = \{\mathbf{X}_{V,base}^t, \mathbf{X}_{V,1}^t, ..., \mathbf{X}_{V,k}^t\}$, $\mathbf{X}_V^t \in \mathbb{R}^{|V| \times d_t}$, where d_t is the dimension of final embeddings and k is the number of edge types. Then, we concatenate these learned features after t layers training for two types of nodes, $\mathbf{X}_U^t$ and $\mathbf{X}_V^t$, and pass them to a linear layer to obtain the final embeddings:

$$\overline{\mathbf{X}}_U = \mathbf{X}_U^t \cdot \mathbf{W}_U + b_U, \quad \overline{\mathbf{X}}_V = \mathbf{X}_V^t \cdot \mathbf{W}_V + b_V, \tag{9}$$

where $\mathbf{W}_U, \mathbf{W}_V \in \mathbb{R}^{((k+1)*d_t) \times d_t}$, and $b_U, b_V \in \mathbb{R}^{d_t}$ are trainable parameters, and $\overline{\mathbf{X}}_U \in \mathbb{R}^{|U| \times d_t}$, $\overline{\mathbf{X}}_V \in \mathbb{R}^{|V| \times d_t}$. Thus, after training the network, we can obtain final embeddings: $\mathbf{Z} = \{\overline{\mathbf{X}}_U, \overline{\mathbf{X}}_V\}$.

4.4 Optimization

To learn the weights of DualHGCN, we maximize the probability of positive edges (existing edges in the multiplex bipartite network) and minimize the probability of negative ones (unseen edges):

$$L = \sum_{(u,v) \in E} \Bigg[\lambda \cdot \log \sigma(\mathbf{Z}_u^{\top}\mathbf{Z}_v) + (1-\lambda) \cdot \sum_{i=1}^{n} \Big(\mathbb{E}_{u_i \sim P(u)} \log(1 - \sigma(\mathbf{Z}_u^{\top}\mathbf{Z}_{u_i})) + \mathbb{E}_{v_i \sim P(v)} \log(1 - \sigma(\mathbf{Z}_v^{\top}\mathbf{Z}_{v_i})) \Big) \Bigg] \tag{10}$$

where σ is the sigmoid activation function, λ denotes the weight to balance the importance between positive and negative samples, $P(u)$ denotes the negative candidate nodes distribution of u, and n is the number of the negative samples. The existing edges in the multiplex bipartite network are treated as positive samples. For each positive pairwise edge (u, v), we randomly sample n negative edges incident to node u and v, respectively. The pseudocode for DualHGCN is shown in Algorithm 1.

Algorithm 1: The DualHGCN algorithm.

```
Input: Multiplex bipartite network $G = (U, V, E, X)$,
       number of iterations $t$, number of negative samples
       per positive sample $n$, the initial features $X$, initial
       parameters;
Output: Node Embedding $\mathbf{Z}$
1  Model Construction:
2  Construct dual homo-hypergraphs $\mathcal{G}$;
3  Add spectral convolutional layers on Base
     homo-hypergraph (Eq. (4) or (5)), $\mathbf{X}_{U,base}^i$, $\mathbf{X}_{V,base}^i$;
4  for each edge-type $j \in [1, 2, ..., k]$ do
5      Add intra and inter-message passing layers (Eq. (6), (7)
         and (8)), $\mathbf{X}_{U,j}^i$, $\mathbf{X}_{V,j}^i$;
6  end
7  Add linear layer to outputs of hypergraph convolutions;
8  Optimization:
9  Initialize Embeddings $\mathbf{Z}$ with initial features;
10 Randomly sample $n$ negative edges for each positive edge;
11 Optimize loss (Eq. (10)) via gradient descent ($t$ iterations);
12 return $\mathbf{Z}$;
```

5 EXPERIMENTS

We benchmark our proposed model with several baselines to validate the effectiveness of DualHGCN for unsupervised multiplex bipartite network representation learning. Specifically, we investigate the following questions in these carefully designed experiments:

Q1 How does DualHGCN perform in predicting unknown interactions or user behaviors (i.e., the link prediction task)?

Q2 How does DualHGCN perform in classifying items according to user behaviors (i.e., the node classification task)?

Q3 How do the inter- and intra-message passing strategy contribute to final unsupervised embeddings of DualHGCN?

Q4 How do the multiple types of edges and the sparsity of networks affect the performance of DualHGCN?

Q5 How sensitive is the performance of DualHGCN to its parameter settings?

5.1 Datasets

We use four real-world datasets in our experiments. Their detailed statistics are given in Table 2.

DTI.[1] This Drug-Target Interactions bipartite network was randomly sampled from the data in the Drug Target Commons platform [32]. The sampled dataset mainly contains two types of nodes (drugs and targets) and five bio-activities (Potency, IC50, KI, Inhibition, and Activity) that form the edges.

Amazon.[2] This dataset is a heterogeneous non-bipartite network [4]. We follow the strategy in BGNN [16] to process this dataset to derive a multiplex bipartite network. This multiplex bipartite network contains two types of edges and two types of nodes, where the attributes of nodes include the price, sales-rank, brand, and category.

Alibaba-s and **Alibaba.**[3] These real-world datasets consist of behavior logs of users and items collected from the e-commerce platform Alibaba.com from 1st April 2020 to 30th April 2020. It contains two types of nodes (users and items) and three types of activities (click, enquiry and contact). The items are classified into five categories (women's clothing, men's clothing, etc.). Alibaba-s is a smaller unattributed dataset, and Alibaba is a multiplex bipartite network where users have attributes such as country, gender and search logs, and items have attributes including the category, price, search counts, visit counts, buy logs, etc. We have anonymized all sensitive data, e.g., user and item id, in both Alibaba-s and Alibaba datasets.

