Heterogeneous Information Network Embedding with Adversarial Disentangler

Ruijia Wang*, Chuan Shi*†, *Member, IEEE*, Tianyu Zhao, Xiao Wang, Yanfang Ye

Abstract—Heterogeneous information network (HIN) embedding has gained considerable attention in recent years, which learns low-dimensional representation of nodes while preserving the semantic and structural correlations in HINs. Many of existing methods which exploit meta-path guided strategy have shown promising results. However, the learned node representations could be highly entangled for downstream tasks; for example, an author's publications in multidisciplinary venues may make the prediction of his/her research interests difficult. To address this issue, we develop a novel framework named HEAD (*i.e.*, HIN Embedding with Adversarial Disentangler) to separate the distinct, informative factors of variations in node semantics formulated by meta-paths. More specifically, in HEAD, we first propose the meta-path disentangler to separate node embeddings from various meta-paths into intrinsic and specific spaces; then with meta-path schemes as self-supervised information, we design two adversarial learners (*i.e.*, meta-path and semantic discriminators) to make the intrinsic embedding more independent from the designed meta-paths while the specific embedding more meta-path dependent. To comprehensively evaluate the performance of HEAD, we perform a set of experiments on four real-world datasets. Compared to the state-of-the-art baselines, the maximum 15% improvement of performance demonstrates the effectiveness of HEAD and the benefits of the learned disentangled representations.

Index Terms—Heterogeneous Information Networks, Network Embedding, Representation Disentanglement

1 Introduction

Heterogeneous information networks (HINs) [1], involving different types of nodes and relations, are ubiquitous in the real world, ranging from bibliographic and social networks to transportation and telecommunication systems [2]. With heterogeneous types of nodes and relations, HINs are able to model complex interactions and immensely rich semantics in real-world scenarios. Thus, HIN analysis has emerged as a promising direction for many data mining tasks [3]. With the surge of network embedding [4], [5], numerous recent research has shifted towards HIN embedding [6], [7], [8], [9], [10], [11], which aims to project the nodes into a low-dimensional space whilst preserving the structural and semantic properties of HINs. The learned low-dimensional embedding has been a *de facto* solution towards fundamental problems in various applications, such as node classification [12], [13], link prediction [14], [15] and recommendation [16], [17].

HIN embedding has recently attracted considerable attention, the major line of which [7], [8], [16], [18] adopts meta-path [19] to retain both structural and semantic correlations between nodes. Specifically, meta-path is a type-constrained relation sequence connecting two nodes, which is regarded as a basic tool in HINs to extract sub-structure and semantic. As illustrated in Figure 1 (a) and (b) for the bibliographic data, the toy HIN consists of multiple types of nodes and relations; meta-path APA and APCPA describe the co-author and co-conference relationship between two authors, respectively. These meta-path based HIN embedding methods either use only one meta-path inevitably resulting in information loss, or rely on certain strategies to merge multiple meta-paths, which often boil down to a two-step framework as displayed in Figure 1 (c). Firstly, they extract node sequences along each meta-path and embed them with optimization models, such as skip-gram [20]. Secondly, the node embeddings learned by multiple meta-paths are fused with some strategies (*e.g.*, concatenation or weighted combination [7], [18]) to obtain the final representations. Through the above operations, they produce structure- and semantic-preserving embeddings.

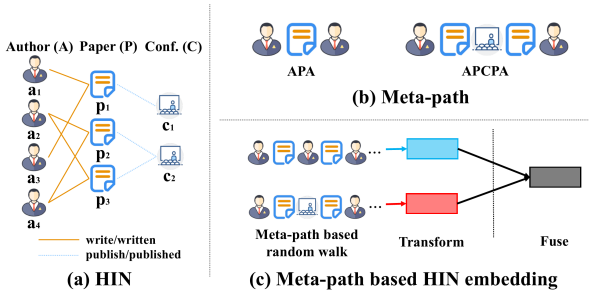


Fig. 1: A toy example of heterogeneous information network and the two-step framework of meta-path based HIN embedding methods.

Although these meta-path based HIN embedding methods have demonstrated great success in learning representations, they may fail to uncover the latent explanatory factors hidden in the node embeddings under various meta-paths. Taking meta-paths APA and APCPA in Figure 1 as instances,

* *Both authors contributed equally to this research.*
† *indicates corresponding author.*
Ruijia Wang, Chuan Shi, Tianyu Zhao and Xiao Wang are with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China.
E-mail: {wangruijia,shichuan,tyzhao,xiaowang}@bupt.edu.cn
Yanfang Ye is with the Department of Computer and Data Sciences, Case Western Reserve Univerity, USA.
E-mail: yanfang.ye@case.edu.

an author's publications with interdisciplinary authors or in multidisciplinary venues may make the prediction of his/her research interests difficult. To address this issue, we review the role of meta-path in HIN analysis, and have the following two insights. (1) Meta-paths are highly correlated, and their common part exposes intrinsic factors of nodes which absorb informative characteristic from all meta-paths and could more essentially reflect the characteristics. Meta-paths APCPA and APA depict the micro co-author and macro co-conference relations respectively, but they actually capture the intrinsic correlation among authors, *i.e.*, the same research interests, from different perspectives. (2) Meta-paths describe distinguished sub-structures from various aspects, and their specific parts sometimes interfere with the analysis of nodes once connections along meta-path are sparse or noisy [8], [16]. For instance, a broad-subject conference in APCPA may provide misleading information for predicting author's research interest, so does a broad-interest author in APA. Therefore, the mentioned highly-correlated but distinguished sub-structures explored by meta-paths provide an impetus to purify out intrinsic factors during meta-path fusion.

Nevertheless, the type sequences among meta-paths are overlapping somehow but different, revealing a high degree of semantic entanglement. Although aware of the necessity of extracting intrinsic factors, its materialization is non-trivial. (1) *How to determine intrinsic and specific factors of nodes extracted by multiple meta-paths?* From the perceptual understanding of intrinsic and specific factors, the intrinsic factors should be invariant to meta-paths and correspond to the essential feature of nodes, while the specific factors should depend on the meta-path specific semantic. For example, the author's research interests determine which conferences he will publish and which researchers he will collaborate with, thus should be invariant to meta-paths APCPA and APA. But misleading information brought by specific conferences and collaborators is strongly meta-path dependent. (2) *How to disentangle intrinsic and specific factors without explicit supervised information?* There are no supervised labels about intrinsic or specific factors. The author's research interests are sometimes expensive to obtain, let alone meta-path dependent misleading information. Hence, it is imperative to design a mechanism that utilizes the applicable self-supervised information to guarantee the disentanglement.

In this paper, we make the first attempt to employ representation disentanglement on HINs to disentangle intrinsic and specific node embedding from multiple meta-paths. We propose HEAD, a novel **H**IN **E**mbedding framework with **A**dversarial **D**isentangler. To detach intrinsic and specific embedding, a meta-path disentangler is designed to encode node embedding generated by each meta-path into intrinsic and specific space. Furthermore, utilizing meta-path schemes as self-supervised information, two adversarial learners (*i.e.*, meta-path and semantic discriminator) are deployed to try the best to separate intrinsic and specific embedding. Specifically, the meta-path discriminator learns a more intrinsic embedding while the semantic discriminator enhances the correlation between specific embedding and meta-path semantic. The major contributions of our work are summarized as followings.

- To the best of our knowledge, we are the first to investigate intrinsic and specific factors of nodes in HIN embedding and propose a disentangled representation solution to purify out intrinsic embedding. Purifying HIN embedding is important for HIN analysis, because it alleviates meta-path dependence and reduces noise interference.
- We devise a novel framework HEAD that disentangles intrinsic embedding from node embeddings generated by various meta-paths with adversarial disentangler, which boosts the robustness of representation. Besides, HEAD is an unsupervised learner to effectively absorb intrinsic factors from various meta-paths, thus enhancing existing meta-path based embedding methods.
- We perform experiments on four public datasets. Compared to state-of-the-art baselines, the maximum 15% improvement demonstrates the effectiveness of HEAD and the benefit of intrinsic representations.

2 Related Work

We review the most related works in network embedding, HIN embedding and representation disentanglement.

2.1 Network embedding

Network embedding, *i.e.*, network representation learning (NRL), is proposed to learn structure-preserving node representations which can be applied to downstream network tasks [4], [5], [21], [22].

Contemporary methods usually explore network topology as context information. DeepWalk [23] and node2vec [24] construct node sequences by randomly walking on the network, then utilize skip-gram based models to learn node embeddings. LINE [25] and SDNE [26] characterize first-order and second-order proximity to preserve neighborhood information for nodes. Furthermore, GraRep [27] and HOPE [28] are both designed to model the high-order proximity between nodes. Besides learning network embedding only from topology, there are also some works leveraging node content information [29], [30] or temporal information [31], [32], [33] for a more robust representation. Recently, graph neural networks [34], [35], [36] have demonstrated their remarkable ability in network representation learning, which learn a function that generates embeddings by aggregating feature from local neighborhood. For example, Graph Convolutional Networks [35] performs localized first-order approximation of spectral graph convolutions. Inspired by attention mechanism [37], [38], Graph Attention Networks [39] is proposed to distinguish the importance between neighbors and fuse them to obtain final node embeddings.

