MIRROR: Mining Implicit Relationships via Structure-Enhanced Graph Convolutional Networks

JIAYING LIU, Dalian University of Technology
FENG XIA and JING REN, Federation University Australia
BO XU, Dalian University of Technology
GUANSONG PANG, Singapore Management University
LIANHUA CHI, La Trobe University

Data explosion in the information society drives people to develop more effective ways to extract meaningful information. Extracting semantic information and relational information has emerged as a key mining primitive in a wide variety of practical applications. Existing research on relation mining has primarily focused on explicit connections and ignored underlying information, e.g., the latent entity relations. Exploring such information (defined as implicit relationships in this article) provides an opportunity to reveal connotative knowledge and potential rules. In this article, we propose a novel research topic, i.e., how to identify implicit relationships across heterogeneous networks. Specially, we first give a clear and generic definition of implicit relationships. Then, we formalize the problem and propose an efficient solution, namely MIRROR, a graph convolutional network (GCN) model to infer implicit ties under explicit connections. MIRROR captures rich information in learning node-level representations by incorporating attributes from heterogeneous neighbors. Furthermore, MIRROR is tolerant of missing node attribute information because it is able to utilize network structure. We empirically evaluate MIRROR on four different genres of networks, achieving state-of-the-art performance for target relations mining. The underlying information revealed by MIRROR contributes to enriching existing knowledge and leading to novel domain insights.

CCS Concepts: • **Information systems → Web mining;**

Additional Key Words and Phrases: Relation mining, implicit relationships, graph convolutional networks, heterogeneous networks

ACM Reference format:
Jiaying Liu, Feng Xia, Jing Ren, Bo Xu, Guansong Pang, and Lianhua Chi. 2023. MIRROR: Mining Implicit Relationships via Structure-Enhanced Graph Convolutional Networks. *ACM Trans. Knowl. Discov. Data.* 17, 4, Article 55 (February 2023), 24 pages.
https://doi.org/10.1145/3564531

This work is partially supported by National Natural Science Foundation of China under Grant No. 61872054 and No. 72204037.

Authors' addresses: J. Liu, School of Economics and Management, Dalian University of Technology, Dalian, Liaoning 116024, China; email: jiaying_liu@outlook.com; F. Xia (corresponding author) and J. Ren, Institute of Innovation, Science and Sustainability, Federation University Australia, Ballarat, VIC 3353, Australia; emails: f.xia@acm.org, ch.yum@outlook.com; B. Xu, School of Software, Dalian University of Technology, Dalian, Liaoning 116620, China; email: boxu@dlut.edu.cn; G. Pang, School of Computing and Information Systems, Singapore Management University, 188065, Singapore; email: gspang@smu.edu.sg; L. Chi, School of Engineering and Mathematical Sciences, La Trobe University, Melbourne, VIC 3086, Australia; email: l.chi@latrobe.edu.au.

1556-4681/2023/02-ART55 $15.00
https://doi.org/10.1145/3564531

1 INTRODUCTION

Understanding the relations between entities is essential for obtaining new insights from a wide variety of practical applications. For instance, in the field of biomedicine, understanding the relationships among proteins, complexes, and drugs helps a lot in drug discovery. For online shopping scenario, we can inference one's personal preference by his/her reviews and ratings. However, due to the inherent characteristics of data, it is difficult to get such relation information. On one hand, comparing with the colorful physical networks, the data are always still black-and-white, which means that entities are merely unlabeled among complex connections. On the other hand, the underlying networks from the data are usually ambiguous and incomplete [48]. It is difficult to extract useful information in the huge amount of data.

To solve the above problems, researchers are committed to studying relation mining, which can be understood as detecting and classifying of semantic relationship mentions within a set of artifacts. In the specific network scenarios, the basic motivation is to assign the proper labels of specific relationships for the connections among entities based on the relationship identification techniques. Existing studies about relation mining are mainly divided into two categories based on two kinds of information: users' behavior information and network structure. For example, Liu et al. [23] proposed a method named Shifu2 to extract the advisor-advisee relationship based on the scholars' academic attributes by using network embedding. Rotabi et al. [28] tried to detect strong ties in the social network using a special network structure named motif. Based on the fact that exploring the effects of relationships can actually improve the performance of data mining tasks [21], relation mining plays an important role in various fields of research both theoretically and practically, such as user profiling [40], recommendation [50], and semantic proximity search [24].

However, current studies on relation mining have primarily focused on surface connections (explicit relationships) between entities. Actually, much information hides beyond these explicit relationships, which is essential for obtaining new knowledge. Hence, in this article, we define a novel class of relationships called implicit relationships, that hides in the explicit connections. Implicit relationships are very common in practical applications. For instance, in a scientific collaboration network where collaborations between scholars are explicit relationships, we may discover that two scholars collaborate because one is the advisor of the other, or they belong to the same research group. The advisor-advisee relationship and colleague relationship in the collaboration network can be regarded as implicit relationships. From the perspective of national security, we can construct a terrorist attack network, where edges indicate that attacks occur in the same location. According to the explicit connections in terms of occurrence, we can infer organizations that launched the same attacks, which is called the implicit relationship. Thus, we can predict the location of the next attack and take some security measures in advance. In the online social network, we can judge whether users share a common preference (the implicit relationship) based on their communication (the explicit relationship), which will provide useful guidelines for recommendation systems.

In summary, if explicit relationships refer to links existing explicitly between entities, we regard the underlying relations that can be inferred by analyzing entities' characteristics and surface communication as implicit relationships. Identifying and understanding implicit relationships contributes to discovering connotative knowledge and revealing potential rules. It is also conducive to understand the formation and evolution of the network society. Exploring such relations is crucial to many significant applications, such as double-blind peer review, reviewer recommendation, and national security.

In comparison with explicit relationships (represented as edges in a network), which are usually homogeneous, implicit relationships are diverse. It means that we can discover different implicit

relationships hidden in an explicit relationship. At the same time, implicit relationships are usually invisible (cannot be obtained directly in the explicit network), which increases the complexity of extracting such relations. To tackle these problems, in this work, we investigate to develop a generalized framework, which can obtain implicit relationships in heterogeneous networks. Actually, up to now, there is no clear definition of implicit relationships. We study by considering the definition of implicit relationship in different domains. We precisely define the problem and propose an efficient solution named MIRROR based on graph learning [47], to be specific, the **graph convolutional network** (**GCN**) model. To evaluate the performance of MIRROR, we conduct a series of experiments on different genres of networks. The results illustrate that the proposed model has great abilities to model the network and capture the target relation. The main contributions of our work are threefold:

— **Problem Definition.** We formally define a novel class of relationships, called implicit relationship, which widely covers a series of real-world issues. To the best of our knowledge, we are the first to give a clear and generic definition of implicit relationship identification problem. In addition, we also explain the differences between explicit relationships and implicit relationships in detail.
— **Efficient Solution.** We formulate a novel and computationally efficient framework, namely MIRROR, to extract implicit relationships in heterogeneous networks. MIRROR extends traditional GCN model to heterogeneous networks and aggregates feature information from nodes' heterogeneous neighbors. It is practical in a large-scale data environment, especially when information of the target itself is limited, but some information can be obtained in heterogeneous neighbors.
— **Flexible Scalability.** The aforementioned framework can be well applicable to different types of networks and supports various application scenarios. In addition, it is suitable for not only networks with node attributes but also for networks without any attribute information. MIRROR can also achieve good performance while only depending on link connection information.

The remaining part of the article proceeds as follows: Section 2 introduces related work from the perspectives of relation mining and heterogeneous **network representation learning** (**NRL**) methods. Section 3 formally formulates the problem of implicit relationship identification. Section 4 discusses our optimal solution named MIRROR and describes its architecture in detail. The evaluation results are clarified in Section 5. Finally, Section 6 concludes and promises future directions for this work.

2 RELATED WORK

The main issues addressed in this article are related to two broader fields of research, including relation mining and heterogeneous NRL methods. The following gives a brief overview of each topic, respectively.

2.1 Relation Mining

The research into relation mining constitutes a very active field [22, 27, 42]. There are many efforts devoted to inferring ties in different networks, especially in social networks [49]. As shown in Figure 1, in the field of computer science, the focuses of studies in relation mining include relationship formation mechanism (link prediction), relationships semanticization (relation type prediction), and relationship-based interaction (relation interaction prediction).