5.2 Baselines

We compare DualHGCN with fourteen state-of-the-art algorithms in four categories as listed below.

1) Simple Homogeneous Network Embedding. These homogeneous network embedding methods ignore the both node-type and edge-type information in the input multiplex bipartite network. They also do not use hypergraphs to generate low-dimensional representations for each node.

- Node2vec [14] uses a biased random walk procedure and extends the skip-gram model.

[1]https://drugtargetcommons.fimm.fi
[2]http://jmcauley.ucsd.edu/data/amazon
[3]https://www.alibaba.com

Table 2: Statistics of four real-world datasets. The sparsity of the multiplex bipartite network: $\langle \mathcal{S} \rangle = 1 - \frac{|E|}{|U| \times |V|}$.

Datasets		DTI	Amazon	Alibaba-s	Alibaba
#Nodes	U	3,270	3,781	1,869	6,054
	V	1,567	5,749	13,349	16,595
#Edges		16,458	60,658	27,036	45,734
#Edge Types		5	2	3	3
#Features	U	N/A	4	N/A	7
	V				11
#Classes	V	N/A	N/A	5	5
$\#\langle \mathcal{S} \rangle$		99.68%	99.72%	99.89%	99.95%

- GraphSAGE [15] is an inductive network embedding method which contains several message aggregation strategies to generate features for previously unobserved nodes.
- GCN [20] proposes a spectral graph convolutional operator to learn both local network structure and features of nodes.
- GAT [36] uses masked self-attention mechanism to assign different neighbors with different specified weights.

2) Hypergraph Embedding. For these methods, we use the same strategy mentioned in Section 4.1 to build dual 'base' homo-hypergraphs. They also ignore edge-type information in the inputs.

- HGNN [10]: generalizes the spectral convolutional networks to capture high-order structural information.
- HyperGCN [41]: decomposes hyperedges of its hypergraphs into a set of node pairs and then uses a simple graph convolutional network to learn decomposed node pairs.
- HCHA [1]: uses a spectral convolutional and attention-based method to model multi-hop relationships.
- MGCN [5]: generalizes from simple graph convolutional networks without using spectral convolutions.

3) Heterogeneous Network Embedding. These methods can model multiple node and edge types but do not explicitly model the bipartite structure of the input multiplex bipartite network.

- Metapath2vec++ [8]: generates meta-path-based random walks on which a heterogeneous skip-gram model is trained.
- HAT [40]: is a hierarchical attention based heterogeneous network embedding method which uses node-level and edge-level attention to model multiple edges types.

4) Bipartite Network Embedding. These methods are applied on the 'base' bipartite networks constructed in Section 4.1 to derive the final node embeddings.

- BiNE [11]: generates biased random walks and then optimizes to preserve both the explicit and implicit relationships within the bipartite network.
- BGNN [16]: a cascaded and unsupervised embedding method with a communication strategy between the domains to distinguish between the two types of nodes and promote information sharing across two domains simultaneously.
- BiANE [19]: an attributed bipartite network embedding method which can model the intra- and inter-partition proximity simultaneously and uses a latent correlation training approach to jointly learn attribute and structure information.

As another baseline just the initial features are used, i.e., without any network embedding methods. In datasets where node attributes are available, the attributes are used as initial features and in datasets without node attributes, we use a tied autoencoder [2] (where weights across the encoder and decoder are tied) on the adjacency matrix of the multiplex bipartite network to generate initial features.

5.3 Experimental Settings

Link Prediction. We randomly sample 50% of the edges as the training set and the remaining edges are treated as the test set. The network embedding methods are run on the subgraph formed from training set edges only. For each edge in the test set, embeddings of the incident nodes (learnt from the training set) are used as features. 5-fold cross validation is used on the test set edges to evaluate the Logistic Regression classifier performance. The entire procedure is repeated 5 times to obtain different random samples of train and test sets. Mean and standard deviation values of the classification evaluation metrics are reported. We use the area under the ROC curve (AUROC) and the area under the precision-recall curve (AUPRC) as evaluation metrics and Logistic Regression as the classifier.

Node Classification. All the network embedding methods are run on the entire dataset to obtain the node embeddings. We use the micro-F1 and macro-F1 as the evaluation metrics and Stochastic Gradient Descent (SGD) classifier. 5-fold cross validation on the entire data is used to evaluate classifier performance on node classification. We report the mean and standard deviation values.

Statistical Significance. To quantify the significance of the improvement achieved by DualHGCN, when compared with baselines, we compute the one-sided Wilcoxon rank-sum p-value [13] between DualHGCN and the next-best results in each experiment. In Table 3, 4 and 5, asterisks are used to represent where DualHGCN's improvement over baselines is significant (one-sided rank-sum p-value <0.01).

Parameter Settings. The dimensions of initial features (for both with and without attributes) and final embeddings are all empirically set to be 32. We run GraphSAGE with different aggregators (e.g., mean, lstm, and pooling) and show the best results. In Metapath2vec, we use 'U-V-U' as the meta-path to model the 'base' bipartite network. For all baseline methods, we optimize their models with different parameters and report the best performance scores.

For our method, DualHGCN, the default number of layers is 2. The number of negative samples is different from distinct datasets and ranges in $\{1, 2, 3\}$. The Adam optimizer is used in our model to optimize parameters via backpropagation. When the symmetric version of the hypergraph convolutional operator (Eq. 4) is used in DualHGCN, we call the method DualHGCN-sym and when the asymmetric operator (Eq. 5) is used, we call the method DualHGCN-asym. The classifiers used for link prediction and node classification, and evaluation metrics are all from the scikit-learn library [24]. All codes, data and experimental settings of the DualHGCN model are freely available[4].

