Heterogeneous Multilayer Graph Convolutional Network

Hamed Kabiri Kenari[1]

The Dept. of Electrical & Computer Engineering, Faculty of Engineering, Kharazmi University, Tehran , Iran.

ABSTRACT

Graph convolutional network variants for heterogeneous graph data have been utilized in a range of analytical applications, including node classification and link prediction. Many heterogeneous graph neural networks rely on the selection of meta-paths. However, the task of selecting appropriate meta-paths becomes increasingly complex when heterogeneous graphs encompass a diverse array of node and edge types. In practical applications, many graphs exhibit significant complexity, characterized not only by multi-typed nodes but also by varying edge types that may exist between the same pairs of nodes. To effectively model these complexities, it is essential to evaluate the importance of these multiplex relationships within the networks. In response to these challenges, this paper introduces a Heterogeneous Multilayer Graph Convolutional Network (HMGCN) designed for embedding heterogeneous networks. Initially, HMGCN decomposes the heterogeneous graph into subgraphs (or layers) based on edge types. Subsequently, it applies a graph convolutional neural network specifically tailored for these subgraph (or layer). Finally, HMGCN integrates the embeddings of each node through an attention mechanism that evaluates the importance of each subgraphs (or layers) to which the node belongs. This approach enables the effective learning of node embeddings by incorporating various node types, attribute semantics, and the structural characteristics of different relation types within both semi-supervised and unsupervised learning frameworks. Empirical results obtained from five real datasets across multiple network analytical tasks indicate that HMGCN outperforms state-of-the-art embedding baselines.

KEYWORDS

Heterogeneous Networks; Graph Representation Learning; Network Embedding; Graph Convolutional Networks; Multiplex Heterogeneous Networks

1. INTRODUCTION

Graph representation learning has developed into a novel framework that entails the embedding complex graphs into a low-dimensional vector space. This process preserves the proximity of nodes in accordance with the graph's structures and the intrinsic characteristics of the graph. The effectiveness of network representation is essential for a variety of graph analytical tasks, including node classification, link prediction, and recommendation systems. Graph Convolutional Networks (GCNs), which are a specific category of graph neural networks tailored for the representation learning of complex graphs that incorporate feature information, have recently

[1] Email Addresses: hamedkabiri@ieee.org (Hamed Kabiri Kenari)

been deployed in various online services, including advertising, social media platforms, and e-commerce.

Numerous investigations have been conducted on homogeneous graph representation learning, [6, 17, 27, 30]. However, recent scholarly attention has increasingly turned towards heterogeneous network representation learning, as demonstrated by works such as HERec [29] and metapath2vec [5]. Recent research has employed the capabilities of Graph Neural Networks (GNNs) to aggregate information and features from adjacent nodes in homogeneous graphs. Nevertheless, the diverse types of nodes and edges present in heterogeneous graphs may hinder the effectiveness of GNNs originally designed for homogeneous networks. In response to these challenges, several graph neural models have been developed to enhance heterogeneous graph learning, including HetGNN [43], MAGNN [5], and HAN [32].

The efficacy of current heterogeneous network embedding methodologies has been substantiated in prior research [5, 15, 24, 33]. However, these methodologies are predominantly tailored for heterogeneous networks characterized by a singular view. In practice, numerous graphs exhibit a greater degree of complexity, encompassing not only multiple node types and diverse edge types connecting the same pairs of nodes but also an extensive array of attributes [2]. In the context of e-commerce graphs, there exist two distinct categories of nodes, namely items and users, which are interconnected through various types of relationships. These relationships include actions such as purchasing, clicking, adding to preferences, and adding to the shopping cart, all of which occur between the same pairs of items and users [38]. The intricate relationships among different node types in such graphs, referred to as multiplex networks, are defined by multiple distinct perspectives. It is crucial to recognize that multiplex networks are inherently more complex than their heterogeneous counterparts. In the context of an e-commerce network, the heterogeneity is exemplified by the presence of two node types: users and items. Furthermore, users may engage in various forms of interactions (including click, purchase, and review) with items [35], thereby highlighting the multiplex nature of the graph. The varied interactions of users with items offer diverse perspectives on both the items and the users themselves, thereby requiring their separate consideration.

The task of representation learning in heterogeneous networks presents considerable challenges due to the variety of node attributes and the complexity of their structures. The efficacy of such learning is essential for various network mining applications. Recent research has investigated representation learning techniques tailored for heterogeneous networks characterized by multiplex relationships; however, these approaches are often hindered by several significant limitations. The performance of existing representation learning models is heavily contingent upon the meticulous design of meta-paths. Consequently, the development of a learning framework that operates independently of meta-paths for heterogeneous graphs remains a formidable challenge.

Heterogeneous networks differ from homogeneous networks in that they encompass multiple types of nodes and edges. Prior studies have necessitated the manual definition of meta-paths, which are subsequently sampled for the purpose of learning node representations. This manual process is inherently complex, particularly in large-scale networks that may contain an extensive array of meta-paths. The process of sampling a substantial quantity of meta-paths needs more time. Moreover, the choice of meta-paths plays a crucial role in shaping node representations, which in turn affects the efficacy of network embeddings in a range of subsequent tasks. When the number of heterogeneous meta-paths is substantial, characterized by diverse lengths and relational interactions, identifying the most suitable meta-path for embedding models based on meta-path aggregation becomes increasingly challenging. Additionally, in heterogeneous networks that feature diverse node types and multiplex relationships, there is a pressing need for an effective model capable of integrating both node heterogeneity and multiplex relations into the resultant representations.

HMGCN effectively addresses the previously mentioned challenges. Unlike conventional approaches that rely on the selection of meta-paths, HMGCN decomposes the heterogeneous graph into bipartite and homogeneous subgraphs (or layers) based on edge types. Following this decomposition, HMGCN employs a graph convolutional neural network specifically designed for each individual subgraph (layer). As a result, each node possesses distinct embeddings, as it belongs to different subgraphs (or layers). The model evaluates the significance of various subgraphs (or layers) within the graph using an attention mechanism that utilizes subgraph-level (or layer) summary representations. Finally, it integrates the distinct embeddings of each node according to the importance of the different subgraphs (or layers) to which the node belongs.

The contributions of this study can be delineated as follows:

(i) We introduce an innovative graph convolutional network specifically developed for the embedding of heterogeneous graphs.

(ii) We perform a series of experiments to assess the efficacy of the proposed model in the contexts of link prediction and node classification tasks, benchmarking its performance against leading state-of-the-art baselines across five real-world datasets.