Link prediction aims to recommend and predict unknown links in the network. In an undirected network $G = (V, E)$, where V is the set of nodes and E is the link set, the universal set $U = \frac{|V|\cdot|V-1|}{2}$

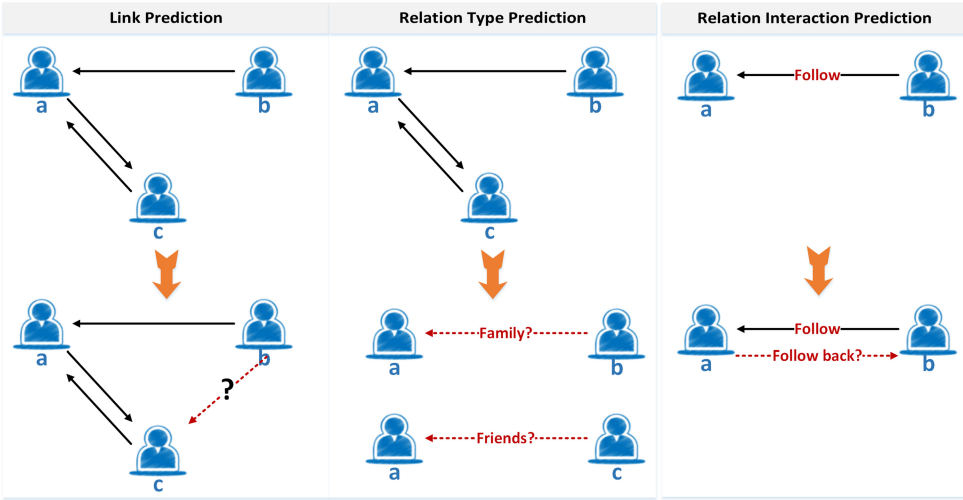


Fig. 1. The focuses of studies in relation mining include link prediction, relation type prediction, and relation interaction prediction.

contains all possible links. Then the set of nonexistent links can be represented as $U - E$. The task of link prediction is to determine these links, including both existing unknown links and future links. Liben-Nowell and Kleinberg [20] studied the link prediction problem systematically in the social network. They used several unsupervised learning methods to solve the problem based on the similarity of users. In biomedical networks, Peng et al. [26] proposed a **Cluster-based Steiner Tree Miner** (**CST-Miner**) to quickly identify the multi-entity topological relationships in biological networks. Existing research on link prediction mainly utilizes the similarity between users' content and network structure to solve the problem. Most of the existing methods are suitable for unweighted networks, and only a few can be extended to weighted networks [51].

Relation type prediction tries to recognize the semantics involved in each relation automatically. The ultimate goal is to infer the type of relation between nodes. For instance, Cen et al. [3] inferred trust relationships by incorporating type-based dyadic and triadic correlations into a graphic model. However, designing a flexible model for different types of networks is the major problem in relation type prediction. Another challenge is that with the increase of network complexity, the inference accuracy is not satisfactory yet. In addition, the increase of network scale makes most of the links unlabeled. How to solve the sample imbalance problem is also worth studying.

The main purpose of relation interaction prediction is to study how a one-way relation develops into the two-way relationship. Wang et al. [41] studied the formation probability of conference closure in academic networks. In comparison with link prediction and relation type prediction, relation interaction prediction is more difficult. Firstly, relation interaction usually occurs in a dynamic environment. The high dynamics of the network challenges the relation prediction. Secondly, there are few datasets available for relation interaction. The research scope is limited to social networks and academic social networks.

Relation mining techniques can be classified into two types according to the features they used: entity attributes based methods and network structure based methods.

Attributes based methods. This type of method takes advantage of entity characteristics. They take attribute information generated from nodes in the network as input. Considering the local properties of entities helps a lot in constructing validated representations of the network.

Liu et al. [23] identified the advisor-advisee relationship by adopting a deep learning architecture. However, the growth of data makes it difficult to extract features precisely and completely.

Network structure based methods. In contrast to attributes based methods, network structure information is more available and reliable. Network structure based approaches extract a good summary of network topology and then calculate a similarity score. Generally speaking, if the structural similarity can capture the characteristics of the target network well, the approaches will achieve better performance. Rotabi et al. [28] tried to detect strong ties in the social network using a special network structure named motif.

However, there are too many possible entity attributes and structural features of networks. It is challenging to make a good summary according to various features, especially in heterogeneous networks with complex structures. How to design a general model to identify different relationships in heterogeneous networks is worth studying. Tracing the development of relation mining techniques, the main existing difficulties for developing a universal model are:

(1) **No common features.** Maybe there exists no common features or even no interactions between different networks. It is difficult to apply attributes based methods on other different networks directly. How to transfer methods from one type of network to another is a crucial issue.

(2) **Unbalanced network scale.** The scale of different networks varies greatly. The performance may be not good enough if we use a network as the training set and another network as the test set directly.

Our work is related to link prediction problem and relation type prediction in heterogeneous networks. We use both node attributes and network structure to form a robust solution for the task.

2.2 Heterogeneous Network Representation Learning

Most networks are heterogeneous with multiple types of nodes and edges, which contain more and richer semantic information than homogeneous networks. Heterogeneous NRL methods solve the heterogeneity of heterogeneous networks. They aim at embedding the structural and semantic information of heterogeneous networks into the low-dimensional vectors so as to facilitate the application of downstream tasks. Generally speaking, heterogeneous NRL methods [15, 35] can be divided into two classes, including network-structure based embedding methods and graph neural network methods. Compared with the embedding methods based on network structure, the graph neural network methods obtain the node embedding representations by aggregating attribute information of both nodes and their neighbors.

Network-structure based embedding methods only use the similarity of network topology to generate node embeddings [12]. According to the ways of obtaining similarity, it can be divided into random-walk based methods [46] and first or second-order similarity based methods. Metapath2vec [5] uses meta-path based random walk and skim-gram to solve the heterogeneity problem, but only considers a single meta path. HERec [30] uses the constraint strategy to filter node sequences and capture complex semantics in heterogeneous networks. The heterogeneous network is transformed into a homogeneous network through symmetric meta-paths, and then the objective function is optimized according to Node2vec to learn embeddings under a single meta-path. Different from Metapath2vec and HERec, HIN2vec [8] randomly selects the walking nodes as long as the nodes are connected. Then HIN2vec simplifies the relationship between nodes as a binary classification problem. Metapath2vec, HERec, and HIN2vec are three early works of heterogeneous NRL, which provide the reference for future work such as MCRec [14], RHINE [25], and HeGAN [13].

First or second-order similarity based methods including PTE [36], AspEm [31], and HEER [32] learn the node representations by capturing the first-order or second-order similarity of the

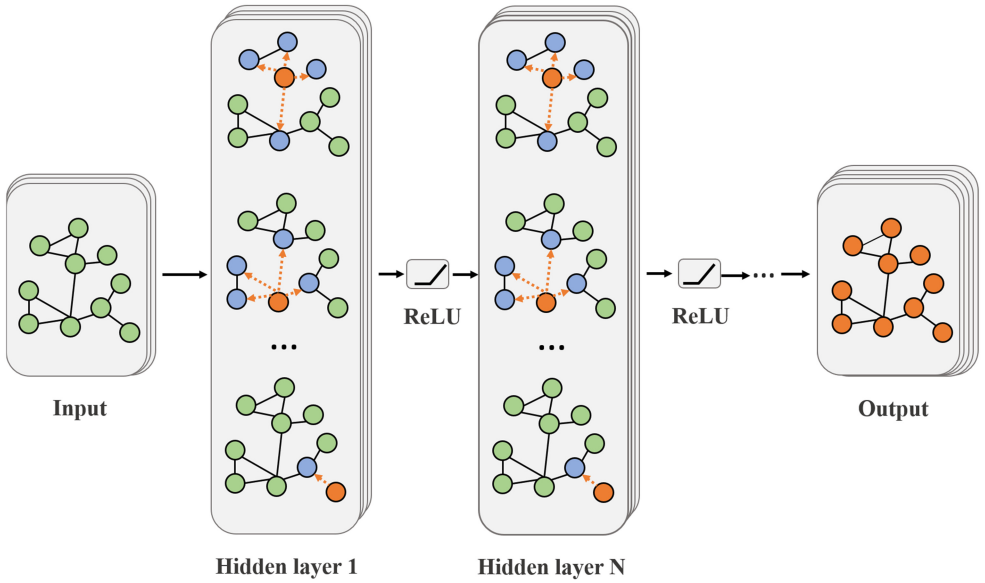


Fig. 2. The main computation process of graph convolutional networks.

network. PTE first decomposes the heterogeneous network into a group of bipartite graph networks, and then learns the representations by capturing the second-order similarity of bipartite graph networks similar to LINE. AspEm captures the incompatibility of the heterogeneous network by decomposing the input into multiple aspects and designs a measurement method of incompatible information to select the appropriate aspect for embeddings. Shi et al. improves AspEm and establishes edge representations based on node representations. Different metric spaces are used for different types of edges. The embeddings for nodes and edges are learned by combining the edge representations and different metric spaces.

In recent years, with the development of graph neural networks [47], GCNs and graph attention networks have been widely used in heterogeneous NRL. Most heterogeneous NRL methods need to specify the meta-path in advance, which requires strong prior knowledge. The choice of meta-path will greatly affect the model. Yun et al. [52] propose the **graph transformation network (GTNs)** that can generate a new graph data structure, aiming to solve the problem of appropriate meta-path chosen in heterogeneous networks. GTNs establishes based on GCNs (see Figure 2). GTNs convert the original graph into a new graph through the GT layer, and then perform convolution operations on the new graph to learn effective node representations. The GT layer can identify useful connections between unconnected nodes in the original graph data, and generate useful meta-paths.

Wang et al. [43] first introduce the attention mechanism into the heterogeneous graph neural network, and propose a heterogeneous graph neural network model named HAN based on hierarchical attention. HAN realizes node-level aggregation and semantic-level aggregation through the attention mechanism. However, it does not consider the internal structure of the meta-path. Zhang et al. [53] propose HetGNN that considers the structural information of the heterogeneous network and the heterogeneous content information of each node at the same time. Fan et al. [7] introduces a principled approach for jointly capturing interactions and opinions in the user-item graph and uses the attention mechanism to differentiate the heterogeneous strengths of social relations to coherently model graph data.