[4]https://github.com/xuehansheng/DualHGCN

5.4 Results on Link Prediction (Q1)

Table 3 shows the results obtained by DualHGCN and baselines on all four datasets without attributes, and Table 4 shows their performance on Amazon and Alibaba with attributes.

Both Table 3 and 4 show that DualHGCN significantly outperforms other baselines on both datasets without and with node attributes. In Table 3, DualHGCN-asym achieves the highest scores on all four datasets. For DTI, the AUROC and AUPRC score achieved by initial features, which trains the adjacency matrix with the tied autoencoder, are 63.43% and 72.34% respectively, and three bipartite network embedding methods achieve the similar highest metric score among baselines (BiNE, BGNN-adv, and BiANE) where the highest AUROC score achieved by BiANE (91.86%) and the highest AUPRC score achieved by BiNE (92.84%). However, DualHGCN-asym performs better than all baselines and achieves the highest score on the DTI dataset (93.85% for AUROC and 95.00% for AUPRC respectively). For Alibaba-s, the AUROC and AUPRC score achieved by DualHGCN-asym are 87.57% and 89.02% respectively, which are both higher than the second highest scores achieved by BGNN-adv (78.35% for AUROC and 80.93% for AUPRC). Moreover, for Amazon and Alibaba with attributes, DualHGCN also achieves the highest metric scores (see Table 4). It demonstrates that our proposed DualHGCN method is effective both with and without the initial attribute information on nodes.

Comparing Table 3 and 4 we see that training tied autoencoder with the adjacency matrix as initial features plays an important role in predicting unknown interactions. For instance, the AUROC and AUPRC scores achieved by initializing features with the adjacency matrix are both almost 15% higher than the scores achieved by just using the attributes as features. DualHGCN appears to be more robust to different initial features, compared to most other methods. We observe that performance gap for baseline methods, across the two tables, is large. In contrast, AUROC and AUPRC values of DualHDCN are similar across the two feature initializations.

5.5 Results on Node Classification (Q2)

Note that the Alibaba dataset contains attributes for each node, and we adopt two different ways to generate initial features. Alibaba(adj) denotes that we use the adjacency matrix as the initial features, and Alibaba(attr) means that attributes are used to generate initial features. For the smaller unattributed dataset Alibaba-s, only the adjacency matrix is used to generate initial features.

The experimental results of both DualHGCN and baselines are summarized in Table 5. The proposed DualHGCN performs significantly better than other baselines. The micro-F1 and macro-F1 of DualHGCN-asym achieved on Alibaba-s dataset are 36.43% and 35.73% respectively, which are significantly higher than the second highest score achieved by BGNN (29.74% for micro-F1 and 22.71% for macro-F1). The comparison between initial features and DualHGCN demonstrates the superior performance of DualHGCN on fusing the initial features and topological information to enhance the quality of unsupervised network embedding. Moreover, DualHGCN and baselines achieve higher metric scores on Alibaba (attr) than Alibaba (adj), which indicates that attributes of nodes contribute to improving the effects of node classification. For instance, the AUROC and AUPRC values achieved by DualHGCN-asym when

Table 3: The AUROC and AUPRC values of DualHGCN and baselines on the task of link prediction(%). The initial features for all datasets are adjacency matrix with tied autoencoder.

Methods	DTI		Amazon		Alibaba-s		Alibaba	
	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Initial features	63.43±0.74	72.34±0.73	70.57±0.55	74.50±0.63	67.08±0.45	68.21±0.67	68.06±0.25	71.38±0.28
Node2vec	50.88±0.37	57.45±0.38	50.30±0.44	55.44±0.47	50.43±0.29	51.42±0.50	50.10±0.49	51.52±0.29
GraphSAGE	79.34±0.39	82.36±0.24	69.99±0.18	69.39±0.30	64.91±0.14	65.76±0.21	66.49±0.09	60.36±0.13
GCN	56.95±0.13	76.00±0.29	64.93±0.12	77.45±0.15	63.08±0.10	79.59±0.15	56.87±0.04	77.66±0.09
GAT	76.33±0.25	80.64±0.31	66.70±0.13	70.16±0.15	53.28±0.28	54.29±0.66	55.38±0.30	54.49±0.47
HGNN	77.87±1.07	83.57±1.02	80.14±0.32	82.94±0.17	67.07±0.12	69.34±0.07	69.64±0.15	73.50±0.07
HCHA	63.77±1.39	69.83±1.07	62.66±0.72	67.84±0.72	63.61±0.16	65.47±0.18	65.84±0.09	68.81±0.06
MGCN	50.14±0.11	62.07±2.86	51.84±1.34	62.35±0.61	66.31±1.19	68.60±1.50	51.23±0.63	52.79±1.20
HyperGCN	68.99±1.70	77.34±1.86	68.42±1.02	73.78±0.60	63.72±0.22	63.54±0.17	61.38±1.12	65.21±0.70
Metapath2vec++	85.99±0.12	87.73±0.43	60.24±0.17	63.58±0.21	78.85±0.22	69.17±0.37	65.98±0.18	70.97±0.20
HAT	86.22±0.14	87.21±0.12	67.26±0.21	70.67±0.17	57.17±0.61	57.64±0.85	56.98±0.88	58.51±1.40
BiNE	90.74±0.45	92.84±0.27	78.70±0.98	80.56±2.20	72.54±0.47	75.99±0.76	78.94±0.56	79.13±0.43
BGNN-mlp	76.78±2.07	83.05±1.59	68.32±0.23	74.11±0.23	61.96±0.64	65.22±0.67	66.54±0.23	68.84±0.18
BGNN-adv	90.35±1.80	92.35±1.11	83.47±0.16	84.70±0.15	77.49±1.10	77.26±1.01	82.76±0.09	82.40±0.10
BiANE	91.86±0.19	92.19±0.25	76.70±0.19	78.64±0.34	78.35±0.31	80.93±0.25	78.47±0.12	82.35±0.12
DualHGCN-sym	93.53±0.28	94.54±0.21	85.47±0.69	87.98±0.61	86.86±0.41	88.50±0.44	84.53±0.55	86.72±0.44
DualHGCN-asym	**93.85±0.25***	**95.00±0.13***	**86.69±0.26***	**88.69±0.85***	**87.57±0.41***	**89.02±0.42***	**85.54±0.80***	**87.51±0.82***

Table 4: The AUROC and AUPRC values of DualHGCN and baselines on the task of link prediction. The initial features for Amazon and Alibaba are attributes.