2. RELATED WORK

2.1 Graph Neural Networks (GNNs) models

The principal objective of a Graph Neural Network (GNN) is to derive a lower-dimensional vector embedding for each node, which can subsequently be applied to a variety of graph mining applications. Graph Convolutional Networks [17] implement convolutional operations on the neighbors of network nodes to facilitate information aggregation. Another model, GraphSAGE [8], presents an inductive variant of GNN model that adeptly generates node embeddings through the use of generalized aggregation functions. The Graph Attention Network (GAT) [31] integrates an attention mechanism to ascertain the weights of the embeddings of neighboring nodes, thereby enabling a nuanced assessment of the influence exerted by these neighbors. In reference [28], R-GCN is introduced that addresses the effects of heterogeneous edge types on nodes, employing coefficient constraints and weight sharing to accommodate multi-graphs characterized by a substantial number of edges. LightGCN [10] streamlines the architecture of graph convolutional networks by omitting non-linear embedding projections through the phase of passing messages. Furthermore, AM-GCN [34] is introduced to facilitate the adaptive learning of deep correlations between node features and topological structures. However, the algorithms previously discussed are primarily designed for homogeneous graphs, which may render them insufficient for accurately representing the characteristics of heterogeneous networks in their embeddings.

2.2 Heterogeneous Graph Representation

Numerous studies have focused on representation learning within heterogeneous graphs [5, 13, 23, 29, 43]. Contemporary techniques for in heterogeneous graphs, such as metapath2vec [5] and HERec [29], utilize random walks to create meta-paths for producing node representation. Graph Neural Networks (GNNs) have emerged as a preferred approach for the representation of network topology. Graph neural network models for heterogeneous networks elevate the GNN framework by integrating the ability to capture both edge and node heterogeneity. For instance, HetGNN [43] simultaneously encodes context heterogeneity and graph structure for representation learning tasks. NARS [40] initially constructs relation subgraphs and subsequently applies one-dimensional convolution on these subnetworks to derive node embeddings, which are then amalgamated. HeGAN [11] introduced heterogeneous Generative Adversarial Networks (GANs). Fu et al. [5] attain metapath-based relational context for node embedding learning through both intra-meta-path and inter-meta-path fusion. However, a significant number of these models rely on the identification of beneficial meta-paths for the heterogeneous network embedding, which may require the incorporation of external domain knowledge to effectively develop appropriate meta-paths.

Furthermore, a number of recent studies seek to alleviate the necessity for meta-path construction in the representation of heterogeneous networks. The HGT model [12] captures the dynamic interdependencies among heterogeneous nodes by utilizing a self-attention mechanism within a graph-based message-passing framework. In contrast, HPN [15] addresses semantic ambiguity by mapping the nodes of meta-paths into a semantic space, subsequently integrating the node embeddings from various meta-paths to derive the final embedding. Nevertheless, it is important to note that many of the previously mentioned heterogeneous network embedding models do not specifically cater to multiplex network data, which is characterized by the presence of multiple types of connections between nodes.

Certain real-world networks exhibit multiplex characteristics, encompassing diverse relationships and connections among nodes. The majority of existing multiplex network embedding models are engineered to translate various types of nodes and edges into latent representations. The MNE model [44] presents a general transformation matrix that can be applied to each layer of the graph, thereby enabling the generation of embeddings with varying dimensions that correspond to each type of edge. GATNE [2] employs edge embeddings, alongside base and attribute embeddings, to create differentiated representations for nodes. It utilizes self-attention mechanisms to generate edge representations that incorporate information from neighboring nodes. DMGI [25] proposes an unsupervised learning approach grounded in a mutual information maximization framework, which aims to reduce discrepancies between relation-aware node embeddings. HGSL [46] initially derives node embeddings based on meta-paths and subsequently applies a Graph Neural Network (GNN) to concurrently learn node attributes and representations from the heterogeneous graph, culminating in the final embedding. Nonetheless, many of the aforementioned methodologies are limited by the requirement for the manual construction of meta-paths.

Recent research has presented FAME [22], which introduces a spectral graph transformation technique that effectively incorporates information from subgraphs while maintaining relation-aware dependencies among nodes. This model employs random projection and adaptively learns parameters to achieve efficient embedding projection. Additionally, in the context of multiplex bipartite networks, DualHGCN [39] initially constructs two collections of homogeneous hypergraphs and subsequently applies spectral hypergraph convolutions to enhance information propagation. Moreover, HDI [16] identifies high-order mutual information to generate supervisory signals for the representation of nodes.

3. PRELIMINARY

We begin by establishing a definition for a heterogeneous network, denoted as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, which comprises a set of nodes $\mathcal{V}$ and a set of edges $\mathcal{E}$.

Definition 1 (Heterogeneous Network): In the graph $\mathcal{G}$, each node within the set $\mathcal{V}$ is associated with attribute feature vectors $X \in \mathbb{R}^{n\times m}$. The cardinality of the node set $\mathcal{V}$ is denoted by n, while the dimensionality of the attribute vectors is indicated by m. To accommodate the heterogeneity of nodes and the presence of multiplex edges, we introduce mapping functions for both edge types and node types, referred to as $\psi: \mathcal{E} \rightarrow \mathcal{R}$ and $\emptyset: \mathcal{V} \rightarrow \mathcal{O}$, respectively. The set of node types is denoted by $\mathcal{O}$, and the set of edge types is represented by $\mathcal{R}$. Each edge $e \in \mathcal{E}$ node $v \in \mathcal{V}$ is linked with a particular type within the sets $\mathcal{R}$ and $\mathcal{O}$, respectively, under the condition that $|\mathcal{O}| + |\mathcal{R}| > 2$.

The problem of representation learning within heterogeneous graphs can be articulated as follows:

Problem (Heterogeneous Graph Representation): Representation learning in heterogeneous networks entails the learning of low-dimensional latent representations for each node, ensuring that the hidden dimensions maintain the heterogeneity of the nodes and the multiple types of edges present.

A summary of the notations employed in the methodology is provided in Table 1.

Table 1: Essential notations

Notation	Definition
$\mathcal{G}$	The input graph
$\mathcal{O}, \mathcal{R}$	The set of node types and the set of edge types in $\mathcal{G}$
$\mathcal{V}, \mathcal{E}$	The set of nodes and the set of edges in the graph $\mathcal{G}$
X	The node attributes matrix in $\mathcal{G}$
$\mathcal{G}^r$	The subgraph pertaining to edge type r
A^r	The adjacency matrix corresponding to $\mathcal{G}^r$
H	The embeddings of nodes
d	Embeddings hidden dimensionality
n, m	The nodes number and attributes number
X^r	The attributes matrix nodes in $\mathcal{G}^r$
H^r	The embeddings matrix of nodes in $\mathcal{G}^r$
s^r	Summary representation of H^r
n^r	The nodes number in $\mathcal{G}^r$
W^r	The trainable weight matrix of the subgraph r
β^r	The importance of subgraph r
q^r	the feature vector associated with subgraph r

4. THE PROPOSED MODEL

In this section, we present Heterogeneous Multilayer Graph Convolutional Network (HMGCN), as illustrated in Figure 1, which depicts its architecture for embedding of node v. Specifically, HMGCN consists of four primary learning components: (i) decomposing the heterogeneous network into bipartite and homogeneous subgraphs (layers) based on the types of edges; (ii) applying a graph convolution module to each individual layer; (iii) Generating a summary representation at the subgraph level that effectively captures the overall content of the embeddings from each layer; and (iv) aggregating the distinct node embeddings through the application of an attention mechanism, which facilitates the integration of the latent vectors derived from (ii) into the ultimate node embeddings.