Most representation learning methods based on meta-paths discard the node information of the meta-path, and only consider the start and end nodes, resulting in information loss. Fu et al. [9] propose the MAGNN model. MAGNN first maps the node attribute information of the heterogeneous network to the same space, then uses the attention mechanism to integrate the semantic information of the meta-path. Finally, the attention mechanism is used to aggregate multiple meta-path information for final node representations. DisenHAN [44] uses meta relations to decompose high-order connectivity between node pairs and proposes a disentangled embedding propagation layer to learn disentangled user/item representations from different aspects. However, as real-world networks become more complex, how to effectively aggregate the rich information in the network while reducing the complexity of the model is a direction for future research.

3 PROBLEM FORMULATION

In this section, we first define the concept of implicit relationship in different networks. Then, we describe the notations used in this article. Finally, we formulate the task of implicit relationship identification in heterogeneous networks using the above definitions.

3.1 Definition of Implicit Relationship

We can regard explicit relationships as active, superficial interactions between entities in the given network. That is, entities are connected directly in terms of edges in a network based on their direct communication. However, the underlying causes and mechanisms of these surface connections remain unexplored. Although some studies have proposed the concept of "implicit relationship" [33, 34, 45], they assume that the co-occurrence relationship between two target entities is implicit. Actually, the definition of implicit relation should be more extensive. The implicit relationship can be considered as possible latent relations that could be inferred with the help of the current state of knowledge. Usually, implicit relationships are obtained through related models by analyzing entity behavior. For a given network $G = (V, E)$, where V is the set of nodes and E is the set of edges. In this article, the explicit relationship r_e and the implicit relationship r_i are defined as

Definition 1 (Explicit Relationship). The explicit relationship is the relationship represented by the edges between entities in the network. In the network $G = (V, E)$, the explicit relationship r_e is the relationship represented by e_{ij} in G.

Definition 2 (Implicit Relationship). The implicit relationship refers to the latent possible relations between entities that are hidden in explicit relationships. The implicit relationship r_i is related with p_i where p_i is the probability (confidence) obtained by an algorithm for inferring the relationship. r_i is hidden in the network G, which is derived by fusing entity feature X, exogenous knowledge set K, and explicit link connection information M_e, where $r_i \neq r_e$.

Although the implicit relationship is invisible, it is rife, particularly in social networks. For instance, the advisor-advisee relationship could be considered as an implicit relationship in a scientific collaboration network, where the collaboration represents the explicit relationship. Another example is the friendship in an online communication network where the online conversation represents the explicit relationship. In contrast to explicit connections (relationships), implicit relationships cannot be observed directly via the edges in the network. The differences between implicit relationships and explicit relationships are:

(1) Implicit relationships are often invisible in the network. In contrast, explicit relationships can be observed directly from real connections. Specifically, implicit relationships are not

represented as edges in the network. Thus, they need to be extracted based on entities' behavior and surface communication with relation mining techniques.

(2) The explicit relationship between entities must exist in the network. However, there is not always an implicit relationship existing between two nodes connected by an explicit relationship.

Here, we take the academic collaboration network as an example. If scholar A and B have co-authored one article, they will be connected in the collaboration network. The collaboration relationship is considered as an explicit relationship. Suppose A is B's research supervisor (i.e., advisor) in real life, in which case they will normally co-author some article(s). However, the advisor-advisee relationship cannot be observed directly from those explicit co-authorship links (i.e., collaboration network). We regard this kind of relationship as the implicit relationship.

Implicit relationships can be obtained by analyzing the characteristics of entities' behavior and surface communication. Mining implicit relationships provide interesting insights for understanding the underlying architecture of the network, modeling the formation and evolution in the connections of entities, and exploring the underlying causes for network dynamics.

It is worth mentioning that relations that are implicit in one setting may be explicit in another. For example, the advisor-advisee relationship in the coauthor networks is implicit. However, in a knowledge base, it may be explicit. From the definition, we can see that there is no absolute demarcation line between the two concepts "explicit" and "implicit" about relation, which is just considered differently in different scenarios. We focus on mining implicit relations given a clear scenario for the explicit relations.

3.2 Mathematical Preliminaries

In this article, we focus on mining the implicit relationship in the undirect heterogeneous network $G = (V, E, R)$, where G contains nodes with an object type mapping function $\psi : V \rightarrow \mathcal{T}$. It means that each node v_i belongs to a particular type $\psi(v_i) \in \mathcal{T}$. Similarly, G also contains edges with the object type mapping function $\phi : E \rightarrow R$. Each edge $e \in E$ connects nodes with different relation types $r \in R$, which could be described as $\phi(e) \in R$. $A_t \in \mathbb{R}^{n_t \times m_t}$ preserves attribute information for each node in the network, where m_t represents node attribute categories and n_t is the total number of nodes of type $t \in \mathcal{T}$ in the network. Note that V, E, R, and T are finite and disjoints. Table 1 lists the meaning of the notations mainly used throughout this article.

3.3 Problem Definition

Based on the notations explained above, we can formulate the task of implicit relationship identification as: How can we infer the implicit relationships between targets v_i and v_j with explicit connections under relation type r? Given an explicit network, the goal is to detect the unknown relationships (implicit relationship) in the heterogeneous network. More precisely, the problem is formulated as

Problem 1 (Implicit relationship mining). Given a heterogeneous network $G = (V, E, R)$, entity feature X, exogenous knowledge set K, and explicit link connection information M_e, the objective is to learn a predictive function $g : G = (V, E, R, X, K, M_e) \rightarrow r_i$.

As shown in Figure 3, the input of the task consists of a heterogeneous academic network G composed of multiple types of nodes (organizations (o), scholars (s), and publications (p)). If scholar s_1 collaborates with s_2 on a joint article p and belongs to the organization o, s is connected with p and o in G. Nodes are related to the attribute matrix A_t. The ultimate goal is to infer the implicit relationship (advisor-advisee relationship) between s and his/her collaborators, which cannot be

Table 1. Description of Main Notations

Notation	Description
V	the set of nodes in the heterogeneous network
E	the set of edges in the heterogeneous network
$\|V\|$	the total number of nodes
$e_{i,j}$	the edge from v_i to v_j
$r \in R$	relation type
$\mathcal{T}$	node type
n_t	the total number of nodes of type t
m_t	the number of node attribute categories for nodes with type t
$M_r \in \mathbb{R}^{n_{t1} \times n_{t2}}$	link information matrix under relation type r
$A_t \in \mathbb{R}^{n_t \times m_t}$	initial node attribute information matrix for nodes of type t
$M_t \in \mathbb{R}^{n_t \times n_t}$	link connection information between nodes with type t
N_r	the set of neighbors of the node with the relation type r
W_t^l	the importance of nodes of type t in the lth layer
h_d	the dimension-specific representation in dimension d
z_i	the final representation of v_i
$Z \in \mathbb{R}^{n_t \times d}$	the final representations matrix
W_r^l	the weight matrix of relation r in lth layer of MIRROR
E_p	the positive samples
E_n	the negative samples
n_p	the number of positive samples
n_n	the number of negative samples
$M_p \in \mathbb{R}^{n_p \times n_p}$	the adjacency matrix with the positive relationship
$M_n \in \mathbb{R}^{n_n \times n_n}$	the adjacency matrix with the negative relationship
$f(v_i, v_j)$	the feature of edge e_{v_i, v_j}
n_{eg}	the dimension of the edge feature
$A_e \in \mathbb{R}^{n_t \times n_{eg}}$	the edge connection feature matrix

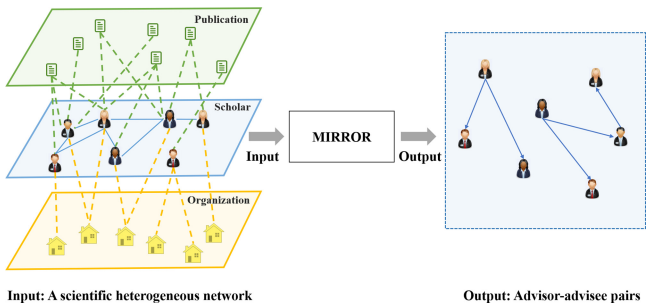


Fig. 3. An example of implicit relationship identification in the heterogeneous academic collaboration network.

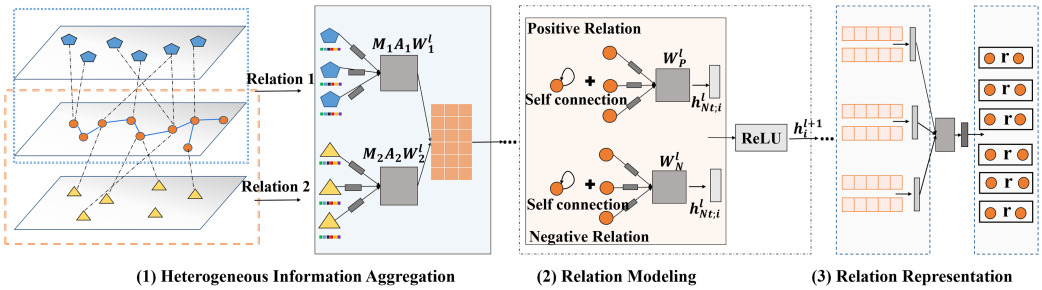


Fig. 4. The framework of MIRROR, which contains three critical components: (1) Heterogeneous information aggregation, (2) Relation modeling, (3) Relation prediction.

observed directly from the initial network. Thus, we need to learn a predictive function to infer the relationship.