METH	Amazon		Alibaba	
	AUROC	AUPRC	AUROC	AUPRC
Inits	57.19±0.32	61.94±0.34	54.05±0.24	55.59±0.28
N2v	50.30±0.44	55.44±0.47	50.10±0.49	51.52±0.29
GSA	67.76±0.28	70.10±0.22	77.69±0.43	76.79±0.49
GCN	56.26±0.35	58.19±0.41	71.38±0.36	69.16±0.18
GAT	62.44±0.22	67.11±0.13	59.12±0.68	59.73±0.66
HGNN	77.41±0.20	81.14±0.13	63.16±0.27	66.16±0.20
HCHA	62.66±0.72	67.84±0.72	57.98±0.15	59.03±0.25
MGCN	64.52±1.10	71.34±0.67	50.54±0.25	52.43±0.98
HGCN	58.61±1.39	70.59±1.54	63.76±0.16	66.09±0.10
M2v++	60.24±0.17	63.58±0.21	65.98±0.18	70.97±0.20
HAT	69.45±0.26	72.22±0.20	50.51±0.77	52.09±0.95
BiNE	78.70±0.98	80.56±2.20	78.94±0.56	79.13±0.43
BGN-m	68.97±1.54	73.80±1.11	67.62±0.34	70.92±0.21
BGN-a	79.43±0.33	81.48±0.36	83.42±0.25	81.53±0.23
BiANE	79.57±0.25	81.99±0.20	76.31±0.16	79.89±0.15
DHG-s	84.87±0.75	87.19±0.65	86.27±0.56	88.07±0.45
DHG-a	**86.55±0.15***	**88.55±0.16***	**86.76±0.46***	**88.59±0.23***

initialized with the adjacency matrix as features are 34.29% and 33.95% respectively, which are about 10% lower than the scores achieved by the model with attributes as initial features.

Note that the micro-F1 scores of many baselines are much larger than the corresponding macro-F1 scores, which indicates that these baselines result in classifications biased towards the large classes. For DualHGCN, the small gap between the micro-F1 and macro-F1 scores shows that DualHGCN is good at handling imbalanced classes for node classification.

5.6 Effects of Message-passing Strategies (Q3)

In DualHGCN, we propose two message passing strategies, intra-message passing, and inter-message passing. The intra-message passing strategy transfers learned features of 'base' homo-hypergraph to other specific homo-hypergraphs, i.e., 'click', 'enquiry', and 'contact' homo-hypergraphs, and the inter-message passing strategy shares features across dual homo-hypergraph sets (users and items) to enable communication between two distinct types of nodes. To answer **Q3**, we perform an ablation study on the DualHGCN model. Figure 3 shows the performance of DualHGCN-asym on Alibaba-s and Alibaba dataset with different message passing strategies. Overall, both intra- and inter-message passing strategies contribute to the tasks of link prediction and node classification, and the inter-message passing strategy plays an essential role in modeling real-world user behavior logs in the e-commerce platform.

The effect of intra- and inter-message passing strategy depends on the distribution of nodes and edges among different types. In Alibaba, the inter-message passing strategy plays a more important role because of the imbalance of the average hyperedge degrees in different dual homo-hypergraphs. Note that the degree of a hyperedge is the number of nodes of the hypergraph incident to this hyperedge.

Table 5: The micro-F1 and macro-F1 values of DualHGCN and baselines on the task of node classification (%). Alibaba(adj) denotes the adjacency matrix is used as initial features, and Alibaba(attr) means attributes are preprocessed as initial features.

Methods	Alibaba-s		Alibaba(adj)		Alibaba(attr)	
	micro-F1	macro-F1	micro-F1	macro-F1	micro-F1	macro-F1
Initial features	25.97±0.34	8.25±0.09	26.63±0.47	8.41±0.12	39.34±0.21	25.61±0.37
Node2vec	21.17±0.41	20.09±0.30	21.37±0.64	19.95±0.43	21.37±0.64	19.95±0.43
GraphSAGE	22.06±0.71	19.40±0.62	23.83±1.08	20.00±0.72	23.47±1.14	19.52±0.43
GCN	21.91±0.71	19.51±0.38	23.67±1.07	19.32±0.33	24.21±1.10	18.33±0.93
GAT	22.70±0.51	19.70±0.48	23.15±0.62	19.98±0.68	23.64±1.54	18.70±0.34
HGNN	25.59±0.97	9.06±0.93	26.77±1.03	13.03±1.84	32.82±0.74	21.51±0.75
HCHA	26.22±0.10	8.31±0.03	27.10±0.03	8.53±0.02	44.88±0.35	29.17±0.41
MGCN	25.93±1.00	19.51±0.92	26.49±0.53	11.36±1.37	40.48±1.51	30.47±1.91
HyperGCN	26.27±0.13	8.32±0.03	27.11±0.06	8.53±0.02	40.66±0.31	26.53±0.34
Metapath2vec++	22.34±0.68	20.32±0.60	22.72±0.23	20.13±0.46	22.72±0.23	20.13±0.46
HAT	25.64±1.48	15.06±1.57	27.07±0.38	16.14±0.42	26.28±1.62	16.05±1.33
BiNE	26.65±0.93	20.77±0.98	26.98±0.44	20.64±0.57	26.98±0.44	20.64±0.57
BGNN-mlp	28.01±1.67	22.71±1.96	24.50±0.52	19.86±0.54	25.04±0.67	17.43±1.12
BGNN-adv	29.74±1.82	16.92±1.74	28.39±0.51	21.59±1.09	46.32±1.53	36.98±0.60
BiANE	22.29±0.45	19.91±0.54	22.65±0.20	20.01±0.53	22.84±0.63	19.89±0.47
DualHGCN-sym	34.68±1.19	34.02±1.06	31.54±0.74	29.77±0.60	45.21±0.85	41.65±0.86
DualHGCN-asym	**36.43±0.81***	**35.73±1.20***	**34.29±0.54***	**33.95±0.36***	**46.63±0.48**	**43.59±0.52***

