Information Fusion 111 (2024) 102525

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full length article

Interest-driven community detection on attributed heterogeneous information networks

Mengyue Liu [a,*], Jun Liu [a], Yixiang Dong [a], Rui Mao [b], Erik Cambria [b]

[a] School of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China
[b] School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798, Singapore

ARTICLE INFO

Keywords:
Community detection
Attributed heterogeneous information networks
Network representation learning

ABSTRACT

Community structures within attributed heterogeneous information networks (AHINs) serve as valuable tools for comprehending the functional properties inherent in the real-world systems they mirror. The diverse semantics embedded in AHINs play a pivotal role in shaping distinct community formations. Many existing methods detect communities in AHINs based on the same-type nodes without specifying semantic meanings, resulting in unclear and potentially ambiguous outcomes. Additionally, by adopting a simple strategy that focuses solely on either graph structures or node attributes, they fail to sufficiently integrate heterogeneous information and maintain semantic consistency throughout the entire detection process. In this paper, we propose IDCD, a user-interest-driven community detection framework on AHINs, which groups heterogeneous nodes following specified user guidance for semantics-clear results. Firstly, we map heterogeneous nodes into one unified representation space by neatly integrating user-interest-related heterogeneous information. Then we assign pseudo community labels to nodes and refine them in a self-training way. These two processes are iteratively optimized and enhanced mutually in a unified framework. Extensive experiments demonstrate the superiority of IDCD.

1. Introduction

A community or modular structure is generally defined as a group of nodes that have dense connections to each other and probably share common properties [1]. Such structure widely exists in real-world networks and accounts for their functionality [2]. Community detection, aiming at partitioning a network into multiple subgraphs [2–5], provides direct insight into how the network is organized. It has many interesting applications in various practical domains [6–8].

Heterogeneous Information Networks (HINs) (in which there are multiple type nodes/edges) are effective in modeling real-world networks because they are much more expressive than homogeneous information networks (in which all nodes/edges are of one single type) in capturing complex graph-structured knowledge [9–11]. HINs with node attributes are called *a*ttributed HINs or AHINs for short. The multiple types of nodes are connected via different relations, referred to as meta-paths, implying diverse semantic meanings[1] [12]. The wealth of information in AHINs provides great potential for community detection [11,13] while complicating the community detection task.

Firstly, multi-typed nodes with similar characteristics are often densely connected, sharing the same topic [14]. Secondly, various semantics implied by meta-paths can be irrelevant or even opposite for a community detection task, leading to completely different detecting processes of communities. As shown in Fig. 1, we use a toy-attributed heterogeneous bibliographic network as an example to demonstrate community detection processes in an AHIN. The toy bibliographic network contains three types of nodes: *author* (A), *paper* (P), and *venue* (V), and two types of edges: *write* and *accept*. Two abstracted meta-paths, A-P-A and A-P-V-P-A, indicate semantic meanings of the co-author relationship and research area, respectively.

Detecting communities via each of them will generate different results: three communities in orange (Group A, B, C) via A-P-A, or two communities in green (Group 1, 2) via A-P-V-P-A. In each result, heterogeneous nodes cohere with the same semantic meaning, forming a complete semantic unit. For example, Group 1 (2) includes a set of research area-related authors, papers, and venues. Obviously, via each meta-path, the detected communities are reasonable, but via a combination of them, the result will be meaningless.

* Corresponding author.
E-mail addresses: liumengyue@stu.xjtu.edu.cn (M. Liu), liukeen@xjtu.edu.cn (J. Liu), dyx1102@stu.xjtu.edu.cn (Y. Dong), rui.mao@ntu.edu.sg (R. Mao), cambria@ntu.edu.sg (E. Cambria).

[1] We employ the term "semantic meaning" for differentiation from "pragmatic meanings" in our analytical framework, where "pragmatic meanings" are not the focus of this work.

https://doi.org/10.1016/j.inffus.2024.102525
Received 21 January 2024; Received in revised form 26 April 2024; Accepted 7 June 2024
Available online 10 June 2024

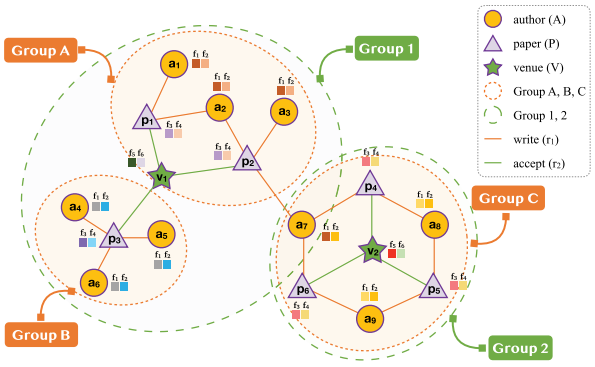


Fig. 1. A toy heterogeneous bibliographic network, and the detected communities are enclosed by several dash circles in two different colors (green and orange). The results of community detection vary with the selection of meta-path. Following A-P-A path, three orange communities *Group A, B, C* are detected, indicating scholar cooperation. Following A-P-V-P-A path, two green communities *Group 1, 2* are detected, indicating research areas.

The unique characteristics inherent in AHINs necessitate the development of a novel paradigm for community detection. Within this framework, users play a pivotal role by offering guidance to tailor the identification of communities to their specific areas of interest. Following the new paradigm, community detection of AHINs presents richer and more meaningful community information. However, it faces two main challenges, *i.e.*, how to exploit heterogeneous information related to user interest; how to capture community information.

Interest-related heterogeneous information utilization. Heterogeneous information in AHINs consists of heterogeneous network structures, multiple semantic meanings, and node attributes, which are important sources when measuring the similarity between nodes.

Firstly, it is difficult for users to capture their interest-related heterogeneous information or provide clear guidance for a certain community detection task. Most existing methods detect communities on AHINs without any user supervision, obtaining inaccurate communities of which the physical formation mechanism is inexplicable [15,16]. A few community detection methods for HINs (AHINs) adopt label constraints such as anchor nodes [17] or a "must-link" set [18,19] to determine a weighted combination of meta-paths for their desired community detection task. However, label constraints can be invalid and unreliable when they appear in more than one community detection process. Besides, it is difficult for users to provide explicit label constraints and exhaustive meta-paths.

Secondly, the challenge lies in the simultaneous integration and utilization of heterogeneous information due to their disparate data structures. Certain earlier methods [16,17] characterized by a link-based approach, tend to overlook the incorporation of node attributes, which are instrumental in elucidating the formation of communities; Some recent methods [15,18,20] intend to leverage all information to make themselves more general whereas have the same information missing problem [21] as homogeneous methods: multiple homogeneous sub-networks projected from original AHINs are fed to them. Then, the information of other types of nodes is not utilized essentially.

Community information capture. Traditional community detection methods for HINs design various community division metrics for distinct topology characteristics, which cannot be generalized to all kinds of real-world HINs [22,23]. In recent years, heterogeneous network embedding techniques have become a potent tool for the community detection task because they are not limited to specific heterogeneous network structures. They leverage heterogeneous information to embed nodes within AHINs, subsequently identifying community structures through the application of various clustering algorithms, such as k-means. The subsequent community detection process can be independent of the embedding process [24–26] or unified with it [15,16]. Nonetheless, an independent optimization approach is susceptible to error propagation, given the typically independent assumptions between network embedding techniques and clustering techniques. On the other hand, methods employing a unified optimization approach tend to neglect aligning the semantics of the network embedding process with the community detection process. This oversight allows for the introduction of noisy semantic meanings, disrupting the semantic consistency throughout the entire community detection task.

In this work, we propose IDCD (Interest-Driven Community Detection) to address the above challenges and detect meaningful communities for heterogeneous nodes in AHINs. To address the first challenge, IDCD follows a simple yet efficient user guidance, *i.e.*, a meta-path related to user interest, to guarantee a unique and accurate community detection process. A heterogeneous information encoder is proposed to leverage user interest-related heterogeneous information and map heterogeneous nodes into one lower-dimensional representation space. Specifically, meta-path-based random walks capture the network structures of heterogeneous nodes, and an attributes aggregator utilizes similarities measured by attributes of heterogeneous nodes to select more informative nodes for the community detection task. To address the second challenge, we assign pseudo community labels to the obtained embeddings via a Gaussian mixture model and adopt a variant self-training clustering objective to make the assumptions between the network embedding process and community detection process coherent. During this process, heterogeneous node representations and community assignments are jointly optimized and mutually improved in a unified framework.

The main contributions of our work are as follows:

- We elaborate and refine the community detection task in AHINs by introducing a user-interest-driven approach, allowing for the customization of community detection according to specific semantic meanings within the network. The detected communities are not only heterogeneous in composition but also aligned with user-specified semantics, leading to more meaningful and targeted communities.
- We propose a simple but efficient network embedding-based community detection model, termed IDCD. It contains an efficient two-step sampling strategy that first leverages meta-path-based random walks to capture network structure, succeeded by an attribute-based selection to identify informative nodes. By favoring Gaussian Mixture Models over the common reliance on Student's t-distribution, IDCD provides novel perspectives for selecting distributions in community detection.
- We extract four real-world datasets for applying to the new paradigm of community detection on AHINs. Extensive experiments on these datasets demonstrate that IDCD consistently and significantly outperforms the state-of-the-art baselines (3.68% on ACC, 5.91% on NMI, 4.06% on ARI).