4 MIRROR: IMPLICIT RELATIONSHIPS IDENTIFICATION FRAMEWORK

The proposed method is established based on the non-linear, multi-layer convolutional graph neural network. It operates directly on the heterogeneous network G. As shown in Figure 4, our model primarily consists of three components, including (1) Heterogeneous information aggregation, (2) Relation modeling, and (3) Relation prediction. The following illustrates the details for each part of the proposed model.

4.1 Heterogeneous Information Aggregation

To perform the heterogeneous information aggregation, we take advantage of a graph encoder and utilize the information generated from multiple types of nodes. The main idea is to transform and propagate information captured by neighbors' features across the heterogeneous network.

In the process of aggregating heterogeneous information for targets, we consider the edge (relation) type between targets and their neighbors, and process each relation type individually. If A_t preserves attribute of targets' neighbor under type t, and M_t preserves link connection information between them, then the matrix of heterogeneous information aggregates over $|\mathcal{T}|$ types of neighbors can be represented as

$$H = \sum_{t=1}^{\mathcal{T}} M_t A_t W_t. \tag{1}$$

The weight matrix W_t models the importance of neighbors under type t to targets. The process of heterogeneous information aggregation is summarized as Algorithm 1.

ALGORITHM 1: Heterogeneous Information Aggregation Algorithm.

Input: Link matrices $M_1, M_2, \ldots, M_t$; Nodes attribute matrices $A_1, A_2, \ldots, A_t$
Output: Generated node representations H for the target nodes

1: **Initiate**: Randomly initiate $W_1, W_2, \ldots, W_t$
2: **Train**:
3: **for** $j = 1$ to t **do**
4: Row-wisely normalize M_j
5: **end for**
6: Concatenate $H \leftarrow M_1A_1W_1 + M_2A_2W_2 + \ldots + M_tA_tW_t$
7: **Return**: H_T

4.2 Relation Modeling

R-GCN is the first model to apply the GCN framework to relational data modeling. It can efficiently aggregate information from locally structured neighbors and encode features, enabling classification, link prediction, and other tasks with significant performance improvements. Techniques for sharing parameters and implementing sparse constraints, including basic and block decomposition of the weight matrix are introduced in R-GCN and can be applied to multiple graphs with a large number of relationships. Experiments show that using GCN with factorization to form the auto-encoder can improve the effectiveness of factorization models on link prediction, which is 29.8% higher than the baselines. Hence we select the messaging framework for relation modeling.

After heterogeneous information aggregation, the next step is to model relations by using GCNs. The node feature vectors generated from heterogeneous neighbors can capture how information transform and propagate across the graph. To perform relation modeling, we adopt an encoder that takes node feature vectors as input and generates embedding for each node. Motivated by previous work [6, 18], the convolution process that operates directly on graphs can be understood as a special case of the message-passing framework:

$$h_i^{l+1} = \sigma\left(\sum_{m \in MI_i} g_p\left(h_i^l, h_j^l\right)\right), \tag{2}$$

where $h_i^l \in R^{d^{(l)}}$ is the state of node v_i, represented as a d-dimensional vector in the lth hidden layer. MI_i is the set of the incoming message of node v_i, represented as the identical edges types in the network. σ aggregates incoming messages in terms of $g_p(.,.)$, and transforms it through a specific element-wise activation function.

Previous studies have illustrated the effectiveness of the transformation in terms of accumulating and encoding features from local neighbors. Inspired by this architecture, in the proposed model, we consider the first-order neighbors for a given node and adopt the propagation model as mentioned before. Taking the relation type into account, the forward-pass update of the target node v_i across the heterogeneous graph is

$$h_i^{l+1} = \sigma\left(\sum_{r \in R} \sum_{j \in N_r^i} g_p\left(h_i^l, h_j^l\right)\right), \tag{3}$$

where N_r^i is the set of v_i's neighbors under the relation type $r \in R$. Self-connections are necessary in the model to maintain the features of the node. Here, we adopt the linear transformation $g_p(h_i^l, h_j^l) = Wh_j$ with a weight matrix W as the propagation model. Thus, Equation (2) can be rewritten as

$$h_i^{l+1} = \sigma\left(\sum_{r \in R} \sum_{j \in N_r^i} c_{ij}^r W_r^l h_j^l + W_r^l h_i^l\right), \tag{4}$$

where c_{ij}^r is the specific normalization constant. We choose $c_{ij}^r = 1/\sqrt{|N_i^r||N_j^r|}$ in advance which is derivated from symmetric normalized Laplacian matrix.

Equation (4) accumulates transformed feature vectors by normalizing the sum of representations. In this part, we consider not only neighbors' features but also relation types. Moreover, we add a single self-connection of each relation type. Thus, node representations in the $(l+1)$-th layer can be transformed and passed by the representations in the lth layer. We then establish a deep model by chaining the multiple layers with an element-wise activation function, i.e., $ReLU(.) = max(0; .)^2$. To initialize the procedure, the input general representations are necessary.

Table 2. Description of Link Connection Attributes

Feature	Description	Calculation formula
$d(v_i)$	the degree of node v_i	
$d(N_{v_i})$	the degree of node v_i's neighbors	
$s(v_i, v_j)$	the common neighbors of v_i and v_j	$s(v_i, v_j) = \lvert N_{(v_i)} \bigcap N_{(v_j)} \rvert$
$Adamic(v_i, v_j)$	Admaic-Adar index, which considers the degree information of the common neighbors	$Adamic(v_i, v_j) = \sum_{v_z \in N_{(v_i)} \bigcap N_{(v_j)}} \frac{1}{\log d(v_z)}$
$Jaccard(v_i, v_j)$	Jaccard index, which is used to describe the degree of dissimilarity	$Jaccard(v_i, v_j) = \frac{\lvert N_{(v_i)} \bigcap N_{(v_j)} \rvert}{\lvert N_{(v_i)} \bigcup N_{(v_j)} \rvert}$
$PA(v_i, v_j)$	Preferential attachment, which describes the preferential attachment similarity between two nodes	$PA(v_i, v_j) = d(v_i) \times d(v_j)$

The initial general representations could be features associated with the nodes $h_i^0 = x_i$, or unique one-hot vectors (if there are no features). Then $(l+1)$th layer takes the output of the lth layer as input. The final representation of v_i is computed as $z_i = h_i^K$.

4.3 MIRROR with Network Structure

Although applying external information such as node attributes can achieve good performance, it is very difficult to obtain complete attribute information in many cases. Furthermore, it is also difficult to guarantee the reliability of node attributes. To remedy these problems, instead of only considering node attributes, we incorporate link attributes, i.e., network structure characteristics into the model. Following the route proposed by Li et al. [19], we try to concatenate node and link attributes in this part. In the process of incorporating link attributes, the main idea is to learn a weight for each edge based on link connection information.

In this article, we compute the similarity of two connected nodes (i.e., degree of nodes and common neighbors) as link connection attributes. In general, one of the most common similarity indices is common neighbors (CN). Furthermore, we also consider other indices such as Admaic-Adar Index [1], Jaccard Index [16], and Preferential Attachment Index (PA) [2]. Table 2 lists a more detailed description of link connection attributes.

Next, we will elaborate on how to embed link attributes into MIRROR. After calculating the link connection features $f(e_{v_i,v_j})$ of edge e_{v_i,v_j}, the weight $\lambda_{i,j}$ for e_{v_i,v_j} can be adapted as

$$\lambda_{i,j}^l = \sigma(U^l \cdot f(e_{v_i,v_j}) + b^l), \tag{5}$$

where U^l is the parameter matrix and b is the bias vector in the neural network. Equation (5) evaluates the significance of the neighbor $v_j \in N^i$ to v_i. Our guideline is to make good use of different attributes. We will adopt different solutions for different circumstances. Four kinds of solutions are provided for different cases:

- **Only consider heterogeneous information.** It means that we generate the target node representations only based on information aggregated from their heterogeneous neighbors. The specific procedure refers to Algorithms 1 and 2.
- **Only consider link connection information.** When we only consider link connection information, we only use one-hot embeddings for obtaining H_T in Algorithm 1.
- **Consider both heterogeneous information and link connection information.** In this case, after the process of heterogeneous information aggregation, we calculate the link

weight matrix according to Equation (5) and let it multiply the adjacency matrices of positive samples and negative samples.

4.4 Relation Prediction

So far, we have embedded each target node $v_i \in V$ to a real-valued vector representation. Relying on learning representations, the third part of the proposed model aims to reconstruct labeled edges in G. We try to solve this problem based on the idea of traditional link prediction problem, which deals with predicting the relation r of new triples $(source, object)$ represented as (s, r, o) in a labeled graph G. It is implemented by assigning scores for (s, r, o) to decide the probability of edge (s, o) belonging to G through an element-wise function $\varphi(s, r, o)$.