For instance, the average hyperedge degrees of different hypergraphs built from the Alibaba dataset are as follows, i.e., 2.28 for 'base' homo-hypergraph, 2.23 for 'click' homo-hypergraph, 1.79 for 'enquiry' homo-hypergraph and 1.59 for 'contact' homo-hypergraph respectively. Note that the gap of hyperedge degrees between 'base' and 'click'/'enquiry'/'contact' homo-hypergraphs is small, thus the contribution of the intra-message passing strategy on Alibaba is limited. The most conspicuous improvement of the intra-message passing is on Alibaba(attr) for the task of node classification. DualHGCN-asym without both intra- and inter- message passing strategy gets the 24.52% for micro-F1 and 17.04% for macro-F1 on Alibaba(attr) for the node classification task, and the micro-F1 and macro-F1 scores achieve 30.32% and 21.63% respectively if the intra-message passing is added into the DualHGCN-asym model.

In contrast, the gap of hyperedge degrees between dual homo-hypergraph sets of users and items is large. For instance, in 'base' homo-hypergraph, the average hyperedge degrees for users and items are 4.27 and 1.56 respectively. In this case, the effect of the inter-message passing strategy, which transfers information between two distinct domains (users and items), is essential. DualHGCN-asym without both intra- and inter- message passing strategy gets the 70.71% for AUROC and 69.50% for AUPRC on Alibaba(attr) for the link prediction task, and the AUROC and AUPRC scores achieve 86.53% and 87.99% when inter-message passing is added into the DualHGCN-asym model. In real-world datasets, especially in e-commerce, the imbalance of the average hyperedge degrees between users and items is common and difficult to model. Thus, the inter-message passing strategy plays an important role in modeling sparse dual homo-hypergraphs.

5.7 Effects of Multiplex and Sparsity (Q4)

To study the effect of multiplexing, we evaluate the performance of DualHGCN on each homo-hypergraph independently. We also evaluate the performance of DualHGCN at different sparsity levels.

Effects of Multiple Edges. We run the DualHGCN-asym method on each edge-type hypergraph (i.e., 'base', 'click', 'enquiry' and 'contact' homo-hypergraph) and compare it with DualHGCN-asym employed on all homo-hypergraphs. The experimental results are shown in Figure 4. The results demonstrate the superior performance due to integrating different types of edges compared to treating these edges independently in each homogeneous hypergraph. For instance, the AUROC and AUPRC scores achieved by DualHGCN-asym on Alibaba(attr) for node classification are 46.63% and 43.59% respectively, which is significantly higher than other edge-type homo-hypergraphs (e.g., 41.31% for AUROC and 36.94% for AUPRC on 'base' homo-hypergraph). Further, from Figure 4, we find that the performance scores are correlated to the information enrichment of sub-bipartite network. We observe that the performance scores achieved on four homo-hypergraphs decrease progressively with decreasing number of edges in each sub-bipartite network (25,869 for 'base', 25,180 for 'click', 16,125 for 'enquiry', 4,429 for 'contact' sub-bipartite networks).

Effects of Sparsity. We randomly delete a specific ratio of existing edges to increase the sparsity of the multiplex bipartite network, and employ DualHGCN, BiNE, BGNN-adv, and BiANE on these datasets to evaluate the performance.

Figure 5 shows that DualHGCN still significantly outperforms other baselines (BiNE, BGNN-adv and BiANE) when the networks become more sparse. With increase in sparsity, the performance of BiNE drops steeply compared to that of DualHGCN, BGNN-adv,

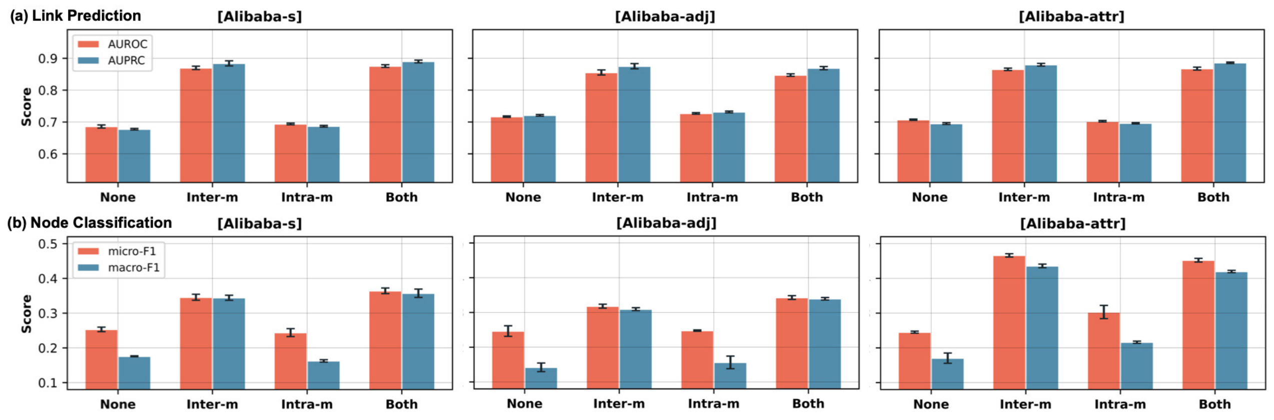

Figure 3: The results of DualHGCN-asym with/without intra-/inter-message passing strategy on two datasets for link prediction and node classification tasks.