4.1 Heterogeneous Relations Decomposing

According to Definition 1, heterogeneous graphs are characterized by the presence of diverse node types and multiple edge types, where each edge type serves a specific function and impacts the representation of nodes in a unique manner. Consequently, similar to the approach described in [22], we first generate several sub-networks by differentiating the relationships among the various edge types within the heterogeneous networks.

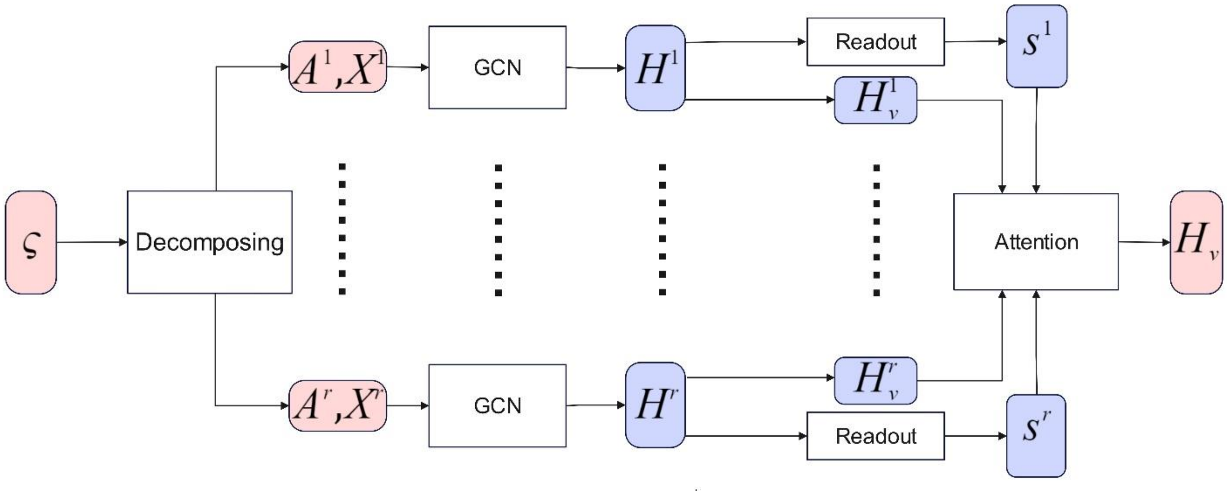


Figure 1: The architecture of HMGCN for embedding of node v

We refer to the subgraphs generated in this study as $\{\mathcal{G}^r|r = 1,2, \dots, |\mathcal{R}|\}$, with their associated adjacency matrices denoted as $\{A^r|r = 1,2, \dots, |\mathcal{R}|\}$, and the attribute vectors represented as $\left\{X^r \in \mathbb{R}^{n^r \times m}|r = 1,2, \dots, |\mathcal{R}|\right\}$, where n^rrepresents the number of nodes within the subgraph $\mathcal{G}^r$. Consider the multiplex relationships between users and items in an online retail context, such as reviews, purchases, and clicks. The decomposed subgraphs correspond to distinct relationship types between users and items: reviews, purchases, and clicks.

4.2 Relation-type Node Embedding

We present a relation-type single-layer Graph Convolutional Network (GCN) denoted as $g_r: \mathbb{R}^{n^r \times m} \times \mathbb{R}^{n^r \times n^r} \to \mathbb{R}^{n^r \times d}$ for each relation type $r \in \mathcal{R}$. This GCN is specifically designed to produce the relation-type embeddings matrix H^rcorresponding to the nodes within$\mathcal{G}^r$:

$$H^r = g_r(X^r, A^r|W^r) = \sigma\left(\widehat{D^r}^{-\frac{1}{2}}\hat{A}^r\widehat{D^r}^{-\frac{1}{2}}X^rW^r\right) \qquad (1)$$

where $\hat{A}^r = A^r + I_n, \widehat{D^r}_{ii} = \sum_j \widehat{A^r}_{ij}, W^r \in \mathbb{R}^{m \times d}$ denote a trainable weight matrix associated with the relation-type GCN g_r, while σ signifies the ReLU nonlinearity.

Subsequently, we derive a graph-level summary representation s^r, which encapsulates the information contained within H^r. To facilitate this process, we utilize the readout function $\mathbb{R}^{n^r \times d}$ → $\mathbb{R}^d$:

$$s^r = Readout(H^r) = \sigma\left(\frac{1}{n^r}\sum_{i=1}^{n^r} H_i^r\right) \qquad (2)$$

where σ represents the logistic sigmoid nonlinearity, and $H_i^r$the i-th row vector of the matrix H^r. Furthermore, it is important to acknowledge that alternative pooling techniques, including SAGPool [18] and max pooling, may also be employed as $Readout(.)$.

4.3 Subgraph-Level Attention

Different types of edges among nodes represent different dependency semantics. As a result, multiplex interactions characterized by different relational semantics will exert diverse influences on the nodes embeddings learning process. To effectively model these multi-typed edge dependencies, we propose the Heterogeneous Multilayer Graph Convolutional Network (HMGCN), which employs an attention mechanism to integrate edge-type-specific embeddings for each node. To address the challenge with the fusion of relation-type node embeddings within a heterogeneous network, we introduce a novel attention mechanism that learns the importance of various relation types and using them to aggregate different node embeddings.

For $v \in \mathcal{V}$, the latent vectors are denoted as $\{H_v^{r_1}, H_v^{r_2}, \dots, H_v^{r_l}\}$, where $\{r_1, r_2, \dots, r_l\}$ represents the set of subgraphs (or layers) to which the node type $\emptyset(v)$ belongs, and l indicates the total number of subgraphs (or layers) to which the node type $\emptyset(v)$ belongs.

Taking a group of l subgraph-level summary representations to which node type $\emptyset(v)$ belongs, these representations are acquired from the $Readout$ layer and serve as input. The learned weights of the subgraphs, denoted as $(\beta^{r_1}, \dots, \beta^{r_l})$, can be expressed as follows:

$$(\beta^{r_1}, \dots, \beta^{r_l}) = att(s^{r_1}, s^{r_2}, \dots, s^{r_l}) \qquad (3)$$

Here, att denotes the deep neural network. Attention mechanism can evaluate importance of various subgraphs (or layers) within a heterogeneous graph.

Consequently, it is possible to utilize the attention mechanism [1] to differentiate among various subgraphs (or layers) in the following manner:

$$H_v = \sum_{i=1}^{l} \beta^{r_i} \,.\, H_v^{r_i} \qquad (4)$$

where H_i signifies the i-th row vector of the matrix H, while β^{r_i} represents the importance of subgraph (or layer) r_i in the process of producing the final embedding for node v, as defined below:

$$\beta^{r_i} = \frac{exp(q^{r_i}.s^{r_i})}{\sum_{j=1}^{l} exp(q^{r_j}.s^{r_j})} \qquad (5)$$

where $q^{r_i} \in \mathbb{R}^d$represents the feature vector associated with subgraph (or layer) r_i.

4.4 Model Learning

For learning the final representation of nodes, we define an objective function to train the model. The HMGCN is trained within two primary learning frameworks—semi-supervised learning and unsupervised learning—contingent upon the availability of node labels and the specific regulations of various downstream tasks.