2. Related work

Traditional community detection methods. Early community detection methods for HINs adopt graph partition algorithms to divide HINs [22,23]. The most used graph partition algorithm is modularity maximization, and the definitions of modularity vary from HIN structures. For example, Barber [22] developed a modularity matrix for bipartite networks, and Zhang and Chen [23] defined a heterogeneous modularity function for general HINs. However, the modularity maximization problem is NP-hard and requires an unreasonable amount of time [27]. Spectral clustering is another prevalent graph partition algorithm. SClump [16] adopts spectral clustering and performs eigen-decomposition on heterogeneous similarity matrix, then applies

standard clustering techniques, such as k-means to partition nodes in HINs. However, spectral clustering can be unreliable when the network is very sparse [2]. Another class of community detection methods adopts statistical inference such as fitting a generative network model on the data. For example, PathSelClus [17] proposes a probabilistic approach and integrates meta-path selection with user guidance, aiming to detect communities that are semantically consistent with the user demands. Significantly, however, all of the algorithms presented so far are link-based clustering algorithms that cannot take advantage of node attributes. Meanwhile, they can only cluster same-type nodes in HINs and are uneasy to generalize to cluster heterogeneous nodes.

Network embedding-based methods. In recent years, network embedding as an important technique to learn low-dimensional representations for nodes in networks presents us with a potent tool for community detection. Network embedding can effectively capture highly nonlinear network structures and preserve the global and local structure, providing deep representations for networks. The general community detection process is: firstly, researchers adopt some HIN embedding methods to obtain node latent representations, and then apply some traditional clustering algorithms to the obtained embeddings to learn their community assignments. These two steps can be optimized in a separate way or a unified way.

For a separate optimization strategy, many outstanding HIN embedding methods can be replaced freely in the first step and can be divided into two categories. (1) *Random walk-based methods*: Metapath2vec [28] defines meta-path based random walks to capture network structures and leverages Skip-gram [29] to learn node embeddings. HIN2Vec [30] carries out multiple prediction training tasks jointly based on a target set of relationships to learn latent vectors of nodes and meta-paths. However, they are limited in capturing node attributes. (2) *Heterogeneous Graph Neural Networks (HGNNs)-based methods*: They obtain more powerful embeddings via employing deep neural networks to the aggregate information of neighboring nodes [18,24–26,31]. For example, HAN [24] and HGT [25] are extensions of GAT [24] and they design node-type and semantic/edge-type attention neural networks to embed nodes in HINs or dynamic HINs. HeCo [26] proposes a novel co-contrastive learning mechanism to embed nodes. HGNN-based embedding methods have shown superior ability to embed nodes, but they may achieve sub-optimal results on community detection, since their representation learning process is independent of the community detection task.

For a unified optimization strategy, a few methods are designed to make the network embedding process aligned with the community detection task. O2MAC [15] employs one informative graph view to reconstruct multiple graph views and integrates the reconstruction process with a self-training clustering process to cluster nodes and learn community-aware node embeddings. VACA-HINE [20] employs a variational approach to preserve pairwise proximity in a cluster-aware manner and utilizes a contrastive approach to preserve high-order HIN semantics. However, O2MAC and VACA-HINE do not provide any user guidance for the community detection task and use multiple meta-paths, which can be opposite to each other or irrelevant to a clustering task, preventing meaningful community detection results. SCHAIN-IRAM [18] notices this problem and uses a "must-link" set and a "cannot-link" set to supervise the clustering process. However, in reality, such supervision signal is possible to exist in different clustering results, and the semantic information in obtained communities can still be ambiguous.

3. Preliminaries and problem definition

3.1. Preliminaries

Definition 1 (*Attributed Heterogeneous Information Network (AHIN) [32]*)**.** An AHIN is defined as $G = (V, E, \mathcal{F})$ with a node-type mapping function $\phi : V \to \mathcal{O}$, an edge-type mapping function $\varphi : E \to \mathcal{R}$, and $|\mathcal{O}| + |\mathcal{R}| > 2$. $\mathcal{F} = \bigcup_{O_j \in \mathcal{O}} F_{O_j}$ and $F_{O_j} = \{f_1, f_2, \dots\}$ is an attribute set corresponding to nodes of type $O_j \in \mathcal{O}$.

Table 1
Notation description.

Notation	Description
G	Attributed heterogeneous information network
V, E, $\mathcal{F}$	Node/edge/attribute set of G
$\mathcal{A}$, $\mathcal{R}$	Node/edge type set
$\mathbf{F}$	Heterogeneous attribute matrix
$\mathcal{P}$	Meta-path
T	Target type
$\mathcal{W}$	Semantic-related node corpus
$\mathbf{z}_i$	Latent representation of node $v_i \in V$
$\mathcal{C}$	Cover (set of detected communities)
K	The numbers of communities
C_k	A set of nodes belonging to kth community
$\mathbf{q}_i$	Community assignment of node v_i
q_{ik}	Strength of association between node i and community k
π_k	Mixing coefficient of community k

Example 1. In Fig. 1, an AHIN $G = (V, E, \mathcal{F})$ contains three types of nodes $\mathcal{O}$ = {A, P, V} (A for *author*, P for *paper*, V for *venue*), and two types of edges $\mathcal{R} = \{r_1, r_2\}$ (r_1 for *write*, r_2 for *accept*), and node attributes set $\mathcal{F} = \{F_A, F_P, F_V\}$, and $F_A = \{f_1, f_2\}$, $F_P = \{f_3, f_4\}$, $F_V = \{f_5, f_6\}$ denote attribute set of *author*, *paper* and *venue*, respectively.

Definition 2 (*Meta-path [12]*)**.** A meta-path $\mathcal{P} : O_1 \xrightarrow{R_1} O_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} O_{l+1}$ defines a composite relation $R = R_1 \circ R_2 \circ \dots \circ R_l$ between O_1 and O_{l+1}, where $\circ$ denotes the composition operator on edges. l is the length of $\mathcal{P}$. Subscripts $(1, 2, \dots, l)$ represent their positions in $\mathcal{P}$.

Example 2. As shown in Fig. 1, meta-paths $\mathrm{A} \xrightarrow{r_1} \mathrm{P} \xrightarrow{r_1^{-1}} \mathrm{A}$ and $\mathrm{A} \xrightarrow{r_1} \mathrm{P} \xrightarrow{r_2^{-1}} \mathrm{V} \xrightarrow{r_2} \mathrm{P} \xrightarrow{r_1^{-1}} \mathrm{A}$ denote the semantic meanings of co-author relationship and research area, respectively.

Definition 3 (*Overlapping Community [33]*)**.** A group of overlapping communities can be defined as a *cover* $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$ [34]. In this *cover*, node v_i is associated with a community by a belonging factor (also referred to as community assignment) $\mathbf{q}_i = [q_{i1}, \dots, q_{ik}, \dots, q_{iK}]$, where $0 \le q_{ik} \le 1$ and $\sum_{k=1}^{K} q_{ik} = 1$ [35].

Example 3. In Fig. 1, *Group* 1 and *Group* 2 are two overlapping communities. $\mathbf{q}_7$ is the community assignment of node a_7, and $\mathbf{q}_7 = [0.49, 0.51]$ denotes the probabilities of a_7 belonging to *Group* 1 and *Group* 2 are 0.49 and 0.51, respectively.

More details about the notations used in this paper are listed in Table 1.

3.2. Problem definition

As we discussed before, detecting communities in AHINs requires user guidance to avoid the ambiguous semantic information hiding in multiple meta-paths. Moreover, the heterogeneous nodes are clustered because of their inherent semantic consistency.

The problem of community detection in an AHIN is formulated as follows:

Problem 1 (*Interest-Driven Community Detection on AHINs*)**.** Given an AHIN $G = \{V, E, \mathcal{F}\}$, an interest-driven community detection needs user guidance on specifying: (1) the number of communities K, and (2) a meta-path $\mathcal{P}$ implying one semantic meaning which user is interested in.

The output includes two parts:

(1) Community-aware heterogeneous node embeddings.
(2) A *cover* $\mathcal{C} = \{C_1, C_2, \dots, C_K\}$ consists of detected communities on G, and the semantic information behind the generation of $\mathcal{C}$ is consistent with the meta-path $\mathcal{P}$.

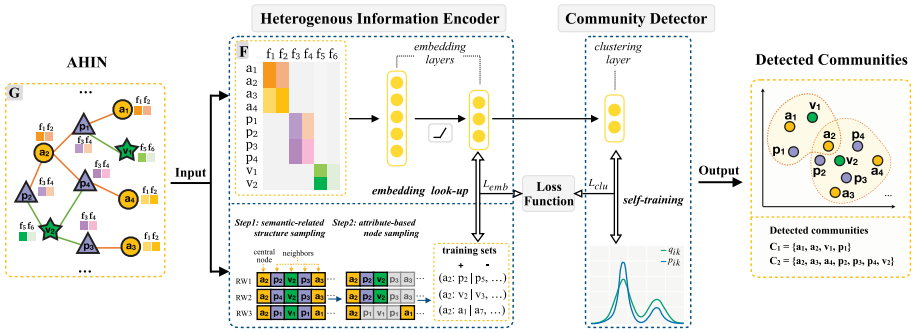


Fig. 2. The framework of IDCD.

4. The proposed methodology

The framework of the proposed model IDCD is shown in Fig. 2. IDCD consists of two main components: heterogeneous information encoder and community detector. (1) With a semantic-specific meta-path and node attributes, the most informative nodes are selected to guide the **heterogeneous information encoder** to embed heterogeneous nodes into one representation space. (2) **Community detector** assigns pseudo community labels to the learned node representations and refines them with an auxiliary distribution in a self-training way. The embedding process and community detection process are jointly optimized and mutually enhanced in a unified way.