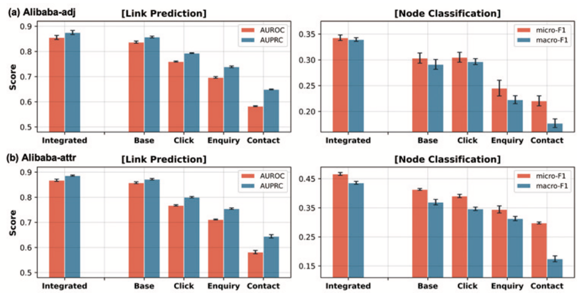

Figure 4: The results of DualHGCN-asym on different edge-type datasets of Alibaba. The figure (a) and (b) use adjacency matrix and attributes as the initial features respectively.

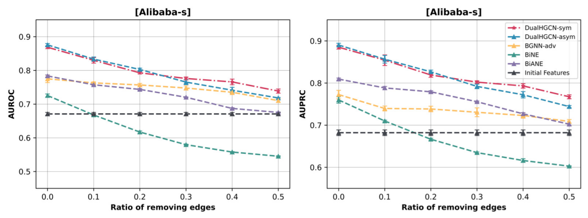

Figure 5: The results of DualHGCN-sym/asym, BGNN-adv, BiNE, BiANE, and initial features on Alibaba-s with different sparsity of networks for link prediction.

and BiANE because BiNE focuses on the topological structure and does not utilize the initial features either from adjacency matrix or attributes. The rate of decrease in performance of DualHGCN is similar to that of BiANE. In extreme cases, when we randomly delete more than 50% edges, the performance scores achieved by DualHGCN is similar to the performance of initial features. Overall, DualHGCN has the best performance at various sparsity levels.

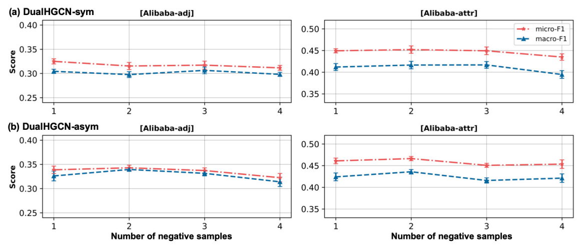

Figure 6: The results of DualHGCN-sym/DualHGCN-asym on Alibaba with different numbers of negative samples for node classification.

5.8 Sensitivity Analysis and Visualization (Q5)

We evaluate the sensitivity of DualHGCN to its three main (number of negative samples, number of layers, and parameter λ) and also qualitatively analyze the embeddings.

Effect of Negative Samples. To train our model, we randomly sample n negative edges (unseen edges) for each positive edge (existing edge). The number of negative samples may affect the performance of final embedding. Here, we vary n from 1 to 4, and evaluate the performance of DualHGCN on Alibaba-attr for the task of node classification (Figure 6). Results demonstrate the robustness of our model DualHGCN on different number of negative samples.

Effect of Layers. We investigate the performance of DualHGCN with different number of layers, ranging from 1 to 5. The results in Figure 7 show that DualHGCN achieves the best performance with two layers. With increase in number of layers, the performance of DualHGCN decreases slightly on both tasks of link prediction and node classification. This phenomenon has also been observed in classical graph convolutional networks [20]. The reason stated in [21] is that the graph convolutional operator is a special form of Laplacian smoothing. Increasing the number of layers makes it more difficult to train. Multiplication of Laplacian smoothing could lead to features of nodes being mixed and difficult to distinguish. This problem also exists in hypergraph convolutional operators.

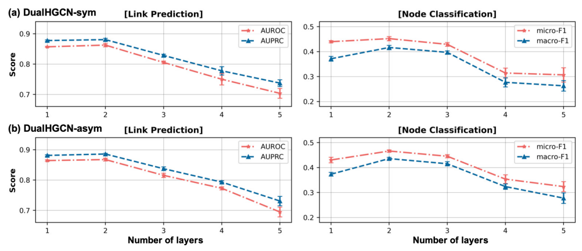

Figure 7: DualHGCN-sym/asym on Alibaba with varying number of layers for link prediction and node classification.

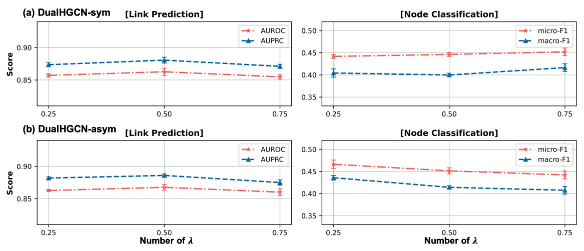

Figure 8: DualHGCN-sym/asym on Alibaba(attr) with varying λ for link prediction and node classification.

Effect of Parameter λ. The hyper-parameter λ in the loss function is used to balance the importance between positive samples and negative samples. We investigate the effect of varying parameter λ, from 0.25 to 0.75, on both tasks of link prediction and node classification. From Figure 8, we find that the performance of the model is not markedly sensitive to changes in λ in both link prediction and node classification tasks.