Nonetheless, these methods mainly deal with homogeneous networks, and they cannot be directly applied to HINs which contain multiple types of nodes and relations.

2.2 HIN embedding

In the past decade, heterogeneous information networks (HINs) [1] have been proposed to model complex entities and their rich relations in various applications [17], [40].

One major line of work leverages meta-path to learn semantic-preserving embedding for HINs. ESim [7] accepts

predefined meta-paths as guidance to learn node embedding for similarity search. Even though ESim utilize multiple meta-paths, it needs to conduct grid search to find the optimal weights of meta-paths. Metapath2vec [8] and HERec [16] design meta-path based random walk and utilize skip-gram to perform HIN embedding. HIN2Vec [41] carries out multi-task classification which learns the latent embeddings of nodes and meta-paths simultaneously. Based on the attention mechanism, HAN [18] proposes node- and semantic-level attention to learn the importance of nodes and meta-paths, respectively. In summary, these aforementioned meta-path based algorithms do not distinguish the intrinsic and specific factors of nodes when fusing multiple meta-paths, and may be susceptible to noise.

On another line, there exist some methods utilizing relation type rather than meta-path to perform HIN embedding. PME [42] projects different types of nodes into the same relation space and conducts heterogeneous link prediction. Based on the adversarial principle, HeGAN [43] designs relation-aware discriminator and generator to learn node distribution for better negative samples. HetGNN [44] introduces a random walk with restart strategy to sample a fixed size of neighbors, and aggregate feature information from them based upon node types. It is worth noting that meta-path has ability to accurately capture corresponding semantic, but these meta-path free methods may lose the advantage of meta-path, making the semantic fuzzy, which may harm the performance of specific relation prediction.

2.3 Representation disentanglement

Recently, representation disentanglement has gained considerable attention, particularly in the field of image representation learning [45], [46], [47], [48]. It aims to disentangle latent factors from image variants, which leads to the understanding of observed data. For instance, [49] learns invertible graphic codes for 3D image rendering in a fully supervised setting. InfoGAN [45] decomposes representation by maximizing the mutual information between latent factors and synthesized images. Such disentangled representations are demonstrated to bring enhanced generalization ability and robustness in downstream tasks [50], [51].

However, the complex network structure makes representation disentanglement on networks rarely explored. The only studies [52], [53] focus on homogeneous networks to identify the latent factor that may cause the link between the node pair. These methods ignore the heterogeneity of networks, and depart from our goal to disentangle intrinsic and specific factors for semantic-rich HINs.

3 Preliminary

In this section, we formalize the problem of HIN embedding and introduce the background on VAEs and GANs.

HIN embedding. Heterogenous information network is a special kind of information network, which contains either multiple types of nodes or relations. It can be formally defined as follows.

Definition 1. **Heterogeneous Information Network (HIN)** [1]. An HIN $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R}, \phi, \varphi)$ is a form of information network, where $\mathcal{V}$ and $\mathcal{E}$ denote sets of nodes and relations, respectively. It is also associated with a node type mapping function $\phi : \mathcal{V} \to \mathcal{A}$ and an relation type mapping function $\varphi : \mathcal{E} \to \mathcal{R}$, where $\mathcal{A}$ and $\mathcal{R}$ denote sets of node and relation types such that $|\mathcal{A}| + |\mathcal{R}| > 2$.

A toy example of HIN is illustrated in Figure 1 for bibliographic data. It is observed that it consists of three types of nodes (*i.e.*, author, paper and conference) and their semantic information based on meta-paths [19] (*e.g.*, author-paper-author).

The goal of HIN embedding is to embed each node $v \in \mathcal{V}$ to a low-dimensional space $\mathbb{R}^d$ ($d \ll |\mathcal{V}|$) while preserving structure and rich semantic on the HINs so that the learned embeddings can be applied to downstream tasks.

Variational Autoencoders. VAEs [54], [55] are deep generative models that simultaneously train both a probabilistic encoder and generator for a dataset $\mathcal{D} = \{\mathbf{x}^1, \cdots, \mathbf{x}^N\}$. The central analogy is that an encoding $\mathbf{z}$ can be considered as a latent variable, casting the generator as a conditional probability density $p_\theta(\mathbf{x}|\mathbf{z})$, where θ denotes parameters of the generator. By placing a weak prior $p(\mathbf{z})$ over $\mathbf{z}$, the generator defines a posterior and joint distribution $p(\mathbf{z}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$. And inference in VAEs can be performed using a variational method that approximates the posterior distribution $p(\mathbf{z}|\mathbf{x})$ using an encoder $q_\phi(\mathbf{z}|\mathbf{x})$, where ϕ denotes parameters of the encoder. The encoder and the generator are trained jointly by performing stochastic gradient ascent [54] on the *evidence lower bound* $\mathcal{L}(\phi, \theta; \mathcal{D})$ (ELBO),

$$\mathcal{L}(\phi, \theta; \mathcal{D}) = \sum_{n=1}^{N} \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^n)} \left[\log p_\theta\left(\mathbf{x}^n|\mathbf{z}\right) + \log p(\mathbf{z}) - \log q_\phi\left(\mathbf{z}|\mathbf{x}^n\right)\right].$$

Typically, the first term $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}^n)}\left[\log p_\theta\left(\mathbf{x}^n|\mathbf{z}\right)\right]$ is approximated by a Monte Carlo estimate and the remaining two terms are expressed as a divergence $-\mathrm{KL}(q_\phi\left(\mathbf{z}|\mathbf{x}^n\right) \| p(\mathbf{z}))$, which can be computed analytically when the encoder model and the prior are *Gaussian*.

Generative Adversarial Networks. Our model is based on adversarial learning, while the typical application of adversarial learning is GANs, which can be viewed as a minimax game between two players, *i.e.*, generator G and discriminator D, in the following manner.

$$\begin{aligned}\min_{\theta^G}\max_{\theta^D} \quad & \mathbb{E}_{x\sim P_{data}}\left[\log D(x;\theta^D)\right] \\ + \quad & \mathbb{E}_{z\sim P_Z}\left[\log\left(1 - D(G(z;\theta^G);\theta^D)\right)\right]\end{aligned}$$

The generator G tries to generate fake samples as close to true data as possible with the noise z from a predefined distribution P_Z, where θ^G denotes parameters of the generator. On the contrary, the discriminator D aims to distinguish real data from fake samples, where θ^D represents parameters of the discriminator. In practice, GANs has been found to work better if the generator minimizes $-\log D(G(\cdot;\theta^G);\theta^D)$ instead of $\log(1 - D(G(\cdot;\theta^G);\theta^D))$ [56].

4 HEAD: The Proposed Model

In this section, we present the proposed model HEAD, a novel HIN embedding framework with adversarial disentangler. We begin with the overview of HEAD. Subsequently we zoom into the meta-path disentangler, followed by elaborations on adversarial learners that guarantee disentanglement. Lastly, we discuss the optimization and characteristics

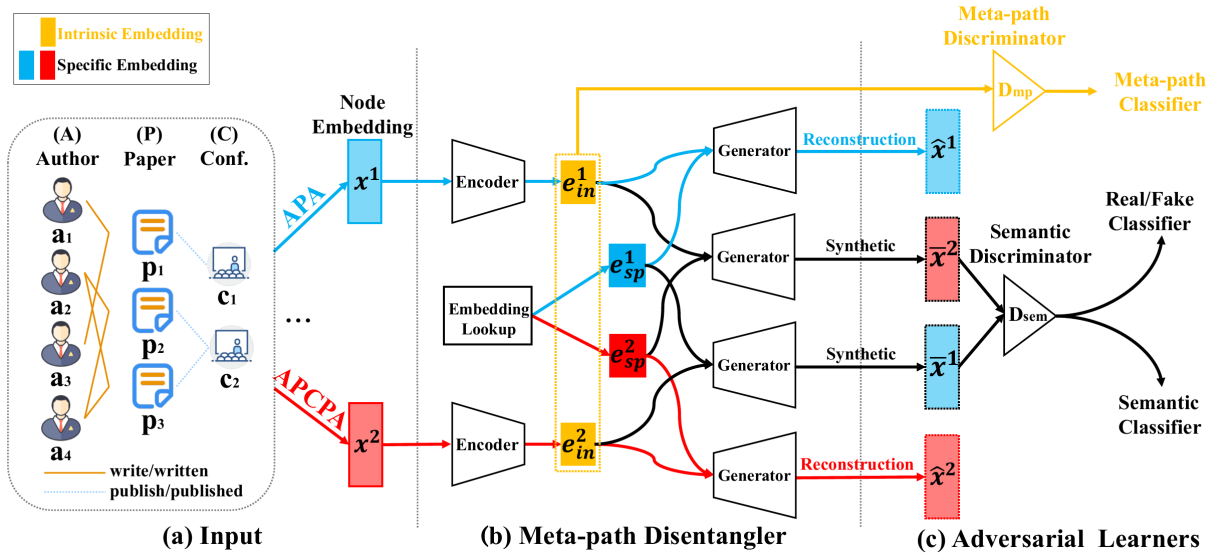


Fig. 2: Overview of our proposed model HEAD for disentangled representation learning on HINs, where takes two meta-paths APA and APCPA as an example. Note that the diagrams of encoder refer to the same one, as well as the generator.