In line with this perspective, in this task, we try to predict the candidate edge (v_i, r, v_j) through a score function g: $R^d \times R \times R^d \longrightarrow R$ by using learning representations. The factorized operation can be described as

$$g(v_i, r, v_j) = z_i^T R z_j. \tag{6}$$

The probability of the edge (v_i, r, v_j) existing is calculated by a sigmoid function ϕ :

$$\phi = \frac{1}{1 + \exp(-x)}. \tag{7}$$

Then the probability p_{ij}^r of an edge existing between node v_i and node v_j under type r is modeled as

$$\begin{aligned} p_{ij}^r &= p((v_i, r, v_j) \in R) = \phi(g(v_i, r, v_j)) \\ &= \frac{1}{1 + \exp(-g(v_i, r, v_j))} \\ &= \frac{1}{1 + \exp(-z_i^T R z_j)} \end{aligned}, \tag{8}$$

where R is a trainable diagonal parameter matrix that models global entity interactions across the implicit relation.

4.5 Model Training

During the training process, we adopt negative sampling for lower calculation complexity [38]. In detail, for each labeled edge (v_i, v_j) (the positive sample) in the partially labeled network, we randomly choose node v_n, which is not connected with v_i in the original graph through a sampling distribution. We regard (v_i, v_n) as the negative sample. In particular, for the specific triple (v_i, r, v_j), by replacing v_j with v_n, the parameters are optimized by calculating the cross-entropy loss:

$$l(i, j) = -\log p_{ij}^r - E_{n \sim \mathrm{Pr}_j} \log(1 - p_{in}^r). \tag{9}$$

Considering all samples, the final loss function L can be represented as

$$\begin{aligned} L = \sum_{(v_i, r, v_j) \in \tau} l(i, j) = -\frac{1}{(1 + \omega)|\hat{E}|} \\ \sum_{(v_i, r, v_j, v_n) \in \tau} \alpha \log p_{ij}^r + (1 - \alpha) \log(1 - p_{in}^r) \end{aligned}, \tag{10}$$

where τ is the total set of positive and negative triples. $\hat{E}$ is the training set. α is a constant parameter and is defined as

$$\alpha = \begin{cases} 0, & (v_i, v_j) \text{ is the postive sample} \\ 1, & (v_i, v_n) \text{ is the negative sample.} \end{cases} \tag{11}$$

To optimize the model, we adopt a mini-batch **Adaptive Moment Estimation** (**Adam**) [17]. Furthermore, we use mini-batch gradient descent for training. More specifically, we divide the training set into multiple batches and calculate the loss for each batch to update parameters, which accelerates the convergence velocity of iteration. Taking advantage of fixed contributions to the loss function, the training process is faster with avoiding results falling into local optimum. At the same time, this operation does not need to load all training data into the memory, which helps to avoid exhausting the memory resource. The overall architecture of the algorithm is summarized as Algorithm 2.

ALGORITHM 2: The Implicit Relationship Mining Algorithm.

Input: link matrices $M_1, M_2, \ldots, M_t$; nodes attribute matrices $A_1, A_2, \ldots, A_t$; edge connection feature matrix A_e; r; f_r; convergence condition ξ
Output: label for each edge $e = (v_i, v_j)$

PRE-TRAIN:
Generate positive samples E_p and negative samples E_n
Calculate the link weight matrix L_e according to Equation (5)
Calculate adjacency matrix M_p for E_p, M_n for E_n
Add self-connection I to matrices $M_p \leftarrow M_p + I$, $M_n \leftarrow M_n + I$
if Only consider the heterogeneous information **then**:
    Calculate adjacency matrix M_p, M_n directly
else
    $M_p = M_p * L_e$, $M_n = M_n * L_e$
end if
Generate H_T according to Algorithm 1:
if Only consider link connection information **then**:
    Calculate H based on one-hot embeddings
else
    Calculate H according to Algorithm 1 directly
end if
INITIATE:
Randomly initiate weight W_p^0, W_p^1, W_n^0, W_n^1, U, b
Row-wisely normalize M_p, M_n
TRAIN:
while ξ is not true **do**
    Randomly generate a batch of data from M_p, M_n
    Calculate $R_1 = \mathbf{Relu}(M_P H W_p^0 + M_n H W_n^0)$
    Calculate $R_2 = \mathbf{Relu}(M_P R_1 W_p^1 + M_n R_2 W_n^1)$
    Calculate L_{sum} based on Equation (10)
    Update W_p^0, W_p^1, W_n^0, W_n^1, M_p, M_n, R_1, R_2, U, b and M_j in Algorithm 1
end while

4.6 Model Complexity Analysis

The proposed model first generates the representation for each heterogeneous entity. The complexity of generating the embedding matrix is $O(cn_adI)$, where c is the sample size, n_a is the maximum of the attribute dimension, d is the largest dimension of the hidden layer, and I is the iteration time. For edge reconstruction, taking advantages of matrix transformation, which substitutes the Laplacian matrix for feature decomposition, the computational complexity becomes $O(n)$ for each

Table 3. Summary Statistics of Datasets

Dataset	Epinions	Publication	Terrorist Attacks	Enron
No. of nodes	10,000	7,872	1,293	151
No. of edges	336,262	8,282	1,648	17,786
No. of positive relationships	60,310	2,787	571	141
No. of relations	3	3	2	2
Explicit Relation	users' co-rating relationship	authors' co-author relationship	attacks' co-location relationship	user communication relationship
Identified Implicit Relation	users' trust/distrust relationship	authors' advisor-advisee relationship	same organization organized relationship	manager-subordinate relationship

node. Thus, the total computational complexity of the second step is $O(mn)$, where m is the attribute dimension and n is the sample size. It can, therefore, be concluded that the computational complexity of MIRROR is $O(mn + cn_a dI)$. If we consider the link connection information, of which computational complexity is $O(m_2 n)$ (m_2 is the dimension for link connection), the total computational complexity becomes $O(mn + m_2 n + cn_a dI)$.

5 EXPERIMENT

In this section, we validate the effectiveness of MIRROR by conducting implicit relationship mining tasks on four real-world networks. We first present the datasets and baselines used for this article. Then, we describe implicit relationship mining tasks with discussions of experimental results. Further experiments are conducted to illustrate the importance of node attributes and network structure in MIRROR.

5.1 Experimental Setup

5.1.1 Datasets. We regard the problem of mining implicit relationship in heterogeneous networks as a multi-relation link prediction task. To evaluate the performance of the proposed model, we perform experiments on four different datasets, including Epinions,[1] Publication,[2] Terrorist Attacks,[3] and Enron.[4] Table 3 provides the summary statistics for the datasets. The details of the datasets are illustrated as follows.

— **Epinions.** Epinions provides users' reviews for products. We form a heterogeneous network based on relations between users and their ratings. The features of users include their profiles. Each rating is associated with its category, score, time point, and helpfulness. We aim at identifying the implicit relationship represented as the trust and distrust relationship between users. Besides, we regard distrust relations as positive samples because distrust relations are less common than the trust relation (e.g., number of users in one's blacklist is generally less than the number of his/her friends).

[1] https://www.cse.msu.edu/~tangjili/trust.html.
[2] https://www.openacademic.ai/oag/.
[3] https://linqs-data.soe.ucsc.edu/public/lbc/TerroristRel.tgz.
[4] https://www.aminer.cn/data-sna.

— **Publication.** We use **Microsoft Academic Graph** (**MAG**), which contains publication-related information to construct a publication-scholar-organization heterogeneous network. We try to infer advisor-advisee relationships in the academic collaboration network. Two scholars are connected if they have collaborated with each other. The ground truth advisor-advisee pairs are extracted from The Academic Family Tree in the fields of Computer Science. We have collected 2,787 advisor-advisee pairs as positive samples.
— **Terrorist Attacks.** The terrorist attacks dataset consists of two types of information related to entities (terrorism attacks), i.e., the attributes of entities and the links that connect various entities. The dataset is provided by Zhao et al. [54]. We have constructed an attack-location heterogeneous network. Besides, the terrorist attacks are connected if they take place in the same places. There is an implicit relationship between two attacks if they are organized by the same organization.
— **Enron.** We construct a heterogeneous e-mail communication network based on the Enron e-mail dataset. The users are connected with the e-mails. The features of users include their profiles. Each e-mail is associated with its sender, recipient, time, type, and subject. We aim at identifying the implicit relationship represented as the manager-subordinate between users.

5.1.2 Comparison Algorithms. MIRROR adopts GCN to learn the representations for destination nodes in heterogeneous networks. Thus, we compare MIRROR with NRL models and relational GCNs to evaluate the performance of MIRROR. To demonstrate the ability of MIRROR in heterogeneous information aggregation, we also test the performance of a MIRROR variant, MIRROR-noa, implemented by setting the same weights for heterogeneous neighbors. We compare our approach with the following baselines:

— **LINE** [37]. LINE preserves both local and global structures by using the edge-sampling algorithm. It can eliminate the limitation of the classical stochastic gradient descent in both weighted and unweighted networks.
— **Node2vec** [10]. Node2vec is a semi-supervised method, which aims to capture the diversity of connection patterns. For each node in the network, it learns a low-dimensional representation by optimizing a neighbor preserving objective.
— **R-GCN** [29]. R-GCN extends GCN to modeling relational data. The learning process can be considered as a coding-decoding process.
— **MIRROR-noa**. MIRROR-noa is a variant of our proposed method. In MIRROR-noa, we simply take the same weights over all heterogeneous entities when generating the representations for target nodes. The weight matrix $w \in W_t$ in the process of heterogeneous information aggregation (Equation (1)) has the same value.
— **HetGCN** [53]. HetGCN is designed for attribute network embedding, which uses meta-path guide random walks to aggregate neighborhood information.
— **GATNE** [4]. GATNE is embedding model for a large-scale heterogeneous graphs with multi-edges. The framework is successfully deployed on the recommendation systems of Alibaba.
— **GAT** [39]. GAT can solve the problems of GCN. GAT introduces masked self-attentional layers to improve the shortcomings of GCN. It assigns corresponding weights to different nodes. At the same time, it does not require matrix operations and the graph structure in advance.
— **GraphSAGE** [11]. GraphSAGE is a typical spatial-based GCN. It aims to optimize the sampling of the entire graph to the sampling of the current neighbor node.