In the context of unsupervised learning, the optimization of model parameters can be accomplished through the minimization of the binary cross-entropy loss function, utilizing negative sampling:

$$\mathcal{L} = -\sum_{(u,v)\in\Omega} \log\sigma(\langle(H_u)^T, H_v\rangle) - \sum_{(u',v')\in\Omega^-} \log\sigma(-\langle(H_{u'})^T, H_{v'}\rangle) \qquad (6)$$

where $\sigma(.)$ represents the sigmoid function, Ω denotes the set of positive (observed) node pairs, while Ω^- represents the set of negative node pairs that are sampled from the entire unobserved node pairs set, which is the complement of Ω. Additionally, H_v denotes the node v representation, T indicates the operation of matrix transposition, and $\langle,\rangle$ can be understood as a function representing any vector similarity measure, including the inner product.

In the context of semi-supervised learning, the optimization of the model's parameters is achieved by minimizing the cross-entropy loss function through the application of gradient descent and backpropagation. The cross-entropy loss function, which is computed for all labeled nodes and serves to evaluate the discrepancy between the ground truth and the model's predictions, is expressed in the following manner:

$$\mathcal{L} = -\sum_{i\in v_{ids}} Y_i \ln(C.H_i) \qquad (7)$$

where v_{ids} represents the indices set of nodes that possess labels, C denotes the parameters associated with the node classifier, Y_i signifies the label vector corresponding to the i-th node, and H_i indicates the embedding vector for the i-th node. In this context of semi-supervised classification, it is possible to optimize the model and subsequently learn the node embeddings, utilizing a limited subset of labeled nodes as guidance.

Algorithm 1 presents the pseudocode for the proposed Heterogeneous Multilayer Graph Convolutional Network (HMGCN).

Algorithm 1 HMGCN Model's Learning Procedure

Input: Heterogeneous network $\mathcal{G}$, node attributes matrix X.

Output: The nodes embeddings matrix H.

```
Produce the adjacency matrices $\{A^r | r = 1,2,\dots,|\mathcal{R}|\}$ and attributes feature vectors
  $\{X^r \in \mathbb{R}^{n^r \times m} | r = 1,2,\dots,|\mathcal{R}|\}$ by decomposing the graph $\mathcal{G}$ into bipartite and
  homogeneous subgraphs
for $r = 1$ to $|\mathcal{R}|$ do
```

```
    Calculate relation-type specific node embedding $H^r = \sigma\left(\widehat{D^r}^{-\frac{1}{2}}\hat{A}^r\widehat{D^r}^{-\frac{1}{2}}X^r W^r\right)$
    Calculate the subgraph-level summary representation $s^r = \sigma\left(\frac{1}{n^r}\sum_{i=1}^{n^r} H_i^r\right)$
end for
for $i \in \mathcal{V}$ do
    for $j \in \{r_1, r_2, \dots, r_l\}$ the subgraphs (or layers) to which the node $i$ belongs
        Calculate the weight of $\beta^j$
    end for
    Calculate the node embedding $H_v = \sum_{i=1}^{l} \beta^{r_i} .\ H_v^{r_i}$
end for
Calculate $\mathcal{L}$ by utilizing either Eq. (6) or Eq. (7)
process of backpropagation and the adjustment of parameters within HMGCN
Return $H$
```

5 EXPERIMENTS

For our experimental evaluation, we employ five real-world datasets: Alibaba[2], Amazon[3], IMDB[4], DBLP[5], and AMiner[6]. The specific versions of these datasets are referenced as indicated in [42]. Table 1 presents the statistics for these five datasets.

i) The Alibaba dataset comprises four distinct types of multiplex relationships between users and items. In the context of node classification, the ground truth labels are aligned with the categories of the items. Due to scalability constraints associated with certain baseline models, we conduct our analyses on a sampled subset of the Alibaba dataset rather than the entire network data. ii) The Amazon dataset incorporates co-purchasing and co-viewing relationships as multiplex edges among various products. The external attributes of these products, including sales rank, category, price, and brand, are utilized as node attributes. iii) The AMiner dataset represents a heterogeneous network that consists of three types of nodes: conference, paper, and author. The domains of the papers are employed as the labels for the nodes. iv) The DBLP dataset is heterogeneous network that encompasses four nodes types: paper, author, venue, and term. The class labels for nodes indicate the research fields associated with the authors. v) The IMDB dataset is a heterogeneous graph featuring three nodes types: actor, director, and movie. The labels assigned to the nodes correspond to the genres of the movies, while bag-of-words representations are utilized as feature vectors for the node attributes.

[2] https://tianchi.aliyun.com/competition/entrance/231719/information/
[3] http://jmcauley.ucsd.edu/data/amazon/
[4] https://github.com/seongjunyun/Graph_Transformer_Networks
[5] https://www.dropbox.com/s/yh4grpeks87ugr2/DBLP_processed.zip?dl=0
[6] https://github.com/librahu/

Table 2: Evaluation network datasets statistical information

Dataset	nodes	edges	node type	edge type	features
Amazon	10,166	148,865	1	2	1,156
Alibaba	21,318	41,676	2	4	19
DBLP	26,128	119,783	4	3	4,635
IMDB	12,772	18,644	3	2	1,256
AMiner	58,068	118,939	3	3	4

5.2 Baselines

We evaluate the HMGCN against eighteen subsequent network learning baselines.

RandNE [45] utilizes a repetitive projection process to effectively represent the high-order network topology, with the aim of facilitating matrix factorization.

node2vec [7] utilizes random walks to generate sequences of nodes within graphs.

FastRP [3] produces a similarity matrix to model the transitive relationships between nodes, subsequently utilizing sparse random projections for dimensionality reduction.

AM-GCN [34] represents a cutting-edge GCN designed as an adaptive multi-channel framework for semi-supervised classification task.

SGC [36] simplifies GCNs by eliminating the non-linear projection through information propagation across the layers of the graph.

HAN [32] employs a manual selection of meta-paths for the purpose of learning node embeddings and utilizes a graph attention network to analyze heterogeneous graphs, taking into account both inter-network and intra-network interactions.

R-GCN [28] aims to assess the influence of various edge types on nodes by integrating coefficient constraints and weight sharing in the context of heterogeneous graphs.

HPN [15] is developed to reduce semantic confusion through the implementation of a semantic propagation mechanism. In order to integrate comprehensive semantics, HPN employs a semantic aggregation mechanism.

NARS [40] employs a method that separates heterogeneous graphs according to the types of edges and subsequently integrates the features of neighbor nodes within the resulting decoupled subgraphs.

MAGNN [6] is a graph neural network that utilizes a meta-path-based approach, specifically tailored for heterogeneous networks. It incorporates mechanisms for both intra-meta-path aggregation and inter-meta-path integration.

MNE [44] generates the overall representation by integrating the low-dimensional hierarchical representation with the high-dimensional common representation.

PMNE [20] employs the PMNE-c, PMNE-r, and PMNE-n models to synthesize the multiplex relationships, thereby generating a conclusive embedding for each node.

GTN [41] proficiently identifies meta-paths characterized by high-order relationships in order to ascertain the ultimate representation.