4.1. Heterogeneous information encoder

Given an AHIN $G = \{V, E, \mathcal{F}\}$, and specified meta-path $\mathcal{P}$, we adopt a two-layer fully connected neural network g to encode heterogeneous nodes into one representation space. As meta-path $\mathcal{P}$ may not involve all types of nodes, we denote the set of involved types as $\mathcal{O}_{\mathcal{P}} = \{\mathrm{O}_1, \ldots, \mathrm{O}_m\}$, $m \le |\mathcal{O}|$. We construct a heterogeneous attribute matrix $\mathbf{F}$ as follow:

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\mathrm{O}_1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{F}_{\mathrm{O}_2} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{F}_{\mathrm{O}_m} \end{bmatrix}$$

where $\mathbf{F}_{\mathrm{O}_j} \in \mathbb{R}^{|V_{\mathrm{O}_j}| \times |F_{\mathrm{O}_j}|}$ denotes attribute matrix of O_j-type nodes ($j = 1, \ldots, m$), and V_{O_j} is the set of nodes belonging to type O_j. The heterogeneous attribute matrix $\mathbf{F}$ is fed to embed heterogeneous nodes into one d-dimensional embedding space:

$$\begin{cases} \mathbf{h}_i = \mathrm{relu}\left(\mathbf{f}_i \mathbf{W}_1 + \mathbf{b}_1\right) \\ \mathbf{z}_i = \mathbf{h}_i \mathbf{W}_2 + \mathbf{b}_2 \end{cases} \tag{1}$$

where $\mathbf{f}_i$ is the i-row of $\mathbf{F}$, and $\mathbf{W}_1, \mathbf{W}_2$ is the weight matrix of layer 1 and 2 of g, respectively.

We apply a graph-based loss function, a variant of negative sampling [29], to the output representations. This loss function encourages nearby nodes to have similar representations, while enforcing that the representations of disparate nodes are highly distinct:

$$\mathcal{L}_{emb} = -\log \sigma(\mathbf{z}_i \cdot \mathbf{z}_j^t) - \sum_{m=1}^{\mathrm{M}} \mathbb{E}_{v_m^t \sim P_{\mathrm{neg}}(v^t)} \left(\log \sigma(-\mathbf{z}_i \cdot \mathbf{z}_m^t)\right), \tag{2}$$

where $\sigma(x)$ is the sigmoid function, P_{neg} is a negative sampling distribution, and M defines the number of negative samples. The superscript t of $\mathbf{z}_j^t$ indicates node v_j belonging to type t. $\mathbf{z}_j^t$ is the representation of a t-type positive samples of $\mathbf{z}_i$, and $\mathbf{z}_m^t$ is the representation of a t-type negative samples. Eq. (2) indicates that for one node v_i, its negative sample and positive sample should belong to the same type, so as to make the contrasting pairs potent. For instance, in Fig. 2, a_3's A-type neighborhood is $N_{\mathrm{A}}(a_3) = \{a_2, a_4\}$, P-type neighborhood is $N_{\mathrm{P}}(a_3) = \{p_3\}$, and the corresponding negative sample is a_1 and p_1, respectively.

Generally, the positive and negative samples can be selected from random walks [36,37]. However, such a process has two limitations. First, the whole process ignores integrating node attributes. Secondly, they pay less attention to sampling informative positive and negative samples [38]. On the one hand, positive samples contribute differently to the training. For example, an expert in the neural network field is the last coauthor of a paper in the software engineering field, thus this paper contributes little to the embedding process of the expert. On the other hand, negative samples should not exist in a positive sample set, while negative sampling has no limitation on this.

To address above mentioned problem, we design a two-step sampling strategy. First, we sample the semantic-related network structures based on the specified meta-path. Then we further sample informative nodes based on node attributes.

Step1. Semantic-related Structure Sampling: We adopt a meta-path-based random walker [28] to traverse the AHIN G, based on a specified meta-path $\mathcal{P}$. The set of traversed nodes is denoted as $V_s = \{v_1, \ldots, v_n\}$, $n \le |V|$, referred to as *node corpus*. For instance, given a meta-path $\mathcal{P}' : \mathrm{A}_1 \xrightarrow{\mathrm{R}_1} \mathrm{A}_2 \xrightarrow{\mathrm{R}_2} \mathrm{A}_3$, we start from a node v_i ($\phi(v_i) = \mathrm{A}_1$), and the next step we will arrive at node v_j, which is satisfied with: (i) $\varphi(v_i, v_j) = \mathrm{R}_1$ and (ii) $\phi(v_j) = \mathrm{A}_2$. If H nodes are satisfied with the above two conditions, we will randomly select one with a probability $1/\mathrm{H}$. By simulating a number of random walks of a fixed length starting from every node belonging to type A_1, G is extracted as lots of random walks. For each walk, a window w is given to demarcate the region of positive samples. For a central node v_i, heterogeneous neighbors within the window w (w-hop) are regarded as its positive samples, *i.e.*, $N(v_i) = \{v_j | d(v_i, v_j) \le w\}$, and $d(v_i, v_j)$ denotes the geometric distance between v_i and v_j. The negative samples are drawn according to their unigram distribution raised to the 3/4 power.

Step2. Node Attribute-based Sampling: In order to sample more informative positive and negative pairs, we further select heterogeneous nodes based on their attributes. For a central node v_i, we collect all its t-type neighbors $N_t(v_i)$ and select informative ones. The sampling probability for each v_j^t is defined as:

$$p(v_j^t | v_i) = \begin{cases} \dfrac{\exp(sim(\mathbf{f}_i, \mathbf{f}_j)/\tau)}{\sum_{v_{j'} \in N_t(v_i)} \exp(sim(\mathbf{f}_i, \mathbf{f}_{j'})/\tau)} & \phi(v_i) = t \\ \dfrac{\exp(sim(\mathbf{f}_i^{\mathrm{aggr}}, \mathbf{f}_j)/\tau)}{\sum_{v_{j'} \in N_t(v_i)} \exp(sim(\mathbf{f}_i^{\mathrm{aggr}}, \mathbf{f}_{j'})/\tau)} & \phi(v_i) \ne t \end{cases}, \tag{3}$$

where $sim(\cdot, \cdot)$ is based on cosine similarity, and τ is a hyper-parameter tuned to the distinction of positives. When the positive sample belongs to the same type as the central node, we can directly calculate the similarity based on their attributes $\mathbf{f}_i, \mathbf{f}_j$. When they belong to different types, we use an aggregator function to assign a new attribute $\mathbf{f}_i^{\mathrm{aggr}}$ for the central node v_i with all its t-type neighbors:

$$\mathbf{f}_i^{\mathrm{aggr}} \leftarrow \mathrm{MEAN}\left\{\mathbf{f}_j, \forall v_j \in N_t(v_i)\right\}. \tag{4}$$

In practice, we set a sampling ratio η for positive node selection, and the value of η varies from different datasets, empirically set to 0.8.

Negative samples are filtered from the obtained negative pair set by eliminating those that exist in v_i's neighborhood. The sampling probability formula for negative samples resembles Eq. (3). However, the distinction lies in the fact that the softmax function operates on the reciprocal of the similarity between a central node and a negative sample. The informative nodes sampling strategy is off-line without much extra consumption.

4.2. Community detector

Aim to align node representation learning to the community detection task, the community detector attaches some community information to the obtained embeddings for guidance, learning community-aware node embeddings. The whole process consists of two parts: community assignment and self-training.

Community assignment. Mixture models are widely used in clustering. They assume that the observations to be clustered are drawn from one of several components [39]. Under this assumption, the problem of community detection undergoes a transformation into an inference problem involving the parameters of components, specifically focusing on identifying the memberships of nodes within these components. As community detection is an unsupervised task, the primary role of community assignment is to assign pseudo-community labels for nodes with the characteristics behind their embeddings.

We adopt Gaussian Mixture Model (GMM) to depict the distributions of node embeddings in the latent space. Suppose that there are K communities in an AHIN $G = \{V, E, \mathcal{F}\}$. With the node representation $\mathbf{z}_i \in \mathbb{R}^d$ learned from heterogeneous node encoder, the marginal distribution of $\mathbf{z}_i$ is formulated as:

$$p(\mathbf{z}_i) = \sum_{k=1}^{\mathrm{K}} \pi_k \mathcal{N}(\mathbf{z}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{5}$$

where $\pi_1, \ldots, \pi_{\mathrm{K}}$ are the mixing coefficients corresponding to K components, satisfying $\sum_{k=1}^{\mathrm{K}} \pi_k = 1$ ($\pi_k \in [0, 1]$). The density function of a multivariate Gaussian distribution $\mathcal{N}(\mathbf{z}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is given by:

$$\begin{aligned}\mathcal{N}(\mathbf{z}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = & \frac{1}{(2\pi)^{d/2}} \frac{1}{|\boldsymbol{\Sigma}_k|^{1/2}} \\ & \cdot \exp\{-\frac{1}{2}(\mathbf{z}_i - \boldsymbol{\mu}_k)^{\top} \boldsymbol{\Sigma}_k^{-1} (\mathbf{z}_i - \boldsymbol{\mu}_k)\},\end{aligned} \tag{6}$$

where $\boldsymbol{\mu}_k \in \mathbb{R}^d$ and $\boldsymbol{\Sigma}_k \in \mathbb{R}^{d \times d}$ is the mean vector and covariance matrix of the kth Gaussian component C_k. $|\boldsymbol{\Sigma}_k|$ denotes the determinant of $\boldsymbol{\Sigma}_k$.