In all the above comparison methods, LINE, Node2vec, GAT, and GraphSAGE are designed for homogeneous networks and R-GCN, GATNE, HetGCN are designed for heterogeneous networks. R-GCN is based on GCN, which is similar to our model. HetGCN uses meta-path guide random

walks to aggregate neighborhood information. GATNE acquires a wealth of attribute information and makes use of the multiple topologies (connection information) of the different nodes. MIRROR aggregates heterogeneous information in a different way to these methods. MIRROR takes advantage of a graph encoder and utilizes the information generated from multiple types of nodes. At the same time, these baselines do not take into account network structural similarities and do not apply to attribute-free networks. In the process of parameter setting for each baseline, we select the parameters with the best performance for fair comparison.

5.1.3 Evaluation Criteria. The performance is determined by the average of the **Area Under Curve** (**AUC**) and **Ranking Score** (**RS**). Generally speaking, AUC is equal to the probability that a classifier will rank a randomly selected instance in the test set (positive one) higher than a randomly selected non-existing one (negative one). More specifically, if E^n is the set of the negative instances and E^p is positive set, we randomly choose one link from E^p and another from E^n. If the edge score of E^t is higher than that of E^n, we add 1 point to the final score. If two scores are equal, add 0.5 points. While we carry out n comparisons independently, of which there are n_1 times E^p having a higher score than E^n, and n_2 times are equal, then AUC is calculated as

$$AUC = \frac{n_1 + 0.5n_2}{n}. \tag{12}$$

Apparently, higher AUC inherently indicates better performance.

RS considers the edges' position of the final ranking in E^p. If $E^u = U - E^p$ is the set of unlabeled links and r_e represents the rank of edge e, then:

$$RS_e = \frac{r_e}{|E^u|}. \tag{13}$$

Traversing all edges in E_p, RS can be derived by

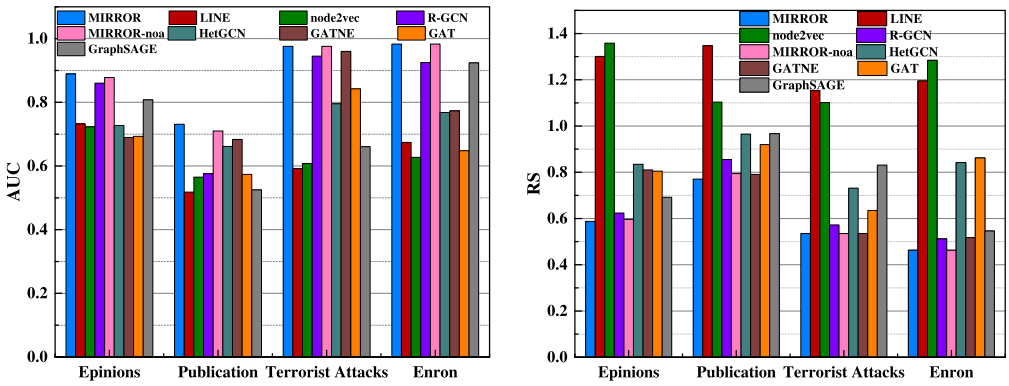
$$RS = \frac{1}{|E^p|} \sum_{e \in E^p} RS_e = \frac{1}{|E^p|} \sum_{e \in E^p} \frac{r_e}{|E^u|}. \tag{14}$$

Obviously, lower RS is regarded as better performance.

5.2 Results and Analysis

To evaluate the performance of MIRROR, we have carried out the experiments from three aspects, i.e., effectiveness evaluation, ablation study, and parameter sensitivity. To verify the effectiveness of MIRROR, we compare it with state-of-the-art methods mentioned in Section 5.1.2. The results are presented in Section 5.2.1. To investigate the effectiveness of relation modeling, we have proposed three solutions, including only considering heterogeneous information, only considering link connection information, and considering both heterogeneous information and link connection information (see Section 5.2.2). In Section 5.2.3, we analyze the influence of different parameter settings including batch sizes and the structure of GCN layers.

5.2.1 Effectiveness Evaluation. We first compare the performance of MIRROR to alternative baselines. Figure 5 shows that the proposed method performs more effectively than other baselines, which supports the effectiveness of our model for implicit relationship mining. Moreover, methods designed for heterogeneous networks, including R-GCN, MIRROR, HetGCN, GANTE, and MIRROR-noa outperform LINE and node2vec significantly on all datasets. This is because simply aggregating attributes of different entities can lead to information loss. In addition, GraphSAGE has also achieved better results on the Epinions and Enron datasets. However, in the other two datasets, the network embedding algorithms designed for heterogeneous networks perform well. In these algorithms, MIRROR outperforms all the baselines on all datasets for two possible



(a) AUC (b) RS

Fig. 5. Performance comparison of the implicit relationship mining task in terms of AUC and RS on each dataset.

reasons: (1) We not only utilize node attributes but also effectively fuse structural features into the feature embedding process. The local structural similarity is important for embedding and the proposed method can facilitate them well to learn better representations. (2) It is also necessary to learn different weights for different types of relationships in the learning process. Overall, MIRROR allows us to consider more complicated interactions between different entities by propagating to each other not only neighbor features under the same relation type but also information across different types of neighbors.

The average performance over different training ratio is reported in Table 4. Different results are presented according to when 40%, 50%, 60%, 70%, and 80% of training samples are labeled. On average, MIRROR outperforms any other baselines. It achieves significantly good performance even though the labeled portion is small. Notably, there exist no significant differences between MIRROR and MIRROR-noa on the Terrorist Attacks and Enron datasets because there are only two types of nodes in the dataset. Thus, relations have the same weight in the process of information aggregation. On the Terrorist Attacks dataset, GATNE and R-GCN perform better, with the highest AUC exceeding 90%. On the Epinions dataset, R-GCN and GraphSAGE have performed better. On the Enron dataset, MIRROR and MIRROR-noa have the same performance, with GraphSAGE and R-GCN being second only to them. In general, the heterogeneous network embedding methods perform better than the homogeneous network embedding methods. The performance of each method also improves with the increase of training samples.

5.2.2 Ablation Study. The performance of MIRROR with different inputs including only node attributes, only link connection attributes, and both nodes and link connection attributes are presented in Table 5 in terms of AUC and RS. The training ratio represents the proportion of training samples to total samples. The results illustrate that MIRROR can accurately identify the target relationship whether or not node attributes or link connection attributes are used as input. Under the condition that the node attributes cannot be obtained, if we only use link attributes for training, the highest AUC of MIRROR is up to 92%. In order to achieve the best results under different datasets, the input of the model needs to be adjusted. We believe that one of the possible reasons is that different identification tasks require different pieces of information. At the same time, the varied sizes of networks may also lead to this phenomenon. For some cases, node attribute information is more important, while for others, the connection structure is more important.