GATNE [2] employs the base embedding, along with the attribute and edge representations to produce the final node embeddings. Self-attention incorporates information from neighboring nodes to generate the edge embedding.

FAME [22] utilizes spectral network transformation to identify meta-paths and enhance efficiency via random projection.

DMGI [25] investigates the interrelations among type-specific node embeddings by introducing a consensus regularization framework. Following this, it utilizes an attention mechanism to aggregate the various relational contextual signals.

DualHGNN [39] designed especially multiplex bipartite networks embedding by employing a dual hypergraph GCN to derive node representations.

HGSL [46] represents a cutting-edge heterogeneous graph neural network (GNN) that simultaneously executes GNN parameter learning and heterogeneous graph structure learning for classification purposes.

5.3 Implementation Details

We employ publicly accessible source codes for the establishment of baseline implementations. In our experimental framework, we standardize the hidden dimension size of the embeddings to $d = 300$ across all models under comparison. Furthermore, we explore a range of hyperparameter configurations as delineated in the original publications. To facilitate equitable comparisons, we consistently limit the training iterations number to 300 for node classification and 500 for link prediction.

In the domain of link prediction, we categorize connected nodes within a graph as pairs of positive nodes, while unlinked nodes are regarded as pairs of negative nodes. The positive node pairs are divided into three subsets: a test set, a validation set, and a training set, allocated in proportions of 10%, 5%, and 85% for each edge type, respectively. Following this, we randomly choose an equivalent negative node pairs number to incorporate into the test set, validation set, and training set. In the link prediction experiments, we aim to predict each edge type by leveraging all edge types available in the datasets, calculating the average across all edge types to obtain the ultimate outcome. We similarly partition the nodes into three sets: a test set, a validation set, and a training set, with proportions of 10%, 10%, and 80%, respectively, for the node classification task. A logistic regression classifier is trained for the node classification task. Each execution is conducted ten times, and the average outcomes are reported.

In the node2vec for controlling random walks, the parameters q and p were designated values of 0.5 and 2, respectively. In the GATNE, the parameters $\alpha_r$and β_r were uniformly set to 1 for each type of edge. In the Implementation of the MNE, we established the dimensions of the supplementary vectors to be 10, the length of the walks to be 10, and the total number of walks to be 20. The learning rate was selected from the set $\{0.0005, 0.0001, 0.005, 0.05, 0.01\}$ designated for competing neural network-based models. The dropout rate was established at 0.6, and the multi-head attention mechanism was configured to utilize 8 distinct head-specific representation spaces. In the GTN model, we determined the graph transformer layers number to be 3. In the Implementation of the DMGI, the parameters α, β, γ were selected from the designated value set $\{ 0.1,0.01,0.001,0.0001\}$ and the weight $\mathcal{W}$ assigned to self-connections was set to 3.

In the Implementation of the FAME, we utilize Optuna[7] to tune the parameters $\alpha_1, \ldots, \alpha_K$ and $\beta_1, \ldots, \beta_{|\mathcal{R}|}$, as outlined in the original research. In AM-GCN, the parameters for loss aggregation β, γ are tuned in $\{ 0.1, 0.01, 0.001, 0.0001\}$. In the Implementation of the MAGNN, the independent attention mechanisms number is configured to 4. In the Implementation of the NARS, both the number of hops and the number of feed-forward layers are established at 2. In HGSL, the architecture is defined with 2 GNN layers and the dimension of the hidden layer output is set to 64, respectively. In the case of HPN, the semantic propagation iterations are set to $k = 3$, and the restart probability is established at $\alpha = 0.1$. In the Implementation of the R-GCN, the configurations include a batch size of 126, 2 GNN layers, and a hidden layer dimension of 64. In the Implementation of the DualHGNN, the parameter λ and the asymmetric operator are established at 0.5.

In the HMGCN model, the parameters for the learning rate, dropout rate, and weight decay are established at 0.05, 0.5, and 0.0005, respectively.

The source code for the implementation of the HMGCN model is available at the following link: https://github.com/hamedkabiri/HMGCN.

In order to apply both heterogeneous and homogeneous graph embedding models to multiplex networks, we initially input each layer of the multiplex network independently into the models, treating them as distinct layers to derive a range of node embeddings. Subsequently, we employ mean pooling to produce the final node representations. It is crucial to note that the DualHGNN model is specifically tailored for bipartite multiplex networks and is exclusively applicable to the Alibaba network.

5.4 Link Prediction

In the context of link prediction, we compare the performance of HMGCN model against fifteen baseline models within an unsupervised learning framework. Table 3 displays the findings, with the best outcomes highlighted in bold.

Table 3: a comparative analysis of the performance of various models in relation to the link prediction task across multiple datasets.

Method	Alibaba			IMDB			Amazon			DBLP			AMiner		
	R-AUC	PR-AUC	F1	R-AUC	PR-AUC	F1	R-AUC	PR-AUC	F1	R-AUC	PR-AUC	F1	R-AUC	PR-AUC	F1
Node2vec	0.615	0.581	0.594	0.478	0.567	0.475	0.947	0.945	0.881	0.448	0.453	0.478	0.593	0.664	0.601
RandNE	0.878	0.887	0.824	0.902	0.934	0.838	0.951	0.942	0.904	0.493	0.492	0.493	0.608	0.631	0.606
FastPR	0.924	0.901	0.925	0.868	0.894	0.812	0.955	0.944	0.894	0.516	0.527	0.506	0.621	0.633	0.601
SGC	0.687	0.707	0.624	0.827	0.888	0.768	0.792	0.803	0.761	0.602	0.607	0.587	0.588	0.584	0.566
R-GCN	0.675	0.711	0.628	0.825	0.877	0.791	0.812	0.821	0.784	0.588	0.593	0.566	0.598	0.602	0.611
MAGNN	0.962	0.964	0.947	0.913	0.924	0.886	0.957	0.948	0.916	0.691	0.697	0.684	0.664	0.682	0.667
HPN	0.959	0.962	0.951	0.901	0.904	0.893	0.948	0.948	0.905	0.693	0.711	0.687	0.657	0.665	0.661
PMNE-n	0.967	0.974	0.892	0.675	0.684	0.647	0.957	0.946	0.894	0.673	0.678	0.663	0.652	0.668	0.678
PMNE-r	0.858	0.916	0.825	0.644	0.647	0.614	0.883	0.891	0.797	0.638	0.641	0.629	0.616	0.654	0.661
PMNE-c	0.596	0.592	0.665	0.652	0.635	0.631	0.935	0.935	0.869	0.623	0.626	0.607	0.612	0.636	0.658
MNE	0.945	0.942	0.902	0.687	0.702	0.682	0.942	0.944	0.913	0.658	0.661	0.633	0.661	0.673	0.680
GATNE	0.982	0.987	0.953	0.873	0.877	0.792	0.962	0.947	0.915	0.782	0.795	0.786	0.718	0.741	0.735
DMGI	0.856	0.782	0.785	0.927	0.936	0.874	0.906	0.877	0.848	0.611	0.616	0.603	0.668	0.726	0.719
FAME	0.993	0.996	0.978	0.945	0.958	0.896	0.958	0.951	0.901	0.643	0.651	0.635	0.686	0.748	0.727
DualHGNN	0.975	0.978	0.967	/	/	/	/	/	/	/	/	/	/	/	/
HMGCN	**0.995**	**0.998**	**0.986**	**0.998**	**0.997**	**0.992**	**0.970**	**0.971**	**0.955**	**0.887**	**0.883**	**0.810**	**0.880**	**0.841**	**0.837**

HMGCN significantly outperforms all baseline models across various evaluation metrics on five real-world datasets. Specifically, when compared to the leading GNN baselines (i.e., HPN, FAME, and MAGNN), HMGCN achieves an average improvement of 10.75% in the F1 score across all datasets. Furthermore, HMGCN attains over 95% accuracy on three datasets: IMDB, Alibaba,

[7] https://github.com/pfnet/optuna

and Amazon. This superior performance can be attributed to HMGCN's utilization of graph convolutional networks that operate separately on different edge types, as well as its effective evaluation of multi type relations topological structures through an attention mechanism.