From Bayes rule, the probability that a data point $\mathbf{z}_i$ comes from kth Gaussian model is calculated as:

$$\begin{aligned} q_{ik} &= \frac{p(C_k = 1) p(\mathbf{z}_i|C_k = 1)}{\sum_{j=1}^{\mathrm{K}} p(C_j = 1) p(\mathbf{z}_i|C_j = 1)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{z}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{\mathrm{K}} \pi_j \mathcal{N}(\mathbf{z}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \\ &= \frac{\pi_k \exp(\|\mathbf{z}_i - \boldsymbol{\mu}_k\|^2 / 2\boldsymbol{\Sigma}_k^2)}{\sum_{j=1}^{\mathrm{K}} \pi_j \exp(\|\mathbf{z}_i - \boldsymbol{\mu}_j\|^2 / 2\boldsymbol{\Sigma}_j^2)}. \end{aligned} \tag{7}$$

q_{ik} measures the similarity between embedded point $\mathbf{z}_i$ and k-centroid $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, and is interpreted as the probability of assigning node v_i to cluster C_k. Therefore, the soft community assignment of v_i is defined as $\mathbf{q}_i = [q_{i1}, \ldots, q_{i\mathrm{K}}]$.

Self-training. According to the soft assignment vector q_i, we obtain a pseudo community label of node v_i. Aim to enhance the "credibility" of the pseudo label, inspired by [40,41], we introduce an auxiliary target distribution to refine the process of community assignment. The auxiliary target distribution is defined as:

$$p_{ik} = \frac{q_{ik}^2 / \sum_i q_{ik}}{\sum_{k'} (q_{ik'}^2 / \sum_i q_{ik'})}. \tag{8}$$

Here, we first raise $\mathbf{q}_i$ to the second power, strengthening the confidence of the higher community assignments. Then we normalize it with cluster frequencies $\sum_i q_{ik}$.

Then, we approximate auxiliary distribution P with the obtained Gaussian mixture distribution Q to refine the community assignments. The difference between P and Q is measured with Kullback–Leibler(KL) divergence, and the community detection loss is defined as:

$$\mathcal{L}_{clu} = \mathrm{KL}(P \parallel Q) = \sum_i \sum_k p_{ik} \log \frac{p_{ik}}{q_{ik}}. \tag{9}$$

Note that KL divergence is asymmetric and the positions of P and Q in Eq. (9) are important.

By minimizing $\mathcal{L}_{clu}$, nodes with higher "credibility" will supervise the rest nodes gathering towards the right communities.

4.3. Jointly optimization

We jointly optimize these two components and define an overall loss function of IDCD as:

$$\mathcal{L} = \mathcal{L}_{emb} + \alpha \mathcal{L}_{clu}. \tag{10}$$

where $\alpha > 0$ is a coefficient that balances the heterogeneous node learning and clustering optimization. $\mathcal{L}_{clu}$ not only enhances the cohesiveness of nodes within communities but also introduces community-level knowledge into the heterogeneous node embedding process. However, before we learned effective node representations, approximating the target distribution P with posterior distribution Q would mislead the training direction. To tackle the problem, we only train the heterogeneous node embedding loss at the initial stage, and then gradually increase the weight of clustering loss. The pseudocode for IDCD, is given in Algorithm 1.

Algorithm 1: The proposed IDCD

Input: an AHIN: $G = \{V, E, \mathcal{F}\}$, semantic-related meta-path $\mathcal{P}$, window size w, walk length *len*, number of walks wk, number of communities K, coefficient α, update interval I, sampling probability of positive samples p_{pos}, sampling probability of negative samples p_{neg}, the number of pre-train epoch L.
Output: node embeddings $\mathbf{Z}$, community detection result C.

Initialize an node corpus $\mathcal{W} = \emptyset$.
for *iter = 1 : wk* **do**
  for ***each*** $v_i \in V_s$ **do**
    W_{v_i} = metapathBasedRandomWalker(G, T, $\mathcal{P}$, *len*),
    Append W_{v_i} to $\mathcal{W}$.
  end
end
Select informative positive pair set $S_p \leftarrow p_{\mathrm{pos}}(\mathcal{W})$.
Select informative negative pair set $S_n \leftarrow p_{\mathrm{neg}}(\mathcal{W})$.
for *epoch = 0 : L* **do**
  Pre-train g and updated its parameters by minimizing $\mathcal{L}_{emb}$
end
repeat
  $\mathbf{Z} = g(\mathbf{F}, S_p, S_n)$.
  if $epoch\%I == 0$ **then**
    Update $\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}$ of clustering layer with GMM($\mathbf{Z}$).
  end
  Calculate soft assignment Q by Eq.(7);
  Calculate target distribution P by Eq.(8);
  Calculate $\mathcal{L}$ according to Eq.(2), Eq.(9);
  Update parameters of IDCD by minimizing $\mathcal{L}$.
  Update community assignments in C according to Q.
until *Early Stopping or Convergence*;
return node embeddings $\mathbf{Z}$, community detection result C.

4.4. Complexity analysis

IDCD first generates semantically related random walks using one or more assigned meta-paths. This step involves a variable S representing

Table 2
Dataset statistics.

Dataset	Node types	# Nodes	Relations (A_s-A_t)	# Relations	Avg.Degree of A_s	Avg.Degree of A_t	Meta-path used	# clusters	# attribute dimension	% of largest cluster	% of smallest cluster
DBLP	*Author (A)	14,475	AP	41,794	2.89	2.91	APA	4	61	33.54%	15.40%
	Paper (P)	14,376	PV	14,376	1.00	718.80	APTPA				
	Venue (V)	20	PT	114,624	7.97	12.85	*APVPA				
	Term (T)	8,920									
MovieLens	*User (U)	1,631	UM	128,926	79.04	23.33	*UMU	4	14	35.87%	16.72%
	Movie (M)	5,526									
Yelp	*User (U)	15,883	UR	198,397	12.49	15.18	UU	10	1603	44.00%	0.65%
	Restaurant (R)	13,064	RC	14,284	1.09	1428.40	URU				
	City (C)	10	UU	158,590	9.98	9.98	*URCRU				
	Description (D)	11	UD	76,875	4.84	6988.63					
	Style (S)	511	RS	40,009	3.06	78.30					
AMiner	*Author (A)	41,274	AP	106,026	2.57	5.81	APA	4	512	35.37%	15.25%
	Paper (P)	18,243	PV	18,235	1.00	911.75	*APVPA				
	Venue (V)	12									

the number of meta-paths, the number of walks wk, and the walk length L. The time complexity for traversing a graph with $|V|$ nodes using meta-path-based random walks is $\mathcal{O}_{rw}(|V| * S * L * wk)$. However, this step is executed only once before training begins and hence does not contribute to the per-epoch time complexity. Then, IDCD designs a heterogeneous information encoder to warm up the model and transforms D-dimensional input attributes into d-dimensional node embeddings with time complexity $\mathcal{O}_{enc}(|V| * D * d)$, which is part of the per-epoch complexity. The embedding loss (a negative sampling-based loss function), which includes M negatives, is computed with time complexity of $\mathcal{O}_{emb}(|V| * (\mathrm{M}+1) * d)$. The community detector assesses the discrepancy between each of the $|V|$ node embeddings and the k central embeddings, with a time complexity of $\mathcal{O}_{clu} = \mathcal{O}_{clu}(|V| * k * d)$. Combining these complexities, the per-epoch time complexity of IDCD is derived as $\mathcal{O}_{\mathrm{IDCD}} = \mathcal{O}_{enc}(|V| * D * d) + \mathcal{O}_{emb}(|V| * (\mathrm{M}+1) * d) + \mathcal{O}_{clu}(|V| * k * d)$, which simplifies to $\mathcal{O}(|V| * (D + \mathrm{M} + 1 + k) * d)$.

5. Experiments

We evaluate IDCD with the following research questions:

- **RQ1**: How effective is our IDCD compared to the state-of-the-art baselines?
- **RQ2**: How different soft assignment distribution Q affects the performance?
- **RQ3 (Sensitivity Analysis)**: How the runtime parameters of IDCD affect the overall performance?
- **RQ4 (Case Study)**: How different meta-paths influence the process of community detection in AHINs?

5.1. Experiments setup

5.1.1. Datasets

As the existing datasets do not provide labels for heterogeneous nodes in AHINs. We construct four AHINs from four widely used networks, including AMiner,[2] DBLP,[3] MovieLens[4] and Yelp,[5] ranging from academic networks to social networks. Table 2 shows their statistics.

- **AMiner** and **DBLP** are academic networks which contain bibliographic entities. AMiner contains three different types of nodes: venue (V), *author* (A), and *paper* (P), and DBLP contains one more type: *Term* (T). We collect nodes from four research areas including *Database*, *Data Mining*, *Computer Vision* and *Machine Learning*,[6] and leverage the statistical of user publications and the paper abstract to generate node attributes.
- **MovieLens** is a social network collecting data of users interacting on the MovieLens[7] online recommender system. We collect *users* (U), and *movies* (M) from 16 sub-classes which are categorized into four movie genres: *Drama*, *Comedy*, *Horror*, and *Fantasy*. The attributes consist of user ratings and watching history.
- **Yelp** is a rating website where users can write reviews and rate various businesses. We select users (U), restaurants (R), 10 cities (C), and attribute information of users and businesses such as tags, ratings and food preferences to construct our attributed heterogeneous Yelp dataset.

5.1.2. Baselines

To demonstrate the superiority of our approach, we compare IDCD with 17 popular or state-of-the-art methods in two classes: traditional community detection methods, and embedding based community detection methods.