Table 4. Performance Comparison with Different Training Ratios on Each Dataset

Dataset	**Epinions**									
Index	AUC					RS				
Method \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
MIRROR	**0.88**	**0.887**	**0.886**	**0.893**	**0.889**	0.612	**0.614**	**0.615**	**0.588**	**0.616**
LINE	0.732	0.73	0.722	0.711	0.694	1.415	1.301	1.426	1.352	1.401
node2vec	0.723	0.723	0.719	0.711	0.697	1.423	1.441	1.432	1.358	1.409
R-GCN	0.842	0.845	0.849	0.86	0.85	0.648	0.650	0.650	0.624	0.653
MIRROR-noa	0.874	0.878	0.875	0.881	0.878	0.619	0.621	0.621	0.596	0.623
HetGCN	0.704	0.712	0.715	0.723	0.727	0.843	0.842	0.855	0.828	0.834
GATNE	0.712	0.717	0.702	0.696	0.689	**0.525**	0.782	0.797	0.803	0.810
GAT	0.619	0.637	0.657	0.689	0.693	0.881	0.864	0.840	0.812	0.804
GraphSAGE	0.839	0.743	0.738	0.828	0.808	0.661	0.757	0.763	0.675	0.692
Dataset	**Publication**									
Index	AUC					RS				
Method \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
MIRROR	**0.701**	**0.708**	**0.725**	**0.711**	**0.731**	**0.806**	**0.789**	**0.794**	**0.784**	**0.770**
LINE	0.505	0.508	0.511	0.518	0.516	1.388	1.361	1.360	1.359	1.347
node2vec	0.554	0.534	0.565	0.548	0.557	1.115	1.142	1.104	1.118	1.131
R-GCN	0.56	0.559	0.571	0.565	0.576	0.867	0.856	0.871	0.855	0.875
MIRROR-noa	0.697	0.7	0.706	0.696	0.71	0.814	0.802	0.812	0.807	0.795
HetGCN	0.632	0.630	0.657	0.657	0.661	0.967	0.954	0.958	0.946	0.965
GATNE	0.659	0.667	0.634	0.688	0.683	0.814	0.807	0.839	0.806	0.790
GAT	0.621	0.587	0.615	0.638	0.573	0.851	0.896	0.876	0.830	0.919
GraphSAGE	0.608	0.598	0.532	0.556	0.524	0.899	0.905	0.968	0.942	0.967
Dataset	**Terrorist Attacks**									
Index	AUC					RS				
Method \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
MIRROR	**0.95**	**0.961**	**0.96**	**0.958**	**0.976**	**0.557**	**0.551**	**0.553**	**0.541**	**0.535**
LINE	0.594	0.571	0.592	0.564	0.566	1.252	1.194	1.493	1.177	1.153
node2vec	0.593	0.608	0.588	0.567	0.578	1.207	1.189	1.148	1.155	1.101
R-GCN	0.916	0.916	0.908	0.93	0.945	0.597	0.590	0.584	0.572	0.588
MIRROR-noa	**0.95**	**0.961**	**0.96**	**0.958**	**0.976**	**0.557**	**0.551**	**0.553**	**0.541**	**0.535**
HetGCN	0.772	0.764	0.763	0.775	0.795	0.742	0.736	0.755	0.745	0.731
GATNE	0.916	0.888	0.939	0.942	0.960	0.582	0.610	0.558	0.554	0.535
GAT	0.858	0.794	0.896	0.878	0.842	0.640	0.719	0.591	0.623	0.634
GraphSAGE	0.598	0.766	0.637	0.653	0.660	0.898	0.759	0.813	0.834	0.831
Dataset	**Enron**									
Index	AUC					RS				
Method \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
MIRROR	**0.963**	**0.975**	**0.974**	**0.983**	**0.964**	**0.478**	**0.466**	**0.473**	**0.475**	**0.463**
LINE	0.612	0.579	0.583	0.632	0.674	1.254	1.195	1.243	1.264	1.251
node2vec	0.625	0.613	0.627	0.586	0.573	1.332	1.415	1.368	1.379	1.284
R-GCN	0.885	0.912	0.907	0.917	0.925	0.512	0.526	0.524	0.536	0.517
MIRROR-noa	**0.963**	**0.975**	**0.974**	**0.983**	**0.964**	**0.478**	**0.466**	**0.473**	**0.475**	**0.463**
HetGCN	0.745	0.753	0.757	0.768	0.761	0.876	0.842	0.854	0.864	0.857
GATNE	0.772	0.764	0.773	0.768	0.772	0.517	0.543	0.646	0.642	0.713
GAT	0.612	0.648	0.623	0.625	0.631	0.887	0.892	0.862	0.875	0.885
GraphSAGE	0.912	0.915	0.894	0.924	0.916	0.579	0.576	0.546	0.573	0.581

Bold numbers indicate the best values in the experiments.

Table 5. The Performance of MIRROR with Different Inputs

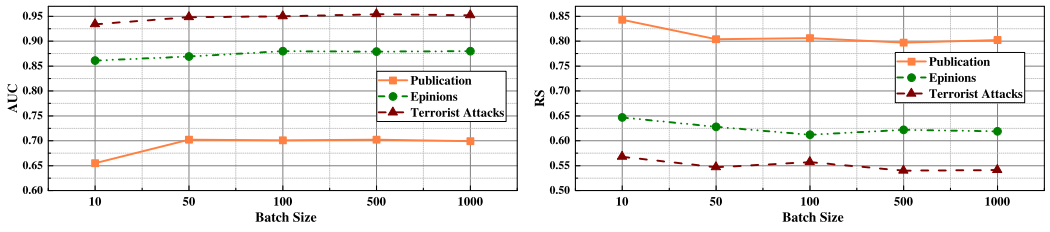
Dataset	**Publication**									
Index	AUC					RS				
Input \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
only node attributes	0.701	0.708	0.725	0.711	**0.731**	0.806	0.789	0.794	0.784	**0.770**
only link connection attributes	0.56	0.55	0.56	0.57	0.57	0.879	0.885	0.869	0.857	0.862
node+link attributes	0.65	0.64	0.63	0.62	0.62	0.849	0.859	0.865	0.876	0.877
Dataset	**Epinion**									
Index	AUC					RS				
Input \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
only node attributes	0.88	0.887	0.886	0.893	0.889	0.612	0.614	0.615	**0.588**	0.616
only link connection attributes	0.89	**0.89**	**0.89**	0.88	**0.89**	0.600	0.608	0.613	0.590	0.609
node+link attributes	0.88	**0.89**	0.88	0.87	0.88	0.611	0.615	0.616	0.602	0.626
Dataset	**Terrorist Attacks**									
Index	AUC					RS				
Input \ Percentage	40%	50%	60%	70%	80%	40%	50%	60%	70%	80%
only node attributes	0.95	0.961	0.96	0.958	0.976	0.557	0.551	0.553	0.541	0.535
only link connection attributes	0.92	0.93	0.93	0.92	0.93	0.582	0.566	0.563	0.576	0.563
node+link attributes	0.96	0.96	0.97	0.95	**0.98**	0.539	0.533	0.529	0.544	**0.518**

Bold numbers indicate the best values in the experiments.

The acquisition of link attributes only needs to calculate the relevant indicators of the network, without other external information. Although the performance of the model will be improved when the external information is added, it is difficult to obtain a large amount of external information, and the computational complexity will also increase. Therefore, it is more universal to model relationships using only network structural information. In general, the model can better capture the relation features when considering external information.

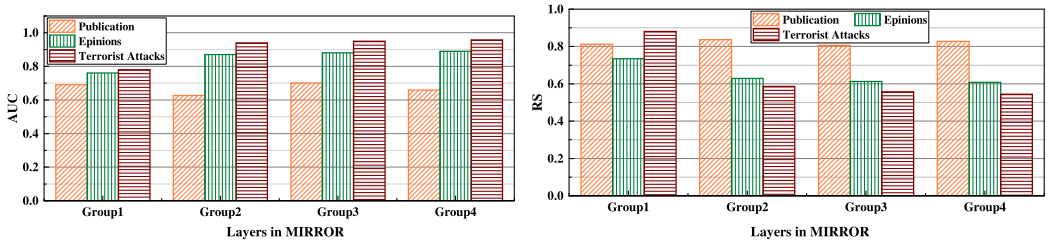
5.2.3 Parameter Sensitivity. In this part, we aim at presenting an in-depth analysis to explore how different parameters influence the performance of MIRROR. Two factors i.e., batch sizes and the layers of GCN architecture, are considered.

Batch size decides the number of samples in each mini batch. Generally speaking, the larger batch size makes the descent direction more accurate. At the same time, the shock will also decrease. However, too many samples in a batch lead to local optimum. On the contrary, too small batch size introduces more randomness and makes it difficult to converge. Figures 6(a) and 6(b) demonstrate the performance of MIRROR varying with different batch sizes. It can be seen that when the batch size is increased from 10 to 50, the model performance is steadily improved, indicating that an appropriate increase in the value of the batch size can make the gradient descent more accurate. At the same time, the performance of the model has also improved. When the batch size continues to increase, the model performance has been improved but at a modest rate. Especially when the batch size is increased from 500 to 1,000, the performance of the model decreases. We



(a) AUC (b) RS

Fig. 6. The performance of MIRROR in terms of AUC and RS with different batch size.



(a) AUC (b) RS

Fig. 7. The performance of MIRROR in terms of AUC and RS with different GCN layers.

can conclude that MIRROR performance tends to be stable when there are at least 50 samples in a batch.

GCN related models have hierarchical mechanisms as deep learning models. It means that extracted features become more abstract and advanced layer by layer. We set the network structure of MIRROR as follows:

(1) Group 1: GCN (64);
(2) Group 2: GCN (512)→GCN (64);
(3) Group 3: GCN (512)→GCN (128)→GCN (64);
(4) Group 4: GCN (512)→GCN (256)→GCN (128)→GCN (64).

The results are presented in Figure 7. With the increase of GCN layers, the performance of the model improves in both social networks and academic networks, especially on the dataset "Terrorist Attacks". However, with the increase of the number of training layers, the training complexity of the model also increases. MIRROR with settings in Group 3 achieves better performance because it returns better performance on the dataset "Publication". Under the settings of Groups 2, 3, and 4, there is little difference in AUC and RS between the other two datasets. Generally, MIRROR can obtain better performance under the settings of Groups 3 and 4. It is also the case that Group 4 produces a model that is significantly more complex than Group 3.

6 CONCLUSION

In this article, we develop MIRROR, a novel approach for identifying implicit relationships in heterogeneous networks. Particularly, we put forward a clear and generic definition of the implicit relationship in heterogeneous networks. MIRROR is able to capture information from heterogeneous neighbors over the entire graph. It predicts implicit associations between targets by using a graph convolutional architecture. MIRROR achieves excellent performance on the destination task. The underlying information revealed by MIRROR contributes to enriching existing knowledge and leading to novel domain insights.

Implicit relationships extracted by MIRROR can be applied to many real-world applications. For example, by identifying implicit relationships among scholars, we can evaluate the scholars' impact more accurately by removing relational citations. In addition, we can recommend reviewers to avoid fake or relational peer reviews. From the perspective of national security, security measures can be implemented in advance if we know the relationship between terrorist organizations. The ability to understand implicit relationships would also be beneficial to many practical real-world applications that rely on the knowledge extracted from knowledge graphs.