TABLE 1: Summary of notations.

Notion	Explanation
D_{mp}, D_{sem}	meta-path and semantic discriminator, resp.
$\mathbf{x}^1, \hat{\mathbf{x}}^1, \bar{\mathbf{x}}^1$	input, reconstructed and synthesized node presentation of meta-path APA, resp.
$\mathbf{x}^2, \hat{\mathbf{x}}^2, \bar{\mathbf{x}}^2$	input, reconstructed and synthesized node presentation of meta-path APCPA, resp.
$\mathbf{e}^1_{in}, \mathbf{e}^1_{sp}$	intrinsic and specific embedding under APA, resp.
$\mathbf{e}^2_{in}, \mathbf{e}^2_{sp}$	intrinsic and specific embedding under APCPA, resp.
$\hat{l}_{mp}, \hat{l}_{sem}$	meta-path type predicted by meta-path classifier and semantic classifier, resp.
$\bar{l}_{mp}$	meta-path type of the randomly selected specific embedding
$\theta_E, \theta_{lkp}, \theta_G$	parameters of encoder, embedding lookup layer and generator, resp.
$\theta_{mp}, \theta_{sem}$	parameters of meta-path and semantic discriminator, resp.

of our model. Notations we use throughout the paper are summarized in Table 1.

4.1 Overview

In order to detach intrinsic and specific factors from multiple meta-paths, we propose a disentangled representation solution which firstly coarsely detaches intrinsic and specific embeddings under each meta-path with a meta-path disentangler, then refine them with adversarial learners to make intrinsic embedding more meta-path independent and specific embedding more meta-path dependent. Figure 2 shows the overall framework.

Given an HIN and a set of meta-paths, HEAD first obtains node embedding for each meta-path with a meta-path based embedding method (*e.g.*, Metapath2vec or HERec) as input. For the node embedding $\mathbf{x}^\phi$ of a meta-path ϕ, the meta-path disentangler employs a modified VAE architecture to detach the intrinsic embedding $\mathbf{e}^\phi_{in}$ and specific embedding $\mathbf{e}^\phi_{sp}$ from $\mathbf{x}^\phi$, which concatenates specific embedding $\mathbf{e}^\phi_{sp}$ in the middle layer of VAE to reconstruct the original node embedding $\mathbf{x}^\phi$. As exemplified in Figure 2, $\mathbf{x}^1$ is reconstructed from $\mathbf{e}^1_{in}||\mathbf{e}^1_{sp}$.

In order to refine the intrinsic and specific embedding, we design two adversarial learners. Specifically, a meta-path discriminator D_{mp} makes the intrinsic embedding more meta-path independent and a semantic discriminator D_{sem} makes the specific embedding more meta-path dependent. Since there is no label information to supervise the disentangled learning in our problem setting, we leverage meta-path schemes as self-supervised information to distinguish intrinsic and specific embedding. More precisely, to purify the intrinsic embedding $\mathbf{e}^\phi_{in}$, the meta-path discriminator D_{mp} makes $\mathbf{e}^\phi_{in}$ more difficult to be distinguished which meta-path it belongs to. To constrain specific embedding $\mathbf{e}^\phi_{sp}$ to be more meta-path dependent, the semantic discriminator D_{sem} is designed to distinguish a synthesized semantic, generating by the concatenation of intrinsic embedding and specific embedding under different meta-paths (*e.g.*, $\bar{\mathbf{x}}^2$ from $\mathbf{e}^1_{in}||\mathbf{e}^2_{sp}$). There are two classifiers inside semantic discriminator D_{sem}: (1) the semantic classifier forces the synthesized semantic belonging to the meta-path type of specific embedding (*e.g.*, $\bar{\mathbf{x}}^2$ belongs to APCPA), which makes the specific embedding more meta-path dependent. (2) The real/fake classifier guarantees the quality of synthesized semantic by adversarial learning, since bad synthesized semantic can be easily tested by the semantic classifier. Note that meta-path disentangler and adversarial learners are shared by all meta-paths. The parameters of these learners are iteratively training as other adversarial models.

4.2 Meta-path Disentangler

Given an HIN $\mathcal{G}$ and a series of meta-paths $(\phi^1, \phi^2, \cdots, \phi^K)$, we first learn node embeddings under each meta-path with existing meta-path based embedding methods (*e.g.*, Mp2vec and HERec) as input of HEAD. Note that we employ Mp2vec in the experiments to obtain the input node embeddings $(\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^K)$ for meta-path set $(\phi^1, \phi^2, \cdots, \phi^K)$.

Similar experimental results can also be obtained for other meta-path based HIN embedding methods (*e.g.*, HERec).

With node embeddings of various meta-paths as input, the meta-path disentangler coarsely detaches intrinsic and specific embedding for each meta-path. Specifically, taking the node embedding $\mathbf{x}^\phi$ of a meta-path ϕ as input, the encoder derives the intrinsic embedding $\mathbf{e}_{in}^\phi$ of $\mathbf{x}^\phi$, then concatenates it with the specific embedding $\mathbf{e}_{sp}^\phi$ from the embedding lookup layer to reconstruct the original node embedding $\mathbf{x}^\phi$ via generator.

It is worth noting that there exists uncertainty in the intrinsic embedding. For example, we cannot pinpoint the research interest of a broad-interest author. To model uncertainty, we are inspired by the development of variational autoencoders (VAEs) [54], [55], [57] to encode intrinsic embedding $\mathbf{e}_{in}^\phi$ into random variables with variance, rather than fixed values. Furthermore, we hypothesize that the intrinsic embeddings under various meta-paths have similar form inside the input embeddings and could be captured by the same encoder, but the specific embeddings are so diverse that they could be generated from the embedding lookup layer with random initialization. Thus we modify the VAE architecture as meta-path disentangler, which concatenates specific embedding $\mathbf{e}_{sp}^\phi$ from the middle layer, and the encoder and generator are implemented using multi-layer perceptron (MLP). The objective function of meta-path disentangler is defined as:

$$\mathcal{L}_{vae} = \|\hat{\mathbf{x}}^\phi - \mathbf{x}^\phi\|_F^2 + KL\left(q(\mathbf{e}_{in}^\phi|\mathbf{x}^\phi)\|p(\mathbf{e}_{in}^\phi)\right), \quad (1)$$

where the first term aims at recovering the original node embedding $\mathbf{x}^\phi$, and the second term calculates *Kullback-Leibler divergence* which penalizes the deviation of intrinsic embedding $\mathbf{e}_{in}^\phi$ from the prior distribution $p(\mathbf{e}_{in}^\phi)$ (as $\mathbf{e}_{in}^\phi \sim \mathcal{N}(0, \sigma)$). The loss function in Eq. (1) retains the reconstruction ability and aligns the latent intrinsic embedding to a Gaussian distribution. However, this property cannot guarantee that intrinsic embedding $\mathbf{e}_{in}^\phi$ is well disentangled from the specific embedding $\mathbf{e}_{sp}^\phi$.

4.3 Adversarial Learners

In order to better detach intrinsic and specific embeddings, we need to try the best to differentiate them with some guidance. However, there is no supervised information in our problem setting. Considering that embeddings are learned from diverse meta-paths, we utilize the meta-path type (denoted as l_{mp}) as self-supervised information to guide the learning. Moreover, enlightened by the adversarial learning [45], we design two adversarial learners to refine the disentanglement, including a meta-path discriminator purifying intrinsic embedding and a semantic discriminator specializing specific embedding.

4.3.1 Meta-path Discriminator

Intrinsic embedding does not depend on meta-paths and should be invariant to meta-path types. Thus to further refine the intrinsic embedding, we deploy meta-path discriminator D_{mp} in intrinsic space, which makes it difficult to tell which meta-path the intrinsic embedding comes from.