(1) Traditional Community Detection Methods.

- **Louvain** [42] is a heuristic method based on modularity optimization for homogeneous information networks, and is widely used because of its speed.
- **CNM** [43] is a hierarchical agglomeration algorithm for homogeneous networks which greedily optimizes the modularity with linear running time.
- **FluidC** [44] is the first propagation-based algorithm which is able to identify a variable number of communities in homogeneous networks.
- **SClump** [16] is a spectral clustering method which constructs a similarity matrix with different meta-paths to detect communities in HINs.

(2) Embedding based Community Detection Methods. The former nine methods are network embedding methods, and we apply k-means to their obtained embeddings for community detection tasks. The latter five methods embed nodes and detect communities in a unified way.

- **Node2vec** [37] is a representative random walk-based embedding model in homogeneous networks and embeds nodes via a biased random walk procedure.
- **VGAE** [45] is a variational inference-based model which proposes a variational auto-encoder to embed nodes in homogeneous information networks in an unsupervised way.

[2] https://www.aminer.cn/citation.
[3] https://dblp.uni-trier.de.
[4] https://grouplens.org/datasets/movielens/25m/.
[5] https://www.yelp.com/dataset.
[6] https://scholar.google.com/citations?viewop=topvenues.
[7] http://movielens.org/.

- **Metapath2vec** [28] is a representative random walk-based embedding algorithms in HINs. It exploits meta-paths and proposes a heterogeneous skip-gram model to learn node embeddings.
- **HIN2Vec** [30] learns embeddings of both nodes and meta-paths in the HIN by predicting relationships between nodes.
- **HHNE** [46] employs meta-path guided random walk and embeds nodes in HINs into a hyperbolic space.
- **HDGI** [47] disassembles HINs into homogeneous graphs with meta-paths and obtains node embeddings by maximizing local–global mutual information.
- **HAN** [24] is a HGNN-based method which adopts hierarchical attention (node-level and semantic-level attentions) to embed nodes in AHINs.
- **HeCo** [26] contrasts network schema view and meta-path view to embed nodes in AHINs in a self-supervised way.
- **ComE** [48] designs a closed loop among community embedding, community detection and node embedding, and jointly optimizes them to embed nodes in homogeneous information networks.
- **DEC** [40] learns node embeddings and cluster assignments by designing a self-training clustering objective.
- **SDCN** [49] introduces a dual self-supervised mechanism to unify GCN and DNN models to cluster nodes in homogeneous information networks.
- **GUCD** [50] combines Markov Random Fields with a GCN layer as an encoder, and introduces a community-centric dual decoder to detect communities in homogeneous information networks.
- **O2MAC** [15] reconstructs multiple graph views with one informative graph view to learn node embeddings, and utilizes a self-training clustering objective to detect communities in AHINs.

5.1.3. Implementation details

Parameter settings. For a fair comparison, we run each algorithm 20 times and report the best results. For methods designed for homogeneous information networks, we map the original AHINs into homogeneous ones by following the semantic-related meta-path (marked by ⋆ in Table 2). For methods designed for HINs that require multiple meta-paths to detect communities, we feed them with all meta-paths listed in Table 2. For the semi-supervised method HAN, the split ratios of the training set, validation set and test set are as same as the original paper. For random walk-based methods including node2vec, metapath2vec, HIN2vec, and HHNE, we set the window size, walk length, number of walks, and the number of negative samples following their original papers or tune to get their best experimental results. For self-training based methods SDCN, O2MAC and IDCD, we train node representation learning process individually at the first 30 epochs to initialize the embedding weight parameters. Embedding dimensions of embedding-based baselines are tuned to be optimal. For the proposed IDCD, we set the number of walks *wk* to 5, walk length *len* to 10, negative sample to 2 and window size *w* to 5. The parameter α is tuned in the range of $[0, 1]$. The sizes of hidden layer dimensions are set to 64, 32, 4 and 10 with respect to AMiner, DBLP, MovieLens, and Yelp datasets, respectively. All baselines are conducted on a Tesla V100-PCIE-32 GB GPU.

Evaluation metrics. We adopt three commonly used metrics to evaluate the community detection performance: Clustering Accuracy (ACC), Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI) [51]. ACC finds the best matching between the community assignment taken from a community detection method and a ground truth assignment. NMI is a normalization of the Mutual Information score ranging from 0 (no mutual information) to 1 (perfect correlation). The value of ARI is in the interval $[-1.0, 1.0]$, where negative numbers indicate poor clustering results, 0.0 stands for random labeling, and 1.0 for perfect matching. The higher these metric values are, the better the community detection performance is.

5.2. Results (RQ1): The effectiveness

Since most baselines are designed to detect communities for the same-type nodes, and a few baselines embed heterogeneous nodes, we set two scenarios: (1) detecting communities for same-type nodes (marked with ⋆ in Table 2); (2) detecting communities for multiple-type nodes.

(1) Detecting communities for *same-type nodes*. We make the following observations from Table 3:

- Generally, network embedding-based methods perform better than most traditional methods. Unified community detection strategies are better than two-stage strategies. Note that SClump achieves satisfying results because it utilizes eigen-decomposition to obtain k most important eigenvectors, which shares similar ideas with network embedding-based methods to represent nodes with key latent features. In addition, not as expected, the performances of O2MAC are poor and do not conform with the above observations. That might be because the most informative view selected via modularity in O2MAC is not consistent with the community detection task, presumably providing harmful signals.
- We can observe that IDCD achieves the best community detection result for a single type of nodes in AHINs, indicating its effectiveness. In particular, IDCD improves the best baselines by relatively 0.56% to 3.68% in ACC scores, 0.92% to 5.91% in NMI scores and 0.86% to 4.06% in ARI scores over the four datasets. It can be attributed to that IDCD avoids noisy information and captures all kinds of related heterogeneous information for the task. Furthermore, by learning node representation and identifying communities simultaneously, IDCD enables a community-aware node representation for community detection.

(2) Detecting communities for *multiple-type nodes*.

From Table 4, we observe that IDCD outperforms the other baselines across different evaluation metrics except for ARI on DBLP. Note that HHNE also performs well and has the same setting as ours: following one meta-path to reconstruct network structures, free from noisy information. However, HHNE is an embedding method and is independent of the community detection task. Such a pipeline way is prone to error propagation and results in suboptimal performances.

Furthermore, we utilize t-SNE [52] to visualize the community structure information within the learned node embeddings. We show representative visualization results on the DBLP network in Fig. 3. Points in HeCo are mixed together, and in SClump gather into a bunch of little clusters, but they have no clear community structures. The results of O2MAC are a little better but the red cluster, cyan cluster and blueviolet cluster are tightly close to each other. HHNE is much better but fails to cluster points in red. Our IDCD performs the best, and the boundaries of each community are clear.

5.3. Result (RQ2): Effects of different soft assignment distribution Q

Inspired by the most trending clustering model DEC [40], most network embedding-based clustering methods [15,49,53] adopt Student's t-distribution as the soft assignment distribution Q (referred to as kernel function) to measure similarities among embeddings and community centroids. However, we find that Student's t-distribution is not a suitable and optimal choice for depicting the membership between nodes and communities.

To understand that, we conduct several experiments (shown in Fig. 4) and give theoretical discussions (shown in Fig. 5).

Experimental Verification. To examine the effects of the different kernel functions for the community detection task, we select three commonly used distributions as the kernel distribution: Gaussian mixture distribution (GMD), Gaussian distribution (GD), and Student's

Table 3
Community detection results (%) of **same-type** node. The best results are shown in bold, and the second-best results are underlined. "–" indicates the datasets are too large for algorithms to execute.

Method	Dataset											
	DBLP			MovieLens			Yelp			AMiner		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
Louvain	51.84	44.98	34.77	62.78	46.57	44.65	48.86	33.24	38.85	82.42	62.22	61.48
CNM	57.60	49.53	31.09	55.97	36.98	31.57	53.31	39.48	46.17	–	–	–
FluidC	48.91	26.71	20.88	65.97	39.04	40.40	[a]	[a]	[a]	68.23	53.02	57.60
SClump	82.08	60.69	62.85	87.92	73.13	69.05	56.65	40.51	33.89	–	–	–
Node2vec	54.12	39.53	30.23	50.95	36.17	27.03	47.20	43.01	23.72	83.55	67.32	59.69
VGAE	59.21	45.75	41.34	47.08	37.44	25.13	37.44	34.64	31.85	84.14	64.32	62.84
Metapath2vec	63.72	43.72	42.82	50.04	30.26	20.09	24.70	15.83	15.42	81.33	65.38	68.02
HIN2Vec	62.53	42.31	41.41	55.07	32.76	22.98	35.64	15.19	13.10	86.86	68.11	70.20
HHNE	95.27	83.28	88.83	75.48	57.99	56.12	49.01	45.81	27.30	87.07	67.55	68.36
HDGI	80.26	58.88	60.48	75.10	63.02	49.97	27.52	28.14	14.80	85.82	66.22	67.13
HAN	58.14	35.96	28.51	71.48	56.61	46.10	–	–	–	–	–	–
HeCo	85.06	65.95	67.42	63.03	52.12	44.22	29.26	20.82	15.17	–	–	–
ComE	65.72	31.95	25.96	38.67	41.15	65.17	59.31	51.32	34.07	–	–	–
SDCN	60.08	45.96	34.75	56.59	31.96	25.66	52.36	19.66	19.31	–	–	–
GUCD	63.47	31.90	30.20	66.40	45.33	45.14	–	–	–	–	–	–
DEC	56.03	28.86	28.58	74.23	63.05	59.39	31.23	18.65	12.81	47.53	12.26	10.34
O2MAC	63.12	39.14	32.97	49.17	28.09	21.08	24.65	0.26	0.55	–	–	–
IDCD	**96.39**	**84.20**	**89.69**	**88.60**	**78.92**	**73.11**	**59.87**	**52.74**	**48.20**	**90.75**	**74.02**	**72.51**

[a] Indicates the dataset is not fully connected, so the propagation-based algorithm FluidC cannot run on it.