Visualization. To conduct a qualitative assessment of the embeddings, we use the t-SNE [35] to visualize the final embeddings. Figure 9 shows the 2D-visualization of the embeddings from Alibaba network from DualHGCN-asym, BGNN-adv, BiANE and BiNE, where red nodes represent users and blue nodes represent items. BiNE produces embeddings in the same space for both users and items and thus visually the embeddings of the nodes are not well separated. BiANE improves on BiNE in terms of separability because BiANE integrates the attribute information of two different types of nodes. BGNN also shows good layout because it models the distinction between two types of nodes. Visually, DualHGCN gives the best separation between the two types of nodes.

6 CONCLUSION

Multiplex bipartite networks appear in numerous important applications. To our knowledge, our model DualHGCN is the first network embedding method that can model multiple edge types and node attributes in bipartite networks. Further, it also effectively addresses common real-world challenges of sparsity and imbalance in node and edge type distributions. The scalable transformation employed in DualHGCN to two sets of dual homogeneous hypergraphs enables the use of hypergraph convolutional operators on sparse inputs. The intra-message passing strategy captures topological information across multiplex edges and addresses the problem of edge-type imbalance. The inter-message passing strategy tackles the challenges of node-degree and node-type imbalance between the two distinct node sets. Further, the DualHGCN architecture effectively uses node attributes when provided as inputs. Our extensive experiments demonstrate the efficacy of DualHGCN on four real-world datasets for the tasks of link prediction and node classification. DualHGCN significantly outperforms 14 state-of-the-art methods from 4 different categories of embedding techniques. They also highlight the strengths of our model with respect to robustness to varying sparsity levels, node attribute initialization strategies and handling of imbalanced classes.

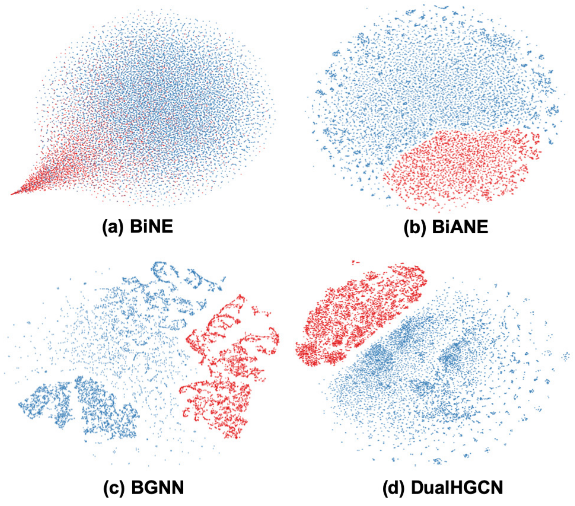

Figure 9: Visualization of node embeddings on Alibaba with attributes dataset (red: users, blue: items).

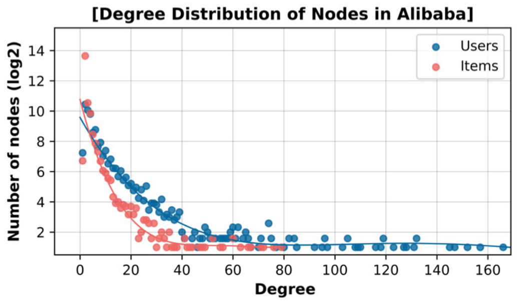

Figure 10: Degree distribution of users, items in Alibaba.

A APPENDIX

Supplementary Material Figure 10 shows the degree distribution of users and items in the Alibaba. Users have more rich and complicated structural information (e.g., users have more cases with degrees more than 2, and the degree of most of the items is 2.) Figure 11 shows the proportion of edge-type and node-number in the Alibaba. In Figure 11 (a), 55% of edges is of type 'click', which is more than other types of edges. In Figure 11 (b), the number of items are more than the number of users. These figures show the problem of edge-type and node-number imbalance in the data.

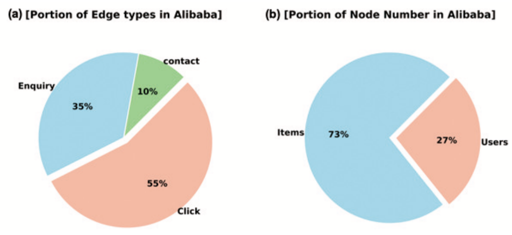

Figure 11: Edge, Node type distributions in Alibaba dataset.

Table 6: Parameters of DualHGCN used in our experiments.

Parameters	DTI	Amazon	Alibaba-s		Alibaba	
	LP	LP	LP	NC	LP	NC
lr	0.002	0.002	0.001	0.001	0.002	0.005
Epochs	4000	3000	3000	5000	3000	5000
Optimizer	Adam					
Dropout	0.5	0.5	0.5	0.5	0.5	0.3
Weight decay	5e-4					
λ	0.5	0.5	0.5	0.5	0.75	0.25
Layers	2					
Neg samples	2	3	1		1	2
Inter	True	True	True		True	True
Intra	False	True	True		True	False
Output emb	32					

Implementation Details of DualHGCN In Table 6, we show the parameters of DualHGCN-sym/-asym used in our experiments.