More precisely, the meta-path classifier inside D_{mp} takes intrinsic embedding $\mathbf{e}_{in}^\phi$ as input and predict the meta-path type $\hat{l}_{mp}$. In the adversarial setting, the encoder attempts to purify out intrinsic embedding $\mathbf{e}_{in}^\phi$ that invariant to meta-path type l_{mp}, whereas the meta-path discriminator tries to distinguish meta-path type l_{mp} from encoded intrinsic embedding $\mathbf{e}_{in}^\phi$. The better trained meta-path discriminator would then force the encoder to produce more intrinsic embedding, and the process is repeated. During such iterations, both the encoder and meta-path discriminator receive mutual, positive reinforcement. Therefore, the objective functions of meta-path discriminator $\mathcal{L}_{mp}$ and encoder $\mathcal{L}_E^{adv}$ are derived as follows:

$$\mathcal{L}_{mp} = \mathbb{H}\left[\log P\left(\hat{l}_{mp} = l_{mp}|\mathbf{e}_{in}^\phi\right)\right], \quad (2)$$

$$\mathcal{L}_E^{adv} = -\mathcal{L}_{mp} = -\mathbb{H}\left[\log P\left(\hat{l}_{mp} = l_{mp}|\mathbf{e}_{in}^\phi\right)\right], \quad (3)$$

where P is probability distribution over meta-paths, which is produced by meta-path classifier, l_{mp} is the ground-truth meta-path type of $\mathbf{e}_{in}^\phi$, and $\mathbb{H}$ represents the entropy.

4.3.2 Semantic Discriminator

To force specific embedding more meta-path dependent, we introduce semantic discriminator D_{sem}, including a semantic classifier and a real/fake classifier, to distinguish synthesized semantics generated by the concatenation of intrinsic embedding and specific embedding under different meta-paths.

Semantic Classifier. For meta-paths $(\phi^1, \phi^2, \cdots, \phi^K)$, we shuffle $(\mathbf{e}_{sp}^1, \mathbf{e}_{sp}^2, \cdots, \mathbf{e}_{sp}^K)$ to $(\mathbf{e}_{sp}^{i_1}, \mathbf{e}_{sp}^{i_2}, \cdots, \mathbf{e}_{sp}^{i_K})$, and concatenate them with intrinsic embeddings $(\mathbf{e}_{in}^1, \mathbf{e}_{in}^2, \cdots, \mathbf{e}_{in}^K)$, which forms the concatenation of each pair $\mathbf{e}_{in}^k||\mathbf{e}_{sp}^{i_k}$ to generate synthesized semantics. Although generated by intrinsic embedding and specific embedding under different meta-paths, a good synthesized semantic should be accurately distinguished as the meta-path type of specific embedding, because a good specific embedding is meta-path dependent while a good intrinsic embedding is meta-path independent. Thus the semantic classifier forces synthesized semantic belonging to the meta-path type of specific embedding. Without loss of generalization, we take two meta-paths in Figure 2 as an example to illustrate the semantic classifier. The synthesized semantic $\bar{\mathbf{x}}^2$ is generated from $\mathbf{e}_{in}^1||\mathbf{e}_{sp}^2$, and the semantic classifier forces it to be APCPA type, the objective function of which is derived as:

$$\begin{aligned}\mathcal{L}_{sem}^{clf} = &\mathbb{H}\left[\log P\left(\hat{l}_{sem} = l_{mp}|\mathbf{x}^2\right)\right] \\ &+ \mathbb{H}\left[\log P\left(\hat{l}_{sem} = \bar{l}_{mp}|\bar{\mathbf{x}}^2\right)\right], \end{aligned} \quad (4)$$

where $\bar{l}_{mp}$ is the meta-path type of specific embedding (*i.e.*, APCPA) and $\hat{l}_{sem}$ is the meta-path type prediction of semantic classifier.

Real/Fake Classifier. We introduce a real/fake classifier to distinguish real and synthesized semantics, which compels the generator to mimic the real semantic, so as to reduce the impact of bad synthesized semantic on classification. The objective functions $\mathcal{L}_{sem}^{r/f}$ and $\mathcal{L}_G^{adv}$ of classifier and generator are defined as:

$$\mathcal{L}_{sem}^{r/f} = \mathbb{H}\left[\log\left(C_{r/f}(\bar{\mathbf{x}}^2)\right)\right] + \mathbb{H}\left[\log\left(1 - C_{r/f}(\mathbf{x}^2)\right)\right], \quad (5)$$

$$\mathcal{L}_G^{adv} = -\mathbb{H}\left[\log\left(C_{r/f}(\bar{\mathbf{x}}^2)\right)\right], \quad (6)$$

Algorithm 1: Model training for HEAD

Input : HIN $\mathcal{G}$,
node embeddings $(\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^K)$ under meta-path set $(\phi^1, \phi^2, \cdots, \phi^K)$,
prior distribution $\mathcal{N}(0, \sigma)$
Output: intrinsic embeddings $(\mathbf{e}_{in}^1, \mathbf{e}_{in}^2, \cdots, \mathbf{e}_{in}^K)$,
specific embeddings $(\mathbf{e}_{sp}^1, \mathbf{e}_{sp}^2, \cdots, \mathbf{e}_{sp}^K)$

Initialize θ_E, θ_{lkp}, θ_G, θ_{mp} and θ_{sem};
while *not converge* **do**
  // **Meta-path discriminator training**
  Encode $(\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^K)$ into $(\mathbf{e}_{in}^1, \mathbf{e}_{in}^2, \cdots, \mathbf{e}_{in}^K)$;
  Update θ_{mp} with Eq. 7;
  // **Semantic discriminator training**
  Lookup $(\mathbf{e}_{sp}^1, \mathbf{e}_{sp}^2, \cdots, \mathbf{e}_{sp}^K)$;
  Shuffle $(\mathbf{e}_{sp}^1, \mathbf{e}_{sp}^2, \cdots, \mathbf{e}_{sp}^K)$ to $(\mathbf{e}_{sp}^{i_1}, \mathbf{e}_{sp}^{i_2}, \cdots, \mathbf{e}_{sp}^{i_K})$;
  Generate $(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \cdots, \bar{\mathbf{x}}^K)$ for each pair $\mathbf{e}_{in}^k || \mathbf{e}_{sp}^{i_k}$;
  Update θ_{sem} with Eq. 8;
  // **Meta-path disentangler training**
  Reconstruct $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \cdots, \hat{\mathbf{x}}^K)$;
  Update θ_E, θ_{lkp} and θ_G with Eq. 9, Eq. 10 and Eq. 11, respectively;
return $(\mathbf{e}_{in}^1, \mathbf{e}_{in}^2, \cdots, \mathbf{e}_{in}^K)$ and $(\mathbf{e}_{sp}^1, \mathbf{e}_{sp}^2, \cdots, \mathbf{e}_{sp}^K)$;

where $C_{r/f}()$ denotes the prediction result of real/fake classifier.

4.4 Optimization

Following the optimization technique in adversarial learning, we iteratively optimize three main modules to train HEAD. In each iteration, we alternately update meta-path discriminator D_{mp}, semantic discriminator D_{sem} and meta-path disentangler. For meta-path discriminator D_{mp}, it needs to accurately classify intrinsic embedding, so the gradient is:

$$\theta_{mp} \overset{+}{\leftarrow} -\Delta_{\theta_{mp}} \left(\mathcal{L}_{mp} \right). \tag{7}$$

With regard to semantic discriminator D_{sem}, its purpose is to identify the meta-path type of synthesized semantic, as well as the real/fake label:

$$\theta_{sem} \overset{+}{\leftarrow} -\Delta_{\theta_{sem}} \left(\mathcal{L}_{sem}^{clf} + \mathcal{L}_{sem}^{r/f} \right). \tag{8}$$

Meta-path disentangler includes encoder, embedding lookup layer and decoder. First of all, their common goal is to reconstruct input node embedding. Then, the encoder also makes intrinsic embedding difficult to distinguish the meta-path type, while the embedding lookup layer makes specific embedding richer with meta-path specific information and the generator ensures the quality of synthesized semantics, so their gradients are as follows:

$$\theta_E \overset{+}{\leftarrow} -\Delta_{\theta_E} \left(\mathcal{L}_{vae} + \mathcal{L}_E^{adv} \right), \tag{9}$$

$$\theta_{lkp} \overset{+}{\leftarrow} -\Delta_{\theta_E} \left(\mathcal{L}_{vae} + \mathcal{L}_{sem}^{clf} \right), \tag{10}$$

$$\theta_G \overset{+}{\leftarrow} -\Delta_{\theta_G} \left(\mathcal{L}_{vae} + \mathcal{L}_G^{adv} + \mathcal{L}_{sem}^{clf} \right), \tag{11}$$

where θ denotes the parameters of corresponding modules. The training framework of HEAD is outlined in Algorithm 1. Note that we concatenate the intrinsic embeddings of all meta-paths as the final intrinsic embedding for HEAD.

TABLE 2: Description of datasets.