Table 4
Community detection results (%) of **multiple-type** nodes on four datasets.

Method	Dataset											
	DBLP			MovieLens			Yelp			AMiner		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
Metapath2vec	55.85	41.72	30.21	49.85	30.27	20.10	24.97	15.06	15.00	81.32	69.95	65.37
HIN2Vec	60.97	40.29	30.30	54.93	32.70	22.86	28.46	22.20	11.85	86.31	66.64	69.08
HHNE	97.22	89.21	**93.66**	75.78	58.06	56.39	55.05	56.62	34.33	89.00	71.90	73.24
IDCD	**97.32**	**89.46**	92.14	**93.06**	**86.55**	**80.45**	**64.29**	**67.13**	**54.80**	**91.37**	**78.08**	**82.33**

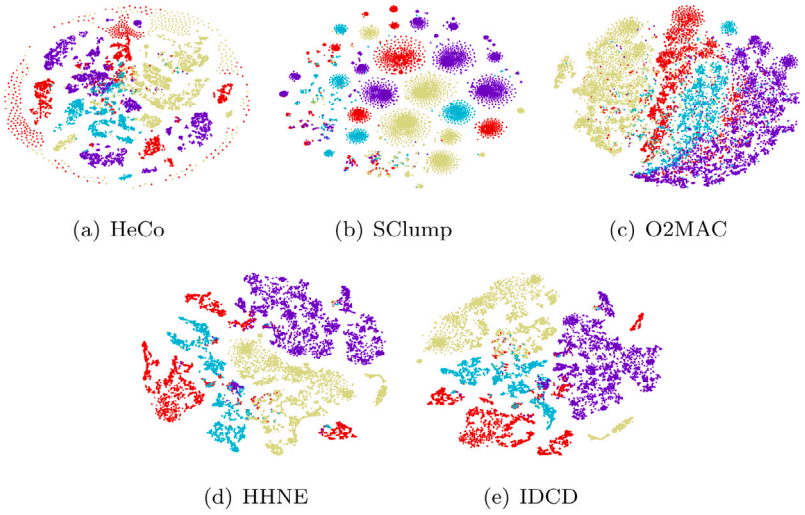
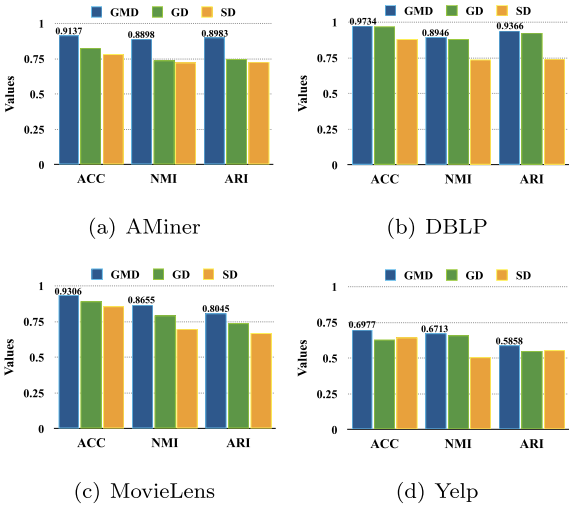


Fig. 3. Visualization of the learned representations on DBLP. Each point indicates one paper. Different colors indicate different communities: cyan for *Machine Learning*, red for *Data Mining*, lime green for *Database*, and blueviolet for *Computer Vision*.

t-distribution (SD). For a fair comparison, the initial values of community centroids, parameters of neural networks, and embeddings are all the same. We sequentially use GMD, GD, and SD as IDCD's kernel function to perform community detection tasks.

From Fig. 4, we observe that adopting GMD as kernel function achieves the best performance over three metrics on four datasets, and its superiority is extremely obvious on AMiner, MovieLens, and Yelp. Meanwhile, GD is a better choice than SD. The kernel function is more conducive to community detection. This is because most of the real-world objects in the same group follow the Gaussian distribution and multiple groups of objects follow the Gaussian mixture distribution naturally, while the Student's t-distribution is not that suitable for clustering.

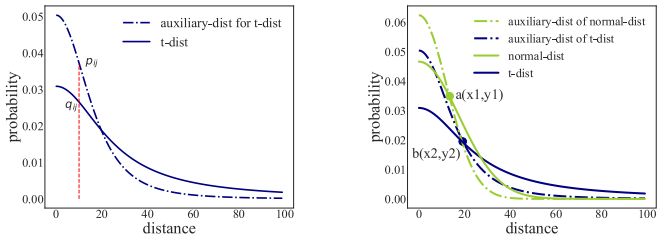
Theoretical Discussion. Since the Gaussian mixture model is a more common model for clustering, we would like to investigate why most



(a) AMiner (b) DBLP

(c) MovieLens (d) Yelp

Fig. 4. The performance of community detection on four datasets with Gaussian Mixture Distribution (GMD), Gaussian Distribution (GD) and Student's t-Distribution (SD) as soft assignment distribution (kernel function) of IDCD, respectively.



(a) t-dist vs. its auxiliary-dist (b) t-dist vs. normal-dist

Fig. 5. Distance of a node to a community centroid vs. Probability of the node belongs to the community. The t-dist and normal-dist is short for Student's t-distribution and normal distribution, respectively.

of the existing methods choose SD but not GMD to depict community assignments. The first method adopting Student's t-distribution to measure the similarities, DEC, [40] takes inspiration from t-SNE [52]. t-SNE is a dimension reduction method that uses a Student's t-distribution to compute the similarity between two points in the *low*-dimensional space and a Gaussian distribution in the *original* space. In this way, minimizing the KL divergence between these two distributions can alleviate the *crowding problem*, showing aggregation.

However, these self-training-based deep clustering models do not compare similarities of embeddings from two spaces as t-SNE does, they compare similarities measured by two different distributions in one space. In other words, k-means and self-training strategy contribute to the clustering process, instead of kernel function. k-means provide nice centroids and self-training improves them. As shown in Fig. 5(a), the blue solid line represents a part of the probability density function of a node that obeys the Student's t-distribution and belongs to one cluster, and the blue dash-dot line indicates its corresponding auxiliary distribution. When a node belongs to a cluster with a high probability q_{ij}, its corresponding auxiliary function will provide a higher belonging probability p_{ij}. By approximating the auxiliary function, this node will be given higher credibility in this cluster which helps to optimize the clustering and node representation and makes these methods perform well.

Moreover, for clustering, it is better to choose the distribution which can depict the community assignments of nodes, *i.e.*, Gaussian mixture distribution. As shown in Fig. 5(b), a and b are the intersections of two distributions (Student's t-distribution and Gaussian distribution) and their corresponding auxiliary distributions. Gaussian distribution is more restrictive ($x1 < x2$ and $y1 > y2$) than Student's t-distribution for a node belonging to a cluster, reducing false pseudo labels and improving the performance of the self-training process.

5.4. Results (RQ3): Parameter sensitivity

We perform sensitivity analysis on four critical hyper-parameters in IDCD: the number of window sizes w for selecting negative samples, the walk length len denotes the maximum number of steps in each random walk, the number of walks wk determines the number of instances of random walks starting from each node, and the dimension of node representation d_e to embed the information. The former three parameters are important to meta-path-based random walks. Fig. 6 shows how these parameters of IDCD affect the performance on all four datasets. We have the following observations:

- For the **window size** w, increasing the number of negative samples generally enhances the performance of our proposed IDCD across most datasets, but this improvement plateaus at a certain saturation point. For example, on the MovieLens and Yelp datasets, as depicted in the first column of the last two lines in Fig. 6, an excessive number of negative samples can potentially lead to diminished performance.
- Prolonging the **walk length** len typically results in better performance across all datasets, indicating the importance of walk length in the quality of network representation. However, the existence of an optimal walk length implies that beyond this point, additional network structural information may begin to hinder the learning process due to redundancy.
- For the **number of walks** wk, IDCD can achieve commendable results with a relatively small value, such as $wk = 5$ on the Aminer dataset, which is more efficient than other meta-path-based random walk methods. For comparison, metapath2vec achieves satisfactory results when $wk = 600$, highlighting that the performance benefit of IDCD may be attributed to the usage of node attributes.
- The choice of the **representation dimension** d_e is crucial and can significantly affect the performance of the model. It necessitates careful adjustment based on the specific characteristics of the dataset and the requirements of the algorithm. For example, on the MovieLens dataset, there is a marked decline in IDCD performance as d_e increases, which may be attributed to higher embedding dimensions that could capture excessive noise, adversely affecting community detection. In contrast, the Yelp dataset, with its high dimensionality of node attributes in the original space, necessitates a higher d_e to ensure these attributes are well represented in the embedding space.

5.5. Results (RQ4): Case study

IDCD holds significant value in the domains of pattern recognition and explainable AI [54]. We select two different meta-paths on a mini MovieLens dataset to show (1) how they lead to different community detection results, (2) how they make community detection results explainable. The mini MovieLens dataset is extracted from the original MovieLens dataset but only contains four genres (*comedy*, *action*, *horror*, *children*) of movies released before 2008, and corresponding actors and directors. The statistics of the dataset are shown in Table 5.