HMGCN has demonstrated superior performance compared to MAGNN and GATNE, underscoring its effectiveness relative to manually configured meta-path methodologies. It achieves significant performance improvements on widely used heterogeneous networks, such as DBLP and IMDB. This success can be attributed to HMGCN's proficient use of an attention mechanism, which assesses the importance of subgraphs (or layers)—a capability that traditional meta-path sampling methods lack.

5.5 Node Classification

In the context of node classification, we compare the performance of HMGCN against leading methodologies in the field. The outcomes are summarized in Table 4, where the most favorable outcomes are emphasized in bold. The initial eight baseline models correspond to unsupervised embedding techniques, whereas the subsequent models pertain to semi-supervised representation approaches.

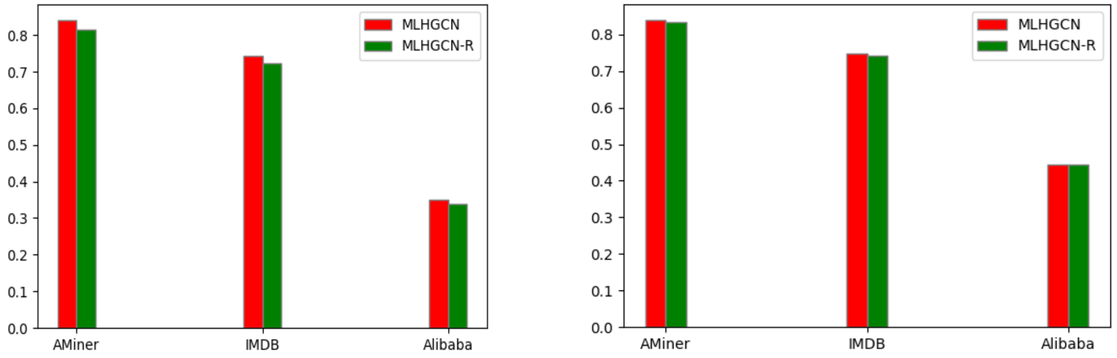
Table 4: Comparison of Various node classification methods performances on four real-world datasets

Method	**Alibaba**		**IMDB**		**DBLP**		**AMiner**	
	Macro-F1	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**	**Macro-F1**	**Micro-F1**
node2vec	0.237 (0.0126)	0.349 (0.0095)	0.364 (0.0238)	0.383 (0.0704)	0.353 (0.0104)	0.352 (0.0113)	0.523 (0.0033)	0.533 (0.0052)
RandNE	0.318 (0.0171)	0.357 (0.0094)	0.377 (0.0145)	0.393 (0.0186)	0.352 (0.0154)	0.373 (0.0151)	0.642 (0.0075)	0.672 (0.0064)
FastRP	0.302 (0.0181)	0.393 (0.0117)	0.367 (0.0235)	0.385 (0.0141)	0.344 (0.0202)	0.376 (0.0198)	0.651 (0.0087)	0.691 (0.0075)
MNE	0.287 (0.0156)	0.391 (0.0022)	0.375 (0.0154)	0.383 (0.0681)	0.367 (0.0118)	0.385 (0.0108)	0.644 (0.0068)	0.687 (0.0046)
GATNE	0.292 (0.0087)	0.391 (0.0015)	0.368 (0.0133)	0.334 (0.0006)	0.334 (0.0189)	0.375 (0.0045)	0.647 (0.0109)	0.647 (0.0114)
DMGI	0.221 (0.0215)	0.393 (0.0027)	0.549 (0.0191)	0.545 (0.0188)	0.782 (0.0304)	0.788 (0.0236)	0.474 (0.0156)	0.627 (0.0094)
FAME	0.324 (0.0155)	0.394 (0.0061)	0.594 (0.0136)	0.595 (0.0144)	0.843 (0.0184)	0.869 (0.0126)	0.723 (0.0115)	0.728 (0.0092)
DualHGNN	0.348 (0.0117)	0.403 (0.0128)	/	/	/	/	/	/
SGC	0.289 (0.0232)	0.362 (0.0176)	0.488 (0.0107)	0.564 (0.0134)	0.623 (0.0008)	0.624 (0.0008)	0.517 (0.0048)	0.588 (0.0158)
AM-GCN	0.306 (0.0233)	0.398 (0.0157)	0.611 (0.0022)	0.641 (0.0014)	0.868 (0.0106)	0.877 (0.0113)	0.703 (0.0176)	0.714 (0.0224)
R-GCN	0.268 (0.0325)	0.382 (0.0126)	0.545 (0.0173)	0.573 (0.0146)	0.863 (0.0054)	0.871 (0.0071)	0.691 (0.0077)	0.693 (0.0107)
HAN	0.276 (0.0325)	0.393 (0.0082)	0.553 (0.0113)	0.569 (0.0079)	0.807 (0.0079)	0.814 (0.0101)	0.691 (0.0148)	0.727 (0.0087)
NARS	0.294 (0.0202)	0.393 (0.0196)	0.566 (0.0038)	0.575 (0.0049)	0.795 (0.0256)	0.805 (0.0321)	0.723 (0.0104)	0.722 (0.0098)
MAGNN	0.347 (0.0484)	0.397 (0.0406)	0.615 (0.0074)	0.616 (0.0088)	0.882 (0.0285)	0.896 (0.0397)	0.756 (0.0106)	0.758 (0.0134)
HPN	0.265 (0.0347)	0.393 (0.0406)	0.579 (0.0024)	0.585 (0.0022)	0.823 (0.0202)	0.831 (0.0202)	0.711 (0.0613)	0.733 (0.0491)
GTN	0.256 (0.0421)	0.393 (0.0072)	0.616 (0.0109)	0.617 (0.0094)	0.853 (0.0138)	0.869 (0.0126)	0.704 (0.0181)	0.689 (0.0149)
HGSL	0.334 (0.0123)	0.394 (0.0238)	0.622 (0.0048)	0.636 (0.0031)	**0.895 (0.0283)**	**0.906 (0.0395)**	0.757 (0.0101)	0.756 (0.0104)
HMGCN	**0.350 (0.0156)**	**0.445 (0.0100)**	**0.744 (0.0106)**	**0.748 (0.0162)**	0.846 (0.0131)	0.859 (0.0126)	**0.848 (0.0147)**	**0.837 (0.0140)**