Dataset	Relations (A-B)	#A	#B	#A-B
Yelp	Business-User	2, 614	1, 286	30, 838
	Business-Service	2, 614	2	2, 614
	Business-Reservation	2, 614	2	2, 614
ACM	Paper-Author	4, 019	7, 167	13, 407
	Paper-Subject	4, 019	60	4, 019
DBLP	Paper-Author	14, 376	14, 475	41, 794
	Paper-Conference	14, 376	20	14, 376
	Paper-Term	14, 376	8, 920	114, 624
AMiner	Paper-Author	28, 722	46, 583	78, 767
	Paper-Reference	28, 722	95, 970	250, 012

4.5 Discussion

We now conduct a complexity analysis. The complexity of HEAD lies in the iterative updating process. Each iteration mainly involves the updating of projection matrices in the meta-path disentangler and discriminators. It is worth noting that the meta-path disentangler and discriminators are shared by all meta-paths. Therefore, the time complexity is $O\left(K \cdot |\mathcal{V}| \cdot d^2\right)$, where K is the number of input meta-paths, $|\mathcal{V}|$ is the number of target-type nodes and d is the embedding dimension. Although HEAD adds complexity in iterative updating, this updating process is still efficient, which is linear with the number of target-type nodes.

Our HEAD provides a flexible framework to absorb multiple meta-paths for robust node embedding and can be seen as a fine-tuning process of existing meta-path based methods, which arms them with the ability to purify out intrinsic factors from each meta-path. Thus, HEAD is capable to enhance existing meta-path based methods.

5 Experiments

In this section, we evaluate the effectiveness of HEAD on four tasks, including node classification, node clustering, relation prediction and network visualization. We further analyze the underlying mechanism inside HEAD. Lastly, we investigate the hyper-parameter sensitivity.

5.1 Experimental Setup

5.1.1 Datasets

We conduct extensive experiments on four heterogeneous information networks, as summarized in Table 2.

- **Yelp** [43]. This dataset consists of four types of nodes, *i.e.*, business (B), user (U), service (S) and reservation (R), where businesses are divided into three classes. The meta-paths we use are $\{UBU, UBSBU, UBRBU\}$.
- **ACM** [18]. This dataset contains three types of nodes, *i.e.*, author (A), paper (P) and subject (S), where papers are divided into three classes according to the conferences they published on. Here we employ the meta-path set $\{PAP, PSP\}$ to perform experiments.
- **DBLP** [41]. This dataset involves four types of nodes, *i.e.*, author (A), paper (P), conference (C) and term (T),

TABLE 3: Quantitative results (%±σ) on node classification. (bold: best; underline: runner-up)

Datasets	Metrics	Training	DeepWalk	LINE-1st	LINE-2nd	ESim	HERec	Mp2vec	HIN2Vec	HetGNN	HeGAN	HEAD
Yelp	Macro-F1	20%	72.90±0.88	75.55±1.23	69.79±0.90	74.83±1.28	70.68±1.30	71.42±1.38	74.27±1.08	68.92±1.06	70.51±1.12	**75.69**±1.25
		40%	75.36±1.37	76.60±0.95	72.57±0.82	76.72±1.19	71.56±1.41	73.15±1.37	76.15±0.77	70.54±1.59	72.01±1.32	**78.16**±1.17*
		60%	76.47±1.69	77.60±0.82	73.57±1.04	77.86±1.41	72.94±1.72	74.36±1.32	77.17±1.21	71.76±1.55	73.41±1.44	**78.61**±1.50*
		80%	76.24±2.26	77.61±1.33	74.50±1.58	77.77±1.99	74.02±1.87	75.21±1.19	76.76±1.99	73.14±1.88	73.56±1.96	**79.23**±2.01*
	Micro-F1	20%	77.19±0.88	79.12±0.93	73.78±0.94	79.28±0.76	76.10±0.83	75.64±0.98	78.75±0.77	74.48±0.94	75.42±0.68	**79.62**±0.89
		40%	79.18±1.24	79.92±0.76	76.63±0.65	80.57±0.97	77.07±1.09	76.89±1.35	80.14±0.52	75.98±1.00	76.88±1.02	**80.92**±0.96*
		60%	80.03±1.51	80.64±0.84	77.67±0.73	81.36±1.22	77.95±1.54	78.16±1.42	80.89±1.15	77.22±1.25	78.19±1.16	**82.07**±1.20*
		80%	80.13±2.25	80.82±1.43	78.55±1.55	81.51±2.06	79.06±1.55	79.18±1.21	80.75±1.92	78.39±1.63	78.45±1.69	**83.31**±1.91*
ACM	Macro-F1	20%	79.32±0.64	80.34±0.74	80.62±0.52	78.94±0.69	81.45±0.81	82.94±0.38	82.09±0.62	79.97±0.60	80.36±0.63	**84.14**±0.67*
		40%	80.06±0.61	81.14±0.81	81.20±0.78	80.41±0.51	82.79±0.49	83.66±0.55	83.00±0.68	81.03±0.74	80.93±0.60	**85.94**±0.48*
		60%	80.50±0.87	81.69±0.76	81.48±0.94	80.76±0.87	83.07±1.02	84.04±0.68	82.61±0.73	81.70±0.82	81.02±0.80	**86.45**±0.59*
		80%	80.74±1.17	81.49±1.56	81.57±1.34	80.74±1.08	83.66±1.16	84.60±1.62	82.91±1.04	82.10±1.18	81.76±1.18	**87.03**±1.52*
	Micro-F1	20%	80.05±0.62	80.81±0.70	81.07±0.55	79.93±0.59	81.87±0.74	83.19±0.41	82.56±0.55	80.67±0.67	81.19±0.69	**84.41**±0.62*
		40%	80.81±0.63	81.48±0.82	81.75±0.83	81.20±0.49	83.12±0.48	83.83±0.58	83.38±0.64	81.70±0.73	81.66±0.56	**86.10**±0.50*
		60%	81.29±0.93	82.01±0.85	82.01±1.01	81.43±0.93	83.40±1.04	84.20±0.79	82.95±0.72	82.29±0.83	81.73±0.79	**86.59**±0.61*
		80%	81.60±1.14	81.82±1.61	82.20±1.36	81.39±1.14	84.00±1.14	84.79±1.50	83.26±0.98	82.79±1.15	82.55±1.20	**87.20**±1.47*
DBLP	Macro-F1	20%	90.99±0.30	89.50±0.35	90.24±0.43	92.65±0.36	90.86±0.27	92.55±0.43	84.63±1.90	93.11±0.22	93.42±0.28	**94.17**±0.23*
		40%	91.64±0.48	90.68±0.64	91.01±0.59	93.19±0.45	91.66±0.48	92.80±0.56	88.95±1.63	93.34±0.34	93.67±0.61	**94.55**±0.24*
		60%	92.22±0.64	91.39±0.58	91.58±0.60	93.71±0.44	92.33±0.50	92.80±0.52	91.13±0.65	93.45±0.42	93.76±0.67	**94.59**±0.46*
		80%	92.12±0.43	91.61±0.67	91.80±0.76	93.57±0.75	92.70±0.37	92.63±0.75	92.23±0.78	93.41±0.73	93.64±0.76	**94.92**±0.75*
	Micro-F1	20%	91.55±0.26	90.00±0.32	90.72±0.41	93.15±0.32	91.46±0.27	93.11±0.38	87.58±1.28	93.56±0.19	93.87±0.27	**94.56**±0.21*
		40%	92.16±0.44	91.13±0.61	91.46±0.56	93.63±0.40	92.21±0.45	93.31±0.52	90.52±1.21	93.77±0.30	94.11±0.56	**94.90**±0.21*
		60%	92.73±0.55	91.82±0.49	91.99±0.55	94.12±0.35	92.87±0.41	93.31±0.49	92.16±0.56	93.88±0.34	94.20±0.57	**94.96**±0.40*
		80%	92.71±0.45	92.04±0.54	92.22±0.66	94.05±0.67	93.24±0.34	93.18±0.73	93.07±0.68	93.90±0.61	94.14±0.69	**95.30**±0.65*
AMiner	Macro-F1	20%	57.46±0.60	63.11±0.35	53.78±0.31	58.41±0.33	59.33±0.36	61.63±0.39	62.78±0.31	61.86±0.55	62.00±0.67	**69.08**±0.67*
		40%	58.15±0.33	63.45±0.30	54.39±0.45	58.72±0.31	60.18±0.43	62.43±0.30	63.16±0.31	62.34±0.41	62.72±0.39	**71.46**±0.65*
		60%	58.23±0.35	63.61±0.46	54.43±0.50	58.90±0.48	60.33±0.41	62.70±0.30	63.19±0.41	62.46±0.39	63.18±0.68	**72.37**±0.80*
		80%	58.16±0.97	63.35±1.03	54.39±0.96	58.93±0.99	60.53±0.86	63.01±0.50	63.45±0.93	62.68±0.64	62.94±1.39	**72.61**±0.79*
	Micro-F1	20%	78.25±0.27	82.04±0.16	75.78±0.22	78.15±0.14	78.90±0.16	80.34±0.17	81.71±0.12	81.35±0.18	79.90±0.14	**83.30**±0.23*
		40%	78.69±0.20	82.24±0.17	76.22±0.32	78.37±0.20	79.60±0.21	81.13±0.19	82.01±0.13	81.56±0.18	80.29±0.17	**84.87**±0.29*
		60%	78.81±0.27	82.31±0.26	76.32±0.41	78.48±0.26	79.80±0.31	81.31±0.24	82.07±0.29	81.64±0.35	80.47±0.36	**85.44**±0.31*
		80%	78.96±0.41	82.41±0.59	76.48±0.82	78.63±0.48	80.12±0.63	81.74±0.47	82.30±0.75	81.95±0.52	80.64±0.59	**85.75**±0.47*