We conduct two different community detection tasks:

- **TASK 1**. Cluster *directors* according to the type of movies they directed:
  – Specify semantic-related *meta-path*: D-M-G-M-D.
- **TASK 2**. Cluster *actors* according to the type of movies they starred in:
  – Specify semantic-related *meta-path*: A-M-G-M-A.

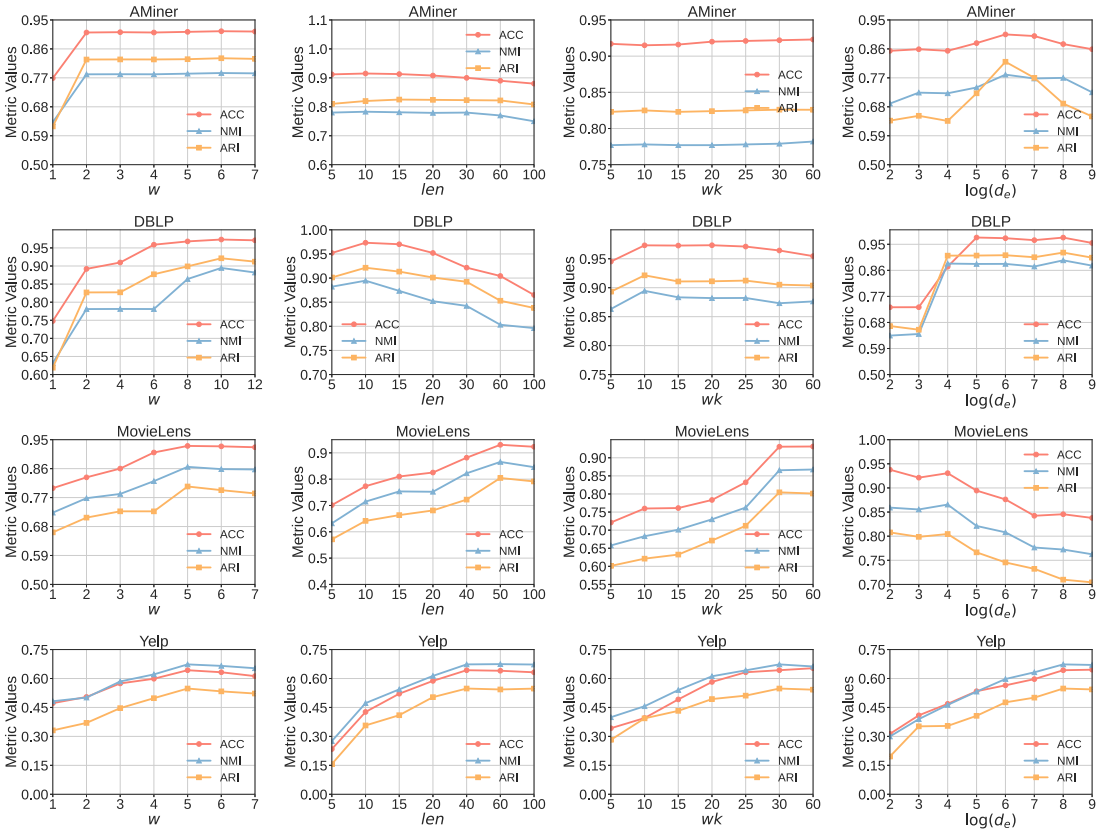


Fig. 6. Experiments of parameter sensitivity (including the window size w, walk length len, number of walks wk, and representation dimension d_e) on Aminer, DBLP, MovieLens, and Yelp datasets.

Table 5
MovieLens dataset used in the case study.

Node types	# Nodes	Relation	# Relations
Movie (M)	2175	AM	7986
Actor (A)	389	DM	1096
Director (D)	159	MG	2844
Genre (G)	4		

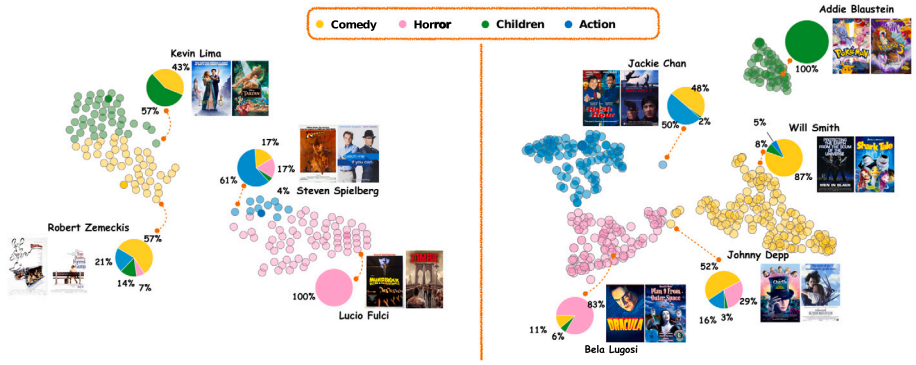
We use t-SNE [52] to visualize the representations of nodes in a two-dimensional space. As shown in Fig. 7, there are four color nodes, corresponding to four genres. In Fig. 7(a), we observe that the embeddings of *comedy* directors are close to those of *children* movies, because *children* movies always have comedy elements. And it is the same as the embeddings of *horror* and *action* directors. The situation is a little different in *Actor* communities. As shown in Fig. 7(b), comedy, horror, and action actors are located close to each other. While the actors in *children* movies are usually voice actors, thus far away from them.

For clearly understand node community assignments, we further select a few representative nodes and enlarge them. The selected nodes are re-marked with big dots, and the proportions of multiple colors indicate their memberships to different communities. The posters of their representative works are listed. Specifically, *Lucio Fulci* and *Addie Bloustein* are painted only in one color, because they just directed or participated in one genre of movie in this dataset; Both *Lucio Fulci* and *Addie Bloustein* have participated in various kinds of movies, but their representative movies are mostly *horror* and *action*, so one color occupies most of the area. *Jackie Chan*, the most well-known action star, can be considered belonging to the *Action* and *Comedy* community simultaneously. It can be ascribed to the movies he participated in are mostly action comedies. We conclude that if a node is located in the middle of the area where the same-color nodes are located, then this node has a high degree of membership in this community; Conversely, if a node is located at the junction of different colors, then this node may belong to multiple communities.

Moreover, as shown in Table 6, directors and actors are gathered into four communities: *Comedy*, *Horror*, *Children*, and *Action*. We enumerate some representative samples from each of the four communities and sort them according to their membership of the community (shown in the bracket). It could be found that actors and directors are assigned to the communities they are famous for. For example, *Woody Allen* directed and participated in a large number of drama and linear comedy movies, and he is assigned to the community *Comedy* no matter being an actor or director; Kung-Fu star *Jet Li* is naturally assigned to the *Action* community; *Desmond Llewelyn* is assigned to community *Action* with a high score because his most well-known masterpiece is the role of Mr. Q, who specializes in developing weapons and props for Bond in the James Bond series.

6. Conclusion

In this work, we propose IDCD, an interest-driven community detection method to cluster heterogeneous nodes. With a user-specified meta-path, IDCD leverages all kinds of heterogeneous information sources and aligns the community detection process to the network embedding process which is optimized in a unified way. Specifically, (1) we propose a sampling strategy via aggregating heterogeneous node attributes and topological structures to select informative nodes for better embeddings. (2) We find that replacing the Student's t-distribution with Gaussian mixture distribution improves the performance. Experiments conducted on four real-world datasets reveal the effectiveness of IDCD, and the case study provides reasonable explanations of the obtained results. Future work can be concentrated on automatically inferring the number of communities and detecting communities simultaneously.



(a) **Task 1.** Cluster *directors* according to the type of movies they have directed.

(b) **Task 2.** Cluster *actors* according to the type of movies they starred in.

Fig. 7. Visualization of community detection results and node community assignments under two different meta-paths guidance.

Table 6
Representative samples in four communities discovered by IDCD.

Meta-path	Community	Samples
AMGMA	Comedy	Jack Lemmon (1), Woody Allen (0.971), Adam Sandler (0.962), Tom Hanks (0.818)
	Horror	Dwight Frye (1), Michael Mark (0.889), Bela Lugosi (0.833), Tom Savini (0.636)
	Children	Addie Blaustein (1), Kath Soucie (0.684), Jim Cummings (0.606), Danny Mann (0.603)
	Action	Desmond Llewelyn (0.889), Jet Li (0.751), Samuel L. Jackson (0.553), Tom Cruise (0.524)
DMGMD	Comedy	Woody Allen (0.912), Brian Robbins (0.802), Brian Levant (0.615), Tim Burton (0.601)
	Horror	Tod Browning (1), George A. Romero (0.889), Tobe Hooper (0.875), Rob Zombie (0.812)
	Children	Wolfgang Reitherman (0.712), Robert Stevenson (0.653), Don Bluth (0.753), Clyde Geronimi (0.571)
	Action	Paul W.S. Anderson (0.7), Renny Harlin (0.642), William Friedkin (0.625), Steven Spielberg (0.609)

CRediT authorship contribution statement

Mengyue Liu: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Jun Liu:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition. **Yixiang Dong:** Writing – review & editing, Validation, Methodology, Formal analysis. **Rui Mao:** Writing – review & editing, Visualization. **Erik Cambria:** Writing – review & editing, Supervision, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

This work was supported by National Key Research and Development Program of China (2020AAA0108800), National Natural Science Foundation of China (62137002, 61937001, 62176209, 62176207, 62106190, 62192781, 61877050 and 62050194), Innovative Research Group of the National Natural Science Foundation of China (61721002), Innovation Research Team of Ministry of Education (IRT_17R86), The National Social Science Fund of China (18XXW005), Consulting research project of Chinese academy of engineering "The Online and Offline Mixed Educational Service System for 'The Belt and Road' Training in MOOC China", China Postdoctoral Science Foundation (2020M683493), Project of China Knowledge Centre for Engineering Science and Technology, "LENOVO-XJTU" Intelligent Industry Joint Laboratory Project, and the Fundamental Research Funds for the Central Universities, China (xzy022021048, xpt012021005, xhj032021013-02).