The HMGCN model exhibits enhanced performance on all assessed graphs, with the sole exception being the DBLP dataset. Specifically, HMGCN achieves an average enhancement of 13.75% and 12.15% in Micro-F1 and Macro-F1 scores, respectively, compared to the leading model, HGSL, across all datasets, excluding DBLP. Recent literature [6, 46] indicates that performance improvements in node classification typically range from 2% to 4%; however, the advancements realized by HMGCN are significantly greater. Additionally, HMGCN markedly outperforms similar models on widely used heterogeneous graphs featuring multiple node types (e.g., AMiner and IMDB), achieving improvements of 17.59% and 19.61% in Micro-F1 and Macro-F1 scores on the IMDB dataset. The model effectively learns node embeddings by employing an attention mechanism that evaluates the importance of subgraphs (or layers) associated with a particular node type. In contrast, existing heterogeneous graph representation models that rely on manual meta-path sampling often fail to adequately account for the importance of these relationship types. Nonetheless, HMGCN's performance on the DBLP dataset is suboptimal, as the author nodes (which possess class labels) belong to a single subgraph (or layer). This lack of diversity in the subgraphs (or layers) associated with a specific node type adversely affects the model's ability to derive effective node representations for those author nodes.

5.6 Ablation Study

To evaluate the efficacy of the model's components, we conducted supplementary experiments on variations of HMGCN. We propose that the HMGCN-R model does not prioritize the significance of distinct relationship types; rather, it employs mean aggregation to combine the embeddings of a specific node. The findings from the ablation study, conducted on three datasets for the purpose of node classification, are illustrated in Figure 2.



(a) by the Macro-F1 score

(b) by theMicro-F1 score

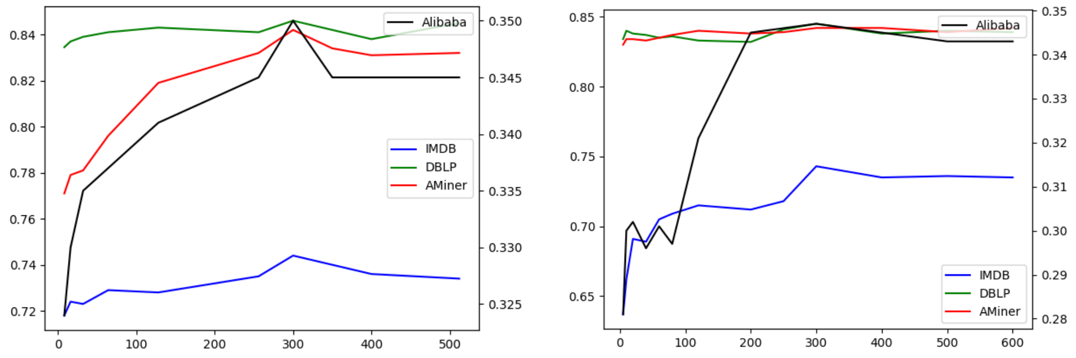
Figure 2: ablation study Experimental results

The findings indicate that the component significantly enhances the performance of the HMGCN model. A comparative analysis of HMGCN and HMGCN-R underscores the significance of acknowledging the various types of relationships in enhancing effectiveness. Notably, HMGCN demonstrates improvements of 3.31%, 2.96%, and 3.05% in Macro-F1 scores compared to HMGCN-R across the Alibaba, IMDB, and AMiner datasets, respectively. This underscores the critical role of the attention mechanism in effectively evaluating the significance of relationship types for learning node representations.

5.7 Parameters Experiments

We undertook a series of experiments to examine the hyperparameters associated with HMGCN, with particular emphasis on critical parameters including the number of training iterations and the hidden dimension for embeddings, represented as d. Figure 3 presents the outcomes of different parameter configurations across four real-world datasets in the context of the node classification task, evaluated using the Macro-F1 score. It is important to note that the results pertaining to Alibaba are depicted on the right vertical axis.

As illustrated in Figure 3(a), an initial increase in the embedding dimension correlates with enhanced performance; however, this trend ultimately reverses, resulting in a deterioration of outcomes. The setting of $d = 300$ achieved the highest prediction accuracy. This occurs because, when the dimension d is small, all node features are compressed into a limited representation space, which hampers the ability to preserve the proximity characteristics of all node pairs. In contrast, an increase in dimensionality results in the distances among the complete set of node representations to become flattened. Figure 3(b) illustrates the performance of HMGCN in relation to the training rounds number. We observe that HMGCN converges efficiently and quickly achieves stable performance across all tested datasets within 300 rounds.



(a) Macro-F1 score with respect to dimension d

(b) Macro-F1 score in relation to the number of training iterations

Figure 3: the investigation of the influence of hyperparameters in relation to dimensionality and the number of training iterations in the proposed methodology.

6 CONCLUSION

This study proposes an embedding model, HMGCN, for heterogeneous networks. HMGCN is not based on the selection of meta-paths. Instead, it decomposes the heterogeneous network into several bipartite and homogeneous sub-networks (or layers) according to edge types. HMGCN employs a graph convolutional module for each sub-network (or layer). Additionally, HMGCN utilizes an attention mechanism to assess the importance of the sub-networks (or layers) to which a particular node type belongs. The assessed importance of these sub-networks (or layers) is subsequently utilized to integrate various embeddings of the node, ultimately producing the final embedding of the node within the heterogeneous network. Empirical findings indicate that the proposed HMGCN outperforms existing methods in various graph representation tasks across five real heterogeneous networks, with the exception of cases where there is a lack of diversity in the subgraphs (or layers) associated with a specific node type.

Conflict of interest: On behalf of all authors, the corresponding author states that there is no conflict of interest.

REFERENCES

[1] Bahdanau, D., Cho, K., and Bengio, Y. (2014). “Neural machine translation by jointly learning to align and translate.” International Conference on Learning Representations.

[2] Cen, Y., Zou, X., Zhang, J., Yang, H., Zhou, J., and Tang, J. (2019). “Representation Learning for Attributed Multiplex Heterogeneous Network.” In KDD. 1358–1368.

[3] Chen, H., Sultan, S., F., Tian, Y., Chen, M., and Skiena, S. (2019). “Fast and Accurate Network Embeddings via Very Sparse Random Projection.” In CIKM. 399–408.

[4] Chen, H., Yin, H., Wang, W., Wang, H., Nguyen, Q., V., H., and Li, X. (2018). “PME: projected metric embedding on heterogeneous networks for link prediction.” In KDD. 1177–1186.

[5] Dong, Y., Chawla, N., V., and Swami, A. (2017). “metapath2vec: Scalable representation learning for heterogeneous networks.” In KDD. 135–144.

[6] Fu, X., Zhang, J., Meng, Z., and King, I. (2020). “Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding.” In WWW. 2331–2341.

[7] Grover, A. and Leskovec, J. (2016). “node2vec: Scalable feature learning for networks.” In KDD. ACM, 855–864.