REFERENCES

[1] Song Bai, Feihu Zhang, and Philip H. S. Torr. 2019. Hypergraph Convolution and Hypergraph Attention. *ArXiv* abs/1901.08150 (2019).
[2] Pierre Baldi. 2011. Autoencoders, unsupervised learning and deep architectures. In *International Conference on Unsupervised and Transfer Learning Workshop*.
[3] HongYun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *TKDE* 30 (2018), 1616–1637.
[4] Yukuo Cen, Xu Zou, J. Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation Learning for Attributed Multiplex Heterogeneous Network. In *KDD*.
[5] Hongxu Chen, Hongzhi Yin, Xiangguo Sun, Tong Chen, Bogdan Gabrys, and Katarzyna Musial. 2020. Multi-level Graph Convolutional Networks for Cross-platform Anchor Link Prediction. *ArXiv* abs/2006.01963 (2020).
[6] Hongxu Chen, Hongzhi Yin, W. Wang, Hao Wang, Quoc Viet Hung Nguyen, and Xue Li. 2018. PME: Projected Metric Embedding on Heterogeneous Networks for Link Prediction. In *KDD*.
[7] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2019. A Survey on Network Embedding. *TKDE* 31 (2019), 833–852.
[8] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. Metapath2Vec: Scalable Representation Learning for Heterogeneous Networks. In *KDD*.
[9] Yihe Dong, Will Sawin, and Yoshua Bengio. 2020. HNHN: Hypergraph Networks with Hyperedge Neurons. *ArXiv* abs/2006.12278 (2020).
[10] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph Neural Networks. In *AAAI*.
[11] Ming Gao, Leihui Chen, Xiangnan He, and Aoying Zhou. 2018. BiNE: Bipartite Network Embedding. In *SIGIR*.
[12] Ming Gao, Xiangnan He, Leihui Chen, and Aoying Zhou. 2019. Learning Vertex Representations for Bipartite Networks. *ArXiv* abs/1901.09676 (2019).
[13] Edmund A. Gehan. 1965. A generalized Wilcoxon test for comparing arbitrarily singly-censored samples. *Biometrika* 52 (1965), 203–23.
[14] Aditya Grover and Jure Leskovec. 2016. Node2Vec: Scalable Feature Learning for Networks. In *KDD*.
[15] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NeurIPS*.
[16] Chaoyang He, Tian Xie, Yu Rong, Wen bing Huang, Yanfang Li, Junzhou Huang, Xiang Ren, and Cyrus Shahabi. 2019. Bipartite Graph Neural Networks for Efficient Node Representation Learning. *ArXiv* abs/1906.11994 (2019).
[17] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. 2018. Leveraging Meta-path based Context for Top- N Recommendation with A Neural Co-Attention Model. In *KDD*.
[18] Linmei Hu, Tianchi Yang, Chuan Shi, Houye Ji, and Xiaoli Li. 2019. Heterogeneous Graph Attention Networks for Semi-supervised Short Text Classification. In *EMNLP/IJCNLP*.
[19] Wentao Huang, Yuchen Li, Yuan Fang, Ju Fan, and Hongxia Yang. 2020. BiANE: Bipartite Attributed Network Embedding. In *SIGIR*.
[20] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.
[21] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. *ArXiv* abs/1801.07606 (2018).
[22] Xin Liu, Tsuyoshi Murata, Kyoung-Sook Kim, Chatchawan Kotarasu, and Chenyi Zhuang. 2019. A general view for network embedding as matrix factorization. In *WSDM*.
[23] Walter Nelson, Marinka Zitnik, Bo Wang, Jure Leskovec, Anna Goldenberg, and Roded Sharan. 2019. To Embed or Not: Network Embedding as a Paradigm in Computational Biology. *Frontiers in Genetics* 10 (2019).
[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
[25] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *KDD*.
[26] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. 2019. Heterogeneous Information Network Embedding for Recommendation. *TKDE* 31 (2019), 357–370.
[27] Chang Su, Jie Tong, Yongjun Zhu, Peng Cui, and Fei Wang. 2020. Network embedding in biomedical data science. *Briefings in bioinformatics* (2020).
[28] Xiangguo Sun, Hongzhi Yin, Bo Liu, H. Chen, J. Cao, Y. Shao, and N. Hung. 2021. Heterogeneous Hypergraph Embedding for Graph Classification. In *WSDM*.
[29] Justin Sybrandt and Ilya Safro. 2019. FOBE and HOBE: First- and High-Order Bipartite Embeddings. *ArXiv* abs/1905.10953 (2019).
[30] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. PTE: Predictive text embedding through large-scale heterogeneous text networks. In *KDD*.
[31] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *WWW*.
[32] Jing Tang, Ziaurrehman Tanoli, Balaguru Ravikumar, and et al. 2018. Drug Target Commons: A Community Effort to Build a Consensus Knowledge Base for Drug-Target Interactions. *Cell Chemical Biology* 25 (2018), 224–229.e2.
[33] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. CANE: Context-Aware Network Embedding for Relation Modeling. In *ACL*.
[34] Ke Tu, Peng Cui, Xiao Wang, Fei Wang, and Wenwu Zhu. 2018. Structural Deep Embedding for Hyper-Networks. In *AAAI*.
[35] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605.
[36] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.
[37] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *KDD*.
[38] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item Recommendation with Sequential Hypergraphs. In *SIGIR*.
[39] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous Graph Attention Network. In *WWW*.
[40] Hansheng Xue, Luwei Yang, Wen Jiang, Yi Wei, Yi Hu, and Yu Lin. 2020. Modeling Dynamic Heterogeneous Network for Link Prediction using Hierarchical Attention with Temporal RNN. In *ECML/PKDD*.
[41] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Pratim Talukdar. 2019. HyperGCN: A New Method For Training Graph Convolutional Networks on Hypergraphs. In *NeurIPS*.
[42] Hongzhi Yin, Hongxu Chen, Xiaoshuai Sun, Hao Wang, Yang Wang, and Quoc Viet Hung Nguyen. 2017. SPTF: A Scalable Probabilistic Tensor Factorization Model for Semantic-Aware Behavior Prediction. In *ICDM*.
[43] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *KDD*.
[44] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. 2019. Heterogeneous Graph Neural Network. In *KDD*.
[45] Ruochi Zhang, Yuesong Zou, and Jian Ma. 2020. Hyper-SAGNN: a self-attention based graph neural network for hypergraphs. In *ICLR*.
[46] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* (2018).