TABLE 4: Quantitative results (%±σ) on node clustering. (bold: best; underline: runner-up)

Datasets	Metrics	DeepWalk	LINE-1st	LINE-2nd	ESim	HERec	Mp2vec	HIN2Vec	HetGNN	HeGAN	HEAD
Yelp	NMI	34.73±1.32	38.66±0.00	38.66±0.00	35.12±0.06	39.01±0.00	36.66±0.43	36.14±0.30	37.76±0.00	37.31±0.32	**40.06**±0.00
	ARI	40.39±1.24	42.47±0.00	42.47±0.00	38.42±0.05	42.67±0.00	41.56±0.84	39.36±0.68	41.69±0.00	43.31±0.73	**44.02**±0.00
ACM	NMI	56.28±0.18	21.58±0.00	40.79±0.04	48.21±0.00	47.26±0.00	48.37±0.00	45.46±0.00	56.28±0.09	56.42±0.00	**56.64**±0.03
	ARI	43.22±0.09	15.45±0.00	38.66±0.03	35.45±0.00	36.39±0.00	35.60±0.00	34.30±0.00	43.45±0.10	**43.84**±0.00	43.51±0.02
DBLP	NMI	73.22±0.25	78.08±0.11	67.35±0.24	75.67±0.13	71.75±0.10	73.60±0.10	77.44±0.06	76.95±0.00	78.98±0.03	**81.39**±0.00
	ARI	78.45±0.26	82.01±0.08	72.87±0.34	81.09±0.17	77.72±0.08	77.73±0.07	82.26±0.06	82.52±0.00	83.84±0.03	**85.92**±0.00
AMiner	NMI	21.17±2.10	11.93±2.33	15.61±0.02	21.03±0.17	10.88±2.27	18.56±1.72	18.58±0.66	19.66±0.37	20.75±0.17	**21.42**±1.39
	ARI	14.61±4.46	4.81±4.69	2.26±0.02	14.20±0.23	11.42±1.89	2.70±1.90	-2.73±2.37	3.41±2.32	12.17±0.14	**14.63**±2.41

where authors are divided into four classes based on the conferences they submitted to. The meta-path set we utilize is $\{APA, APCPA, APTPA\}$.

- **AMiner** [58]. This dataset comprises three types of nodes, *i.e.*, author (A), paper (P) and reference (R), where papers are divided into six classes *w.r.t.* the conferences they published on. The used meta-paths are $\{PAP, PRP\}$.

Please note that for the AMiner dataset, we do not add conference nodes to the heterogeneous information network. Because once the PCP meta-path is used, the classification performances of the meta-path based HIN embedding methods *w.r.t. F1*, all approach 1, losing the meaning of comparison.

5.1.2 Baselines

We consider three categories of methods: homogeneous network embedding (DeepWalk, LINE), meta-path based (ESim, HERec, Mp2vec, HIN2Vec) and meta-path free (HetGNN, HeGAN) HIN embedding algorithms, summarized as follows.

- **DeepWalk** [23] performs truncated random walks, and employs the skip-gram model [20]. Here we ignore the heterogeneity of nodes and evaluate it on the whole HINs.
- **LINE** [25] exploits the first (**LINE-1st**) and second (**LINE-2nd**) order proximity in networks and preserves the local and global network structures meanwhile.
- **ESim** [7] captures semantic information from multiple meta-paths, and assigns weights to them. Here we conduct grid search to find the optimal weights for these meta-paths.
- **HERec** [16] designs a type-constrained strategy to filter node sequences of meta-path and utilizes skip-gram [20] to embed the heterogeneous information networks.
- **Mp2vec** [8] samples meta-path based random walks and employ skip-gram model [20] for semantic-preserving HIN embedding.
- **HIN2Vec** [41] converts HIN embedding into multi-task classification, then learns latent representations for both nodes and meta-paths jointly.
- **HetGNN** [44] utilizes random walk with restart to sample a fixed size of neighbors and group them according to their types, finally learns node embedding based on GNN framework.
- **HeGAN** [43] proposes a relation-aware discriminator and generator to embed HINs on the adversarial principle to

learn node distribution for better negative sampling.

Please note that we do not choose HAN [18] as baseline, because HAN is semi-supervised, while HEAD and other baselines are unsupervised methods, thus cannot be directly compared.

5.1.3 Implementation

We implement the proposed model HEAD with deep learning library PyTorch[1]. All experiments are conducted on a Linux server with GPU (GeForce GTX 1080 Ti) and CPU (Intel Xeon E5-2620), and its operating system is Ubuntu 16.04.1. The Python and PyTorch versions are 3.6.9 and 1.3.1, respectively.

We perform Adaptive Moment Estimation (Adam) [59] to optimize our model and apply a grid search for hyper-parameters: the learning rate is tuned in $\{0.05, 0.01, 0.005, 0.001\}$ and the dropout ratio is tuned amongst $\{0.1, 0.2, \cdots, 0.5\}$. Moreover, we utilize the two-layer MLP for all modules. To avoid gradients vanishing or exploding, we employ Batch Normalization [60] and set LeakyReLU [61] as the activation function. During each iteration of alternative training, we use a batch size of 32 and set the prior distribution to standard Gaussian. For the input of HEAD, we use node embeddings from Mp2vec for each meta-path.

The embedding dimension for all methods is set as 64. For random walk-based methods (*i.e.*, DeepWalk, HERec, HIN2Vec, Mp2vec and HetGNN), we set the number of walks per node to 10, the walk length to 100 and the window size to 5. For meta-path based methods, we select the same meta-paths as our model. Particularly, for ESim, we search the weights for all meta-paths; while for HERec and Mp2vec, we concatenate the embeddings of all meta-paths to ensure they utilize the same global information as ours. For LINE, we set the number of samples as 10000M. For fair comparison, HetGNN and HeGAN also use pre-trained embeddings from Mp2vec, which serve as node attributes for HetGNN meanwhile. In terms of other parameters, we follow the settings in their original papers.

5.2 Experimental Evaluation

5.2.1 Node Classification

In this task, we use different percentages of labeled nodes (20%, 40%, 60%, 80%) to train a logistic regression classifier whose input is the node embeddings learned by different models, and test the classifier on the remaining nodes. We evaluate the classification quality in terms of *Micro-F1* and *Macro-F1 w.r.t.* the node labels. Since the performance of logistic regression is affected by train/test split, we repeat 10 times, and report average and standard deviation results in Table 3. We use * to indicate that HEAD is significantly different from runner-up methods based on paired t-tests at the significance level of 0.01.

Based on results, we make the following observations.

- HEAD consistently and significantly outperforms all baselines and achieves performance gains over the best baseline by 2.2%, 2.9%, 1.4% and 14.4% on four datasets respectively, which demonstrates that disentangled intrinsic embeddings can characterize nodes in a robust manner. Note that the overwhelming performance superiority of HEAD over Mp2vec serving as the input of HEAD implies that HEAD is capable of enhancing existing meta-path based methods.
- Compared with the runner-ups, HEAD greatly improves the classification performance on the AMiner dataset. One feasible reason is that the used meta-paths, *i.e.*, PAP and PRP, are not informative. In this case, using the information of all meta-paths and purifying out intrinsic factors may reduce noise and learn robust representation.
- As the percentage of labeled nodes increases, our performance margins over the runner-ups become larger, since disentangled factors benefit more from the supervision, widening the gaps.
- Both HEAD and HeGAN employ the adversarial principle, thus their performance improvement illustrates the vital role of adversarial learning in enhancing representation robustness. Moreover, HEAD is consistently better than HeGAN, further indicating the effectiveness of disentangling the intrinsic factors.
- Among the baselines, HIN methods generally outperform homogeneous network methods (*i.e.*, DeepWalk and LINE), implying the importance of semantic-preserving embedding.