[8] Hamilton, W., L., Ying, R., and Leskovec, J. (2017). “Inductive Representation Learning on Large Graphs.” In NeurIPS. 1025–1035.

[9] He, L., Chen, H., Wang, D., Jameel, S., Yu, P., and Xu, G. (2021). “Click-Through Rate Prediction with Multi-Modal Hypergraphs.” In CIKM. 690–699.

[10] He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., and Wang, M. (2020). “Lightgcn: Simplifying and powering graph convolution network for recommendation.” In SIGIR. 639–648.

[11] Hu, B., Fang, Y., and Shi, C. (2019). “Adversarial Learning on Heterogeneous Information Networks.” In KDD. 120–129.

[12] Hu, Z., Dong, Y., Wang, K., and Sun, Y. (2020). “Heterogeneous graph transformer.” In WWW. 2704–2710.

[13] Huang, C. (2021). “Recent advances in heterogeneous relation learning for recommendation.” arXiv preprint arXiv:2110.03455 (2021).

[14] Huang, C., Xu, H., Xu, Y., Dai, P., Xiao, L., Lu, M., Bo, L., Xing, H., Lai, X., and Ye, Y. (2021). “Knowledge-aware coupled graph neural network for social recommendation.” In AAAI.

[15] Ji, H., Wang, X., Shi, C., Wang, B., and Yu, P. (2021). “Heterogeneous Graph Propagation Network.” TKDE (2021).

[16] Jing, B., Park, C., and Tong, H. (2021). “Hdmi: High-order deep multiplex infomax.” In The Web Conference. 2414–2424.

[17] Kipf, T., N., and Welling, M. (2017). “Semi-Supervised Classification with Graph Convolutional Networks.” In ICLR.

[18] Lee, J., Lee, I., and Kang, J. (2019). “Self-Attention Graph Pooling.” Proceedings of the 36th International Conference on Machine Learning.

[19] Li, Z., Shen, X., Jiao, Y., Pan, X., Zou, P., Meng, X., Yao, C., and Bu, J. (2020). “Hierarchical bipartite graph neural networks: Towards large-scale e-commerce applications.” In ICDE. IEEE, 1677–1688.

[20] Liu, W., Chen, P., Yeung, S., Suzumura, T., and Chen, L. (2017). “Principled multilayer network embedding.” In ICDMW. IEEE, 134–141.

[21] Liu, Z., Huang, C., Yu, Y., and Dong, J. (2021). “Motif-preserving dynamic attributed network embedding.” In WWW. 1629–1638.

[22] Liu, Z., Huang, C., Yu, Y., Fan, B., and Dong, J. (2020). “Fast Attributed Multiplex Heterogeneous Network Embedding.” In CIKM. 995–1004.

[23] Long, X., Huang, C., Xu, Y., Xu, H., Dai, P., Xia, L., and Bo, L. (2021). “Social Recommendation with Self-Supervised Metagraph Informax Network.” In CIKM. 1160–1169.

[24] Lu, Y., Shi, C., Hu, L., and Liu, Z. (2019). “Relation structure aware heterogeneous information network embedding.” In AAAI. 4456–4463.

[25] Park, C., Kim, D., Han, J., and Yu, H. (2020). “Unsupervised Attributed Multiplex Network Embedding.” In AAAI. 5371–5378.

[26] Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). “DeepWalk: online learning of social representations.” In KDD. 701–710.

[27] Qiu, J., Dong, Y., Ma, H., Li, J., Wang, C., Wang, K., and Tang, J. (2019). “Netsmf: Large-scale network embedding as sparse matrix factorization.” In WWW. 1509–1520.

[28] Schlichtkrull, M., Kipf, T., N., Bloem, P., Rianne Van Den Berg, R., V., D., Titov, I., and Welling, M. (2018). “Modeling relational data with graph convolutional networks. In European semantic web conference.” Springer, 593–607.

[29] Shi, C., Hu, B., Zhao, W., X., and S Yu Philip, S., Y. (2018). “Heterogeneous information network embedding for recommendation.” TKDE 31, 2 (2018), 357– 370.

[30] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. (2015). “Line: Large-scale information network embedding.” In WWW. 1067–1077.

[31] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. (2018). “Graph Attention Networks.” In ICLR.

[32] Wang, X., Ji, H., Shi, C., Wang, B., Ye, Y., Cui, P., and Yu, P., S. (2019). "Heterogeneous Graph Attention Network." In WWW. ACM, 2022–2032.

[33] Wang, X., Lu, Y., Shi, C., Wang, R., Cui, P., and Mou, S. (2020). "Dynamic heterogeneous information network embedding with meta-path based proximity." TKDE (2020).

[34] Wang, X., Zhu, M., Bo, D., Cui, P., Shi, C., and Pei, J. (2020). "Am-gcn: Adaptive multi-channel graph convolutional networks." In KDD. 1243–1253.

[35] Wei, W., Huang, C., Xia, L., Xu, Y., Zhao, J., and Yin, D. (2022). "Contrastive meta learning with behavior multiplicity for recommendation." In KDD. 1120–1128.

[36] Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., et al. (2019). "Simplifying Graph Convolutional Networks." In ICML. 6861–6871.

[37] Wu, Y., Lian, D., Xu, Y., Wu, L., and Chen, E. (2020). "Graph convolutional networks with markov random field reasoning for social spammer detection." In AAAI, Vol. 34. 1054–1061.

[38] Xia, L., Huang, C., Xu, Y., Dai, P., Zhang, B., and Bo, L. (2020). "Multiplex Behavioral Relation Learning for Recommendation via Memory Augmented Transformer Network." In SIGIR. 2397–2406.

[39] Xue, H., Yang, L., Rajan, V., Jiang, W., Wei, Y., and Lin, Y. (2021). "Multiplex bipartite network embedding using dual hypergraph convolutional networks." In WWW. 1649–1660.

[40] Yu, L., Shen, J., Li, J., and Lerer, A. (2020). "Scalable graph neural networks for heterogeneous graphs." arXiv preprint arXiv:2011.09679 (2020).

[41] Yun, S, Jeong, M., Kim, R., Kang, J., and Kim, H., J. (2019). "Graph Transformer Networks." In NeurIPS. 11960–11970.

[42] Yu, P., Fu, C., Yu, Y., Huang, C., Zhao, Z., and Dong, J. (2022). "Multiplex Heterogeneous Graph Convolutional Network." In KDD '22. 2377 - 2387.

[43] Zhang, C., Song, D., Huang, C., Swami, A., and Chawla, N. V. (2019). "Heterogeneous graph neural network." In KDD. 793–803.

[44] Zhang, H., Qiu, L., Yi, L., and Song, Y. (2018). "Scalable multiplex network embedding." In IJCAI, Vol. 18. 3082–3088.

[45] Zhang, Z., Cui, P., Li, H., Wang, X., and Zhu, W. (2018). "Billion Scale Network Embedding with Iterative Random Projection." In ICDM. 787–796.

[46] Zhao, J., Wang, X., Shi, C., Hu, B., Song, G., and Ye, Y. (2021). "Heterogeneous Graph Structure Learning for Graph Neural Networks." In AAAI.