References

[1] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3–5) (2010) 75–174.
[2] S. Fortunato, D. Hric, Community detection in networks: A user guide, Phys. Rep. 659 (2016) 1–44.
[3] Q. Chen, Y. Qiao, F. Hu, Y. Li, K. Tan, M. Zhu, C. Zhang, Community detection in complex network based on APT method, Pattern Recognit. Lett. 138 (2020) 193–200.
[4] A. Clauset, Finding local community structure in networks, Phys. Rev. E 72 (2) (2005) 026132.
[5] S. Cavallari, E. Cambria, H. Cai, K.C.-C. Chang, V.W. Zheng, Embedding both finite and infinite communities on graphs [application notes], IEEE Comput. Intell. Mag. 14 (3) (2019) 39–50.
[6] H. Yin, Q. Wang, K. Zheng, Z. Li, J. Yang, X. Zhou, Social influence-based group representation learning for group recommendation, in: 2019 IEEE 35th International Conference on Data Engineering, ICDE, IEEE, 2019, pp. 566–577.
[7] S.S. Bhowmick, B.S. Seah, Clustering and summarizing protein-protein interaction networks: A survey, IEEE Trans. Knowl. Data Eng. 28 (3) (2015) 638–658.
[8] P. Chen, S. Redner, Community structure of the physical review citation network, J. Informetr. 4 (3) (2010) 278–290.
[9] Y. Sun, Y. Yu, J. Han, Ranking-based clustering of heterogeneous information networks with star network schema, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 797–806.
[10] C. Shi, Y. Li, J. Zhang, Y. Sun, S.Y. Philip, A survey of heterogeneous information network analysis, IEEE Trans. Knowl. Data Eng. 29 (1) (2016) 17–37.
[11] Y. Xie, B. Yu, S. Lv, C. Zhang, G. Wang, M. Gong, A survey on heterogeneous network representation learning, Pattern Recognit. 116 (2021) 107936.
[12] Y. Sun, J. Han, X. Yan, P.S. Yu, T. Wu, Pathsim: Meta path-based top-k similarity search in heterogeneous information networks, Proc. VLDB Endow. 4 (11) (2011) 992–1003.
[13] Y. Fang, Y. Yang, W. Zhang, X. Lin, X. Cao, Effective and efficient community search over large heterogeneous information networks, Proc. VLDB Endow. 13 (6) (2020) 854–867.
[14] H. Gui, J. Liu, F. Tao, M. Jiang, B. Norick, L. Kaplan, J. Han, Embedding learning with events in heterogeneous information networks, IEEE Trans. Knowl. Data Eng. 29 (11) (2017) 2428–2441.

[15] S. Fan, X. Wang, C. Shi, E. Lu, K. Lin, B. Wang, One2multi graph autoencoder for multi-view graph clustering, in: Proceedings of the Web Conference 2020, 2020, pp. 3070–3076.

[16] X. Li, B. Kao, Z. Ren, D. Yin, Spectral clustering in heterogeneous information networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 4221–4228.

[17] Y. Sun, B. Norick, J. Han, X. Yan, P.S. Yu, X. Yu, Pathselclus: Integrating meta-path selection with user-guided object clustering in heterogeneous information networks, ACM Trans. Knowl. Discov. Data (TKDD) 7 (3) (2013) 1–23.

[18] X. Li, Y. Wu, M. Ester, B. Kao, X. Wang, Y. Zheng, SCHAIN-IRAM: An efficient and effective semi-supervised clustering algorithm for attributed heterogeneous information networks, IEEE Trans. Knowl. Data Eng. (2020).

[19] S. Zhou, J. Bu, Z. Zhang, C. Wang, L. Ma, J. Zhang, Cross multi-type objects clustering in attributed heterogeneous information network, Knowl.-Based Syst. 194 (2020) 105458.

[20] R.A. Khan, M. Kleinsteuber, A framework for joint unsupervised learning of cluster-aware embedding for heterogeneous networks, 2021, arXiv preprint arXiv:2108.03953.

[21] T. Zhao, C. Yang, Y. Li, Q. Gan, Z. Wang, F. Liang, H. Zhao, Y. Shao, X. Wang, C. Shi, Space4HGNN: A novel, modularized and reproducible platform to evaluate heterogeneous graph neural network, 2022, arXiv preprint arXiv:2202.09177.

[22] M.J. Barber, Modularity and community detection in bipartite networks, Phys. Rev. E 76 (6) (2007) 066102.

[23] J. Zhang, Y. Chen, Modularity based community detection in heterogeneous networks, Statist. Sinica 30 (2) (2020) 601–629.

[24] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P.S. Yu, Heterogeneous graph attention network, in: The World Wide Web Conference, 2019, pp. 2022–2032.

[25] Z. Hu, Y. Dong, K. Wang, Y. Sun, Heterogeneous graph transformer, in: Proceedings of the Web Conference 2020, 2020, pp. 2704–2710.

[26] X. Wang, N. Liu, H. Han, C. Shi, Self-supervised heterogeneous graph neural network with co-contrastive learning, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 1726–1736.

[27] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, D. Wagner, On Modularity-Np-Completeness and Beyond, Tech. Rep 19, ITI Wagner, Faculty of Informatics, Universität Karlsruhe (TH), 2006, p. 2006.

[28] Y. Dong, N.V. Chawla, A. Swami, Metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.

[29] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.

[30] T.-y. Fu, W.-C. Lee, Z. Lei, Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 1797–1806.

[31] C. Zhang, D. Song, C. Huang, A. Swami, N.V. Chawla, Heterogeneous graph neural network, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 793–803.

[32] X. Li, Y. Wu, M. Ester, B. Kao, X. Wang, Y. Zheng, Semi-supervised clustering in attributed heterogeneous information networks, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 1621–1629.

[33] G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (7043) (2005) 814–818.

[34] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014, pp. 1112–1119.

[35] S. Gregory, Fuzzy overlapping communities in networks, J. Stat. Mech. Theory Exp. 2011 (02) (2011) P02017.

[36] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 701–710.

[37] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.

[38] L. Zhu, W. Li, R. Mao, V. Pandelea, E. Cambria, PAED: Zero-shot persona attribute extraction in dialogues, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, ACL, Vol. 1, 2023, pp. 9771–9787.

[39] C.C. Aggarwal, C.K. Reddy, Data Clustering, Algorithms and Applications, in: Chapman&Hall/CRC Data Mining and Knowledge Discovery Series, Londra, 2014.

[40] J. Xie, R. Girshick, A. Farhadi, Unsupervised deep embedding for clustering analysis, in: International Conference on Machine Learning, 2016, pp. 478–487.

[41] M. Ge, R. Mao, E. Cambria, Explainable metaphor identification inspired by conceptual metaphor theory, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 10681–10689.

[42] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech.: Theory Exp. 2008 (10) (2008) P10008.

[43] A. Clauset, M.E. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 066111.

[44] F. Parés, D.G. Gasulla, A. Vilalta, J. Moreno, E. Ayguadé, J. Labarta, U. Cortés, T. Suzumura, Fluid communities: a competitive, scalable and diverse community detection algorithm, in: International Conference on Complex Networks and their Applications, Springer, 2017, pp. 229–240.

[45] T.N. Kipf, M. Welling, Variational graph auto-encoders, 2016, arXiv preprint arXiv:1611.07308.

[46] X. Wang, Y. Zhang, C. Shi, Hyperbolic heterogeneous information network embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 5337–5344.

[47] Y. Ren, B. Liu, C. Huang, P. Dai, L. Bo, J. Zhang, Heterogeneous deep graph infomax, 2019, arXiv preprint arXiv:1911.08538.

[48] S. Cavallari, V.W. Zheng, H. Cai, K.C.-C. Chang, E. Cambria, Learning community embedding with community detection and node embedding on graphs, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 377–386.

[49] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, P. Cui, Structural deep clustering network, in: Proceedings of the Web Conference 2020, 2020, pp. 1400–1410.

[50] D. He, Y. Song, D. Jin, Z. Feng, B. Zhang, Z. Yu, W. Zhang, Community-centric graph convolutional network for unsupervised community detection, in: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, 2021, pp. 3515–3521.

[51] T. Chakraborty, A. Dalmia, A. Mukherjee, N. Ganguly, Metrics for community analysis: A survey, ACM Comput. Surv. 50 (4) (2017) 1–37.

[52] L.v.d. Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (Nov) (2008) 2579–2605.

[53] H. Sun, F. He, J. Huang, Y. Sun, Y. Li, C. Wang, L. He, Z. Sun, X. Jia, Network embedding for community detection in attributed networks, ACM Trans. Knowl. Discov. Data (TKDD) 14 (3) (2020) 1–25.

[54] E. Cambria, R. Mao, M. Chen, Z. Wang, S.-B. Ho, Seven Pillars for the future of artificial intelligence, IEEE Intell. Syst. 38 (6) (2023) 62–69.