5.2.2 Node Clustering

In this task, we employ the K-Means algorithm on the embeddings produced by all algorithms to perform clustering, and evaluate the clustering quality in terms of *normalized mutual information* (*NMI*) and *adjusted rand index* (*ARI*) *w.r.t.* the node labels. Note that the larger *NMI* and *ARI* mean the better performance. Since the performance of K-Means is affected by initial centroids, we repeat the process for 10 times, and report average and standard deviation results in Table 4.

Similar conclusions to the node classification can be drawn, where HEAD generally shows better performance, which once again proves the effectiveness of intrinsic representations.

5.2.3 Relation Prediction

In this task, we predict the *co-author* relation of authors (*i.e.*, APA) on DBLP, and the *same-author* relation of papers (*i.e.*, PAP) on ACM and AMiner. Specifically, we randomly hide 20% of author-paper relations from the original network and label their corresponding relations as ground-truth positives, then randomly sample the same number of negative instances to form the test set together. Subsequently, we utilize the left network to learn node embeddings. After obtaining node embeddings, we train a logistic regression classifier using node embeddings of positive and negative instances from the left network. We adopt *F1*, *ACC* and *AUC* as evaluation metrics, shown in Table 5.

To investigate the impact of specific factors on the corresponding relation prediction, we evaluate the performances of the final intrinsic embedding and the concatenation of the final intrinsic embedding and corresponding meta-path specific embedding (*i.e.*, tested meta-paths: APA for DBLP, PAP for ACM and AMiner), denoted as HEAD and HEAD$_s$.

1. https://pytorch.org/

TABLE 5: Quantitative results (%) on relation prediction. (bold: best; underline: runner-up)

Datasets	Metrics	DeepWalk	LINE-1st	LINE-2nd	ESim	HERec	Mp2vec	HIN2Vec	HetGNN	HeGAN	HEAD	HEAD$_s$
ACM (Same-Author)	F1	58.36	64.88	61.37	59.35	64.96	63.92	60.84	59.40	59.80	65.60	**65.92**
	ACC	59.32	64.64	62.72	61.45	63.73	63.77	62.68	61.43	61.00	65.04	**65.74**
	AUC	63.39	66.34	66.08	64.91	65.74	67.24	67.01	65.57	65.26	67.56	**69.15**
DBLP (Co-Author)	F1	56.59	56.36	54.41	55.98	57.31	58.32	50.52	56.29	55.32	57.39	**59.19**
	ACC	56.57	57.12	58.27	57.67	57.55	57.60	57.91	56.97	56.90	58.28	**59.43**
	AUC	59.66	61.00	61.36	60.66	60.54	59.69	61.87	60.50	60.75	61.03	**62.00**
AMiner (Same-Author)	F1	62.19	59.11	56.57	58.09	62.39	60.54	62.06	62.38	**62.79**	61.48	62.47
	ACC	62.80	59.49	58.04	59.62	61.73	62.73	61.91	61.91	61.09	62.25	**62.96**
	AUC	65.57	63.31	61.13	64.19	62.03	65.92	66.84	65.78	63.79	64.46	**66.96**

(a) DeepWalk (b) Mp2vec (c) HeGAN (d) HEAD

Fig. 3: Visualization of the embeddings learned by baselines and HEAD on DBLP. Color indicates the categories of authors.

(a) Intrinsic embedding labeled by author categories (b) Specific embedding labeled by author categories (c) Specific embedding labeled by meta-path types (d) Performance change

Fig. 4: Disentanglement analysis of HEAD on DBLP.

We observe the following phenomena. (1) HEAD$_s$ and HEAD generally outperform all baselines, which reveals that detaching intrinsic and specific factors innately learns a better structure- and semantics-preserving embedding space for relations. (2) There exists performance gain of HEAD$_s$ over HEAD. Note that the task is to predict relations defined by given meta-paths, which is directly meta-path dependent. HEADs injects meta-path specific embeddings, which makes it perform better than HEAD. It implies that the learned specific embeddings are also useful for meta-path related tasks.

5.2.4 Network Visualization

To examine the network representation intuitively, we visualize embeddings of author nodes in DBLP using the t-SNE algorithm [62] in Figure 3.

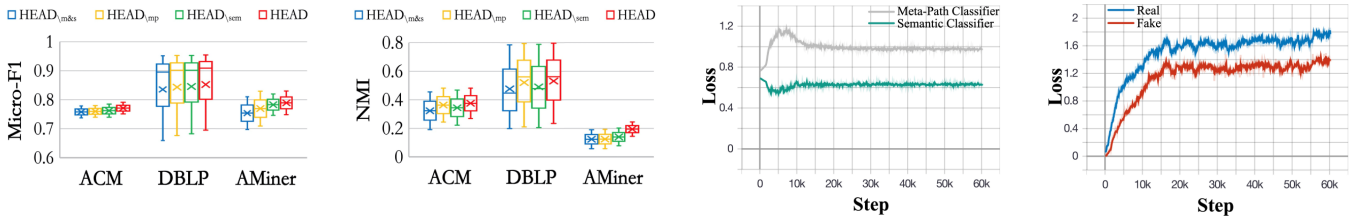
From the plots, we find that DeepWalk cannot effectively identify different author categories due to the ignorance of heterogeneity. On the other hand, Mp2vec and HeGAN can reasonably separate author categories, implying that they learn semantic-preserving embeddings to some degree. Compared to these baselines, HEAD correctly separate these authors with crisp boundaries, demonstrating that disentangling intrinsic factors from multiple meta-paths can exert a vital part in reducing noise.

5.3 Model Analysis

5.3.1 Disentanglement Analysis

First of all, we study whether HEAD can effectively disentangle intrinsic and specific factors of nodes. We use DBLP dataset as an example, and similar patterns have been observed on other datasets. In Figure 4, we visualize intrinsic and specific embedding under APCPA, subsequently present the performance changes of the final intrinsic embeddings and specific embeddings under all meta-paths on node clustering task.

In Figure 4 (a) and (b), the visualizations for intrinsic and specific embeddings are labeled by author categories which can be considered as intrinsic factors, and we find that the intrinsic embedding correctly separates nodes with clear boundaries, whereas the specific embedding is an extremely chaotic. We further label specific embeddings by meta-path types which can be treated as specific factors, and re-visualize them. As shown in Figure 4 (c), the visualization results have obvious clusters, conforming to our hypothesis that specific embedding reflects meta-path dependent semantics. From Figure 4 (d), we observe that the final intrinsic embedding converges quickly and performs well, while the performance of the specific embeddings fluctuates and slowly increases at a low level. The results are in



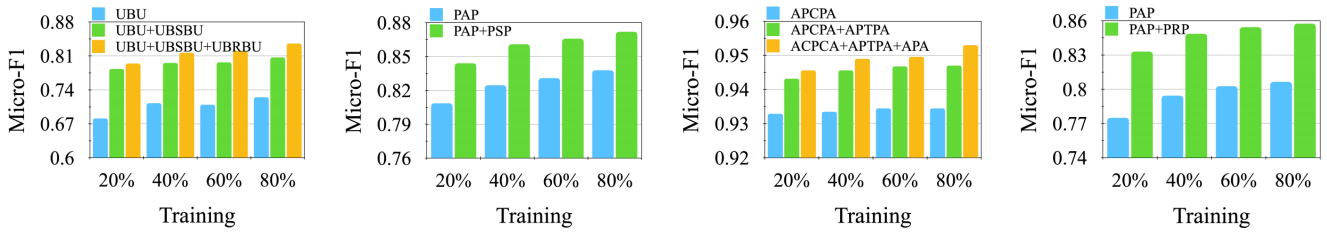
(a) Performances of HEAD and variants on node classification

(b) Performances of HEAD and variants on node clustering

(c) Loss change of meta-path and semantic classifier

(d) Loss change of real/fake classifier

Fig. 5: Analysis on adversarial learning of HEAD.



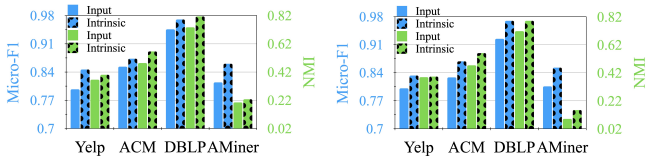
(a) Yelp

(b) ACM

(c) DBLP

(d) AMiner

Fig. 6: Performances of HEAD gradually adding paths.



(a) Mp2vec

(b) HERec

Fig. 7: Improvements to existing meta-path based methods.

line with our assumption that the intrinsic factors involved in various meta-paths reveal the intrinsic characteristic of nodes.

5.3.2 Adversarial Learning

Since adversarial learners in HEAD guarantee representation disentanglement, we conduct an ablation study to validate the effectiveness of them. In particular, we compare HEAD with three variants named HEAD$_{\backslash mp}$ (*i.e.*, HEAD without meta-path discriminator), HEAD$_{\backslash sem}$ (*i.e.*, without semantic discriminator) and HEAD$_{\backslash m\&s}$ (*i.e.*, without both discriminators). For HEAD and its variants, we separately evaluate intrinsic embeddings under every meta-path used in citation networks (*i.e.*, ACM, DBLP and AMiner datasets) on node clustering and classification tasks, and plot their performances of different meta-paths as boxes in Figure 5 (a) and (b).