ACKNOWLEDGMENTS

The authors would like to thank Shiyu Han, Xin Chen, and Lei Wang at Dalian University of Technology for help with experiments.

REFERENCES

[1] Lada A. Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social Networks* 25, 3 (2003), 211–230.
[2] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
[3] Yukuo Cen, Jing Zhang, Gaofei Wang, Yujie Qian, Chuizheng Meng, Zonghong Dai, Hongxia Yang, and Jie Tang. 2019. Trust relationship prediction in alibaba e-commerce platform. *IEEE Transactions on Knowledge and Data Engineering* 32, 5 (2019), 1024–1035.
[4] Yukuo Cen, Xu Zou, Jianwei Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Representation learning for attributed multiplex heterogeneous network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1358–1368.
[5] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 135–144.
[6] David K. Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Proceedings of the Advances in Neural Information Processing Systems*. 2224–2232.
[7] Wenqi Fan, Yao Ma, Qing Li, Jianping Wang, Guoyong Cai, Jiliang Tang, and Dawei Yin. 2022. A graph neural network framework for social recommendations. *IEEE Transactions on Knowledge and Data Engineering* 34, 5 (2022), 2033–2047.
[8] Tao-yang Fu, Wang-Chien Lee, and Zhen Lei. 2017. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1797–1806.
[9] Xinyu Fu, Jiani Zhang, Ziqiao Meng, and Irwin King. 2020. Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding. In *Proceedings of the Web Conference 2020*. 2331–2341.
[10] Aditya Grover and Jure Leskovec. 2016. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 855–864.
[11] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 1025–1035.
[12] Mingliang Hou, Jing Ren, Da Zhang, Xiangjie Kong, Dongyu Zhang, and Feng Xia. 2020. Network embedding: Taxonomies, frameworks and applications. *Computer Science Review* 38 (2020), 100296.
[13] Binbin Hu, Yuan Fang, and Chuan Shi. 2019. Adversarial learning on heterogeneous information networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 120–129.
[14] 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 *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1531–1540.
[15] Chao Huang, Huance Xu, Yong Xu, Peng Dai, Lianghao Xia, Mengyin Lu, Liefeng Bo, Hao Xing, Xiaoping Lai, and Yanfang Ye. 2021. Knowledge-aware coupled graph neural network for social recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4115–4122.
[16] Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull Soc Vaudoise Sci Nat* 37 (1901), 547–579.
[17] D. P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 7-9, 2015.
[18] T. N. Kipf and M. Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, Toulon, France, April 24 - 26, 2017.

[19] Z. Li, L. Zhang, and G. Song. 2019. GCN-LASE: towards adequately incorporating link attributes in graph convolutional networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 2959–2965.

[20] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology* 58, 7 (2007), 1019–1031.

[21] Jiaying Liu, Xiangjie Kong, Xinyu Zhou, Lei Wang, Da Zhang, Ivan Lee, Bo Xu, and Feng Xia. 2019. Data mining and information retrieval in the 21st century: A bibliographic review. *Computer Science Review* 34 (2019), 100193.

[22] Jiaying Liu, Jing Ren, Wenqing Zheng, Lianhua Chi, Ivan Lee, and Feng Xia. 2020. Web of scholars: A scholar knowledge graph. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2153–2156.

[23] Jiaying Liu, Feng Xia, Lei Wang, Bo Xu, Xiangjie Kong, Hanghang Tong, and Irwin King. 2021. Shifu2: A network representation learning based model for advisor-advisee relationship mining. *IEEE Transactions on Knowledge and Data Engineering* 33, 4 (2021), 1763–1777.

[24] Zemin Liu, Vincent W. Zheng, Zhou Zhao, Zhao Li, Hongxia Yang, Minghui Wu, and Jing Ying. 2018. Interactive paths embedding for semantic proximity search on heterogeneous graphs. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1860–1869.

[25] Yuanfu Lu, Chuan Shi, Linmei Hu, and Zhiyuan Liu. 2019. Relation structure-aware heterogeneous information network embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4456–4463.

[26] Jiajie Peng, Linjiao Zhu, Yadong Wang, and Jin Chen. 2019. Mining relationships among multiple entities in biological networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 17, 3 (2019), 769–776.

[27] Jing Ren, Feng Xia, Xiangtai Chen, Jiaying Liu, Mingliang Hou, Ahsan Shehzad, Nargiz Sultanova, and Xiangjie Kong. 2021. Matching algorithms: Fundamentals, applications and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence* 5, 3 (2021), 332–350.

[28] Rahmtin Rotabi, Krishna Kamath, Jon Kleinberg, and Aneesh Sharma. 2017. Detecting strong ties using network motifs. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 983–992.

[29] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the European Semantic Web Conference*. Springer, 593–607.

[30] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S. Yu Philip. 2018. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 31, 2 (2018), 357–370.

[31] Yu Shi, Huan Gui, Qi Zhu, Lance Kaplan, and Jiawei Han. 2018. Aspem: Embedding learning by aspects in heterogeneous information networks. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 144–152.

[32] Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. 2018. Easing embedding learning by comprehensive transcription of heterogeneous information networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2190–2199.

[33] Min Song, Nam-Gi Han, Yong-Hwan Kim, Ying Ding, and Tamy Chambers. 2013. Discovering implicit entity relation with the gene-citation-gene network. *PloS One* 8, 12 (2013), e84639.

[34] Andreas Spitz, Satya Almasian, and Michael Gertz. 2017. EVELIN: Exploration of event and entity links in implicit networks. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 273–277.

[35] Ke Sun, Lei Wang, Bo Xu, Wenhong Zhao, Shyh Wei Teng, and Feng Xia. 2020. Network representation learning: From traditional feature learning to deep learning. *IEEE Access* 8 (2020), 205600–205617.

[36] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. PTE: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1165–1174.

[37] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1067–1077.

[38] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the International Conference on Machine Learning*. 2071–2080.

[39] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, April 30 - May 3, 2018.*

[40] Pengyang Wang, Kunpeng Liu, Lu Jiang, Xiaolin Li, and Yanjie Fu. 2020. Incremental mobile user profiling: Reinforcement learning with spatial knowledge graph for modeling event streams. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 853–861.

[41] Wei Wang, Xiaomei Bai, Feng Xia, Teshome Megersa Bekele, Xiaoyan Su, and Amr Tolba. 2017. From triadic closure to conference closure: The role of academic conferences in promoting scientific collaborations. *Scientometrics* 113, 1 (2017), 177–193.

[42] Wei Wang, Jiaying Liu, Tao Tang, Suppawong Tuarob, Feng Xia, Zhiguo Gong, and Irwin King. 2020. Attributed collaboration network embedding for academic relationship mining. *ACM Transactions on the Web (TWEB)* 15, 1 (2020), 1–20.

[43] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *Proceedings of the World Wide Web Conference*. 2022–2032.

[44] Yifan Wang, Suyao Tang, Yuntong Lei, Weiping Song, Sheng Wang, and Ming Zhang. 2020. Disenhan: Disentangled heterogeneous graph attention network for recommendation. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*. 1605–1614.

[45] Jonathan D. Wren, Raffi Bekeredjian, Jelena A. Stewart, Ralph V. Shohet, and Harold R. Garner. 2004. Knowledge discovery by automated identification and ranking of implicit relationships. *Bioinformatics* 20, 3 (2004), 389–398.

[46] Feng Xia, Jiaying Liu, Hansong Nie, Yonghao Fu, Liangtian Wan, and Xiangjie Kong. 2019. Random walks: A review of algorithms and applications. *IEEE Transactions on Emerging Topics in Computational Intelligence* 4, 2 (2019), 95–107.

[47] Feng Xia, Ke Sun, Shuo Yu, Abdul Aziz, Liangtian Wan, Shirui Pan, and Huan Liu. 2021. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence* 2, 2 (2021), 109–127.

[48] Feng Xia, Shuo Yu, Chengfei Liu, Jianxin Li, and Ivan Lee. 2022. CHIEF: Clustering with higher-order motifs in big networks. *IEEE Transactions on Network Science and Engineering* 9, 3 (2022), 990–1005. DOI: https://doi.org/10.1109/TNSE.2021.3108974

[49] Jin Xu, Shuo Yu, Ke Sun, Jing Ren, Ivan Lee, Shirui Pan, and Feng Xia. 2020. Multivariate relations aggregation learning in social networks. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*. 77–86.

[50] Fajie Yuan, Xiangnan He, Alexandros Karatzoglou, and Liguang Zhang. 2020. Parameter-efficient transfer from sequential behaviors for user modeling and recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1469–1478.

[51] Herman Yuliansyah, Zulaiha Ali Othman, and Azuraliza Abu Bakar. 2020. Taxonomy of link prediction for social network analysis: A review. *IEEE Access* 8 (2020), 183470–183487.

[52] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. 2019. Graph transformer networks. *Advances in Neural Information Processing Systems* 32 (2019), 11983–11993.

[53] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V. Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 793–803.

[54] Bin Zhao, Prithviraj Sen, and Lise Getoor. 2006. Entity and relationship labeling in affiliation networks. In *Proceedings of the 23nd ICML Workshop on Statistical Network Analysis: Models, Issues, and New Directions*.

Received 21 February 2021; revised 19 January 2022; accepted 10 September 2022