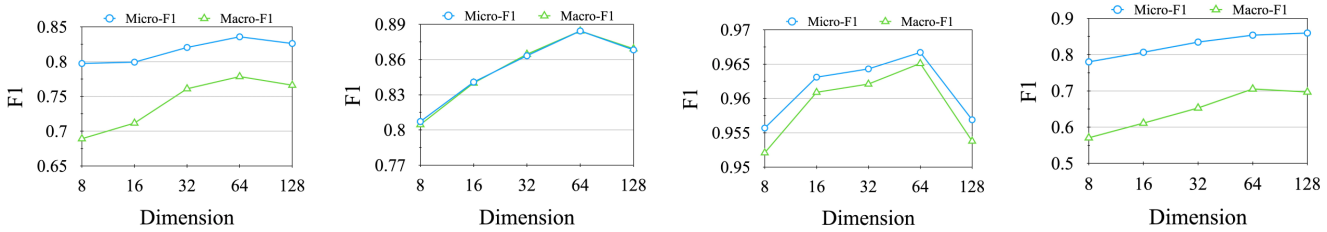
With respect to the average performance, we observe that HEAD is consistently better than the remaining variants and HEAD$_{\backslash m\&s}$ generally performs the worst. Such phenomenon is not surprising, and reveals that disregarding the adversarial learners leads to the intrinsic factors mottled, which harms the performance. In terms of the fluctuation range, HEAD is always minimal. We hypothesize that removing any module affects the mutual guidance of semantic purification among meta-paths, pushing the performances of various meta-paths away. Overall, these observations justify the necessity of adversarial learners.

Next, to understand the adversarial learning process of these two discriminators, we present the loss change of classifiers inside them on DBLP. As shown in Figure 5 (c), the loss of semantic classifier decreases, in the sense that the specific embedding becomes more meta-path dependent. Whereas the loss of meta-path classifier increases, the reason for which is the encoder tries to fool meta-path classifier by disentangling more intrinsic embedding. From Figure 5 (d), we observe the classification performances of real and fake semantics gradually become worse, indicating that the synthesized semantic is getting closer to the real one. After around 10k steps of adversarial training, the losses tend to converge. Note that the step here represents the mini-batch. In all, the experiment results reveal that these adversarial learners can stably and consistently improve performance through mutual adversarial learning.

5.3.3 Effects of Meta-paths

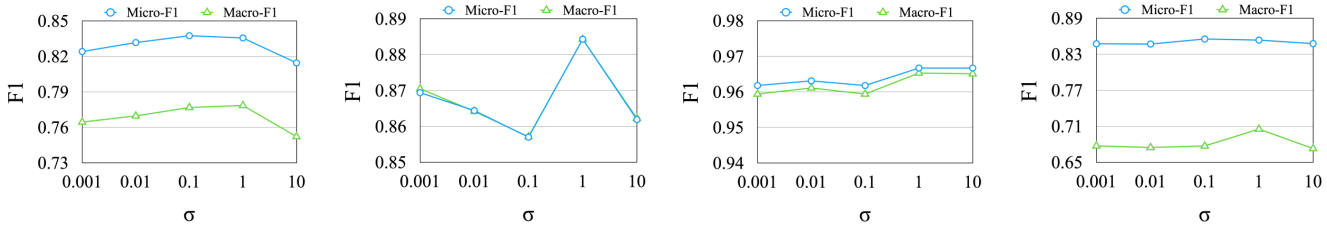
To investigate the effect of different meta-paths, we inject meta-paths to HEAD one by one from scratch, constantly concatenate intrinsic embedding under the new added meta-path and evaluate the resulted intrinsic embedding on node classification. We report their performances in Figure 6.

With the increase of injected meta-paths, the performance of HEAD consistently grows, which further confirms that HEAD can absorb intrinsic factors from diverse meta-paths for better performance. Note that, although many HIN based methods [16], [63] can improve performance when adding more meta-paths, they need supervised information. But the experiments here imply our HEAD is able to robustly improve performance through adding meta-paths, even though we have no supervised information.



(a) Yelp (b) ACM (c) DBLP (d) AMiner

Fig. 8: Impact of embedding dimension on HEAD.



(a) Yelp (b) ACM (c) DBLP (d) AMiner

Fig. 9: Impact of standard deviation on Gaussian prior.

5.3.4 Improvements to existing methods

As mentioned before, our HEAD can purify the learned embeddings of different meta-paths, which indicates that HEAD can effectively absorbs intrinsic factors from various meta-paths and enhance the existing meta-path based methods. Therefore, we design comparative experiments for different input methods to prove that point.

Here we employ two typical models (Mp2vec and HERec) as input methods. Specifically, for each dataset, we train the proposed model HEAD for 100 epochs with Mp2vec and HERec as input methods, respectively. Then the input embedding and disentangled intrinsic embedding are respectively evaluated by node classification and clustering task *w.r.t. Micro-F1* and *NMI* metrics. Please note that the input embedding and intrinsic embedding evaluated here are the concatenation embedding of all meta-paths in the dataset. As shown in Figure 7, the blue bars correspond to the *Micro-F1* on the left Y axis and the green bars correspond to the *NMI* on the right Y axis.

It is obvious that HEAD can significantly improve the performance of two base models, Mp2vec and HERec, on node classification and clustering tasks on all datasets. This demonstrates that HEAD, as an unsupervised learner, can utilize the mutual guidance among meta-paths to extract the intrinsic factors of nodes. Moreover, the performance improvements on node classification seem more significant than clustering, which further confirms that the disentangled factors benefit more from the supervision.

5.4 Hyper-parameter Sensitivity

Finally, we investigate the sensitivity of hyper-parameters and report the results *w.r.t.* node classification on four datasets. In the experiments, we vary the studied hyper-parameter and fix the others.

5.4.1 Dimension Selection

The embedding dimension is a key parameter to control the complexity and capacity. Therefore, we evaluate how it affects the classification performance. As shown in Figure 8, as we gradually increase the embedding dimension d, the performance grows since a larger dimension could enhance the representation capability. Nevertheless, when d is larger than the optimal value, increasing d will hurt the performance probably due to overfitting. Therefore, we employ the applicable embedding dimension 64 to balance the trade-off between performance and complexity.

5.4.2 Standard Deviation

The standard deviation of Gaussian prior reflects the distribution law that the intrinsic factors may follow. A small standard deviation σ will make the meta-path disentangler closer to vanilla auto-encoder, while a large σ gives the intrinsic factors more uncertainty. To investigate whether HEAD can benefit from modeling the uncertainty, we vary the standard deviation σ of Gaussian prior in the range of $\{0.001, 0.01, 0.1, 1, 10\}$, while keeping the other parameters the same. Figure 9 shows the experimental results on four datasets. HEAD achieves optimal performance at the standard deviation $\sigma = 1.0$ and is generally stable around it, and too small or large values would harm the model.

6 Conclusion

In this paper, we first investigate latent explanatory factors of nodes in HIN embedding and propose a novel solution HEAD with disentangled representation. Our HEAD purifies out intrinsic embedding from input embeddings of multiple meta-paths, which absorb informative characteristic from all meta-paths and reduce the noise interference. Specifically, given node embeddings of various meta-paths, a meta-path disentangler in HEAD encodes them

into intrinsic and specific spaces. Subsequently, two adversarial learners (*i.e.*, meta-path discriminator and semantic discriminator) utilize meta-path schemes as self-supervised information to further refine the disentanglement. Extensive experimental results verify the effectiveness of HEAD and its ability to attain robust node embedding.

7 Acknowledgments

This research is supported in part by the National Natural Science Foundation of China (No. U20B2045, 61772082, 62002029), The Fundamental Research Funds for the Central Universities (2021RC28), and BUPT Excellent Ph.D. Students Foundation (CX2021105). In addition, Y. Ye's work is partially supported by the NSF under grants IIS-2107172, IIS-2027127, IIS-2040144, CNS-2034470, IIS-1951504, CNS-1940859, CNS-1814825 and OAC-1940855, and the DoJ/NIJ 2018-75-CX-0032.

References

[1] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *Acm Sigkdd Explorations Newsletter*, pp. 20–28, 2013.

[2] C. Shi, Y. Li, J. Zhang, Y. Sun, and S. Y. Philip, "A survey of heterogeneous information network analysis," *IEEE Transactions on Knowledge and Data Engineering*, pp. 17–37, 2016.

[3] C. Shi and S. Y. Philip, *Heterogeneous information network analysis and applications*. Springer, 2017.

[4] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1616–1637, 2018.

[5] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Transactions on Knowledge and Data Engineering*, pp. 833–852, 2018.

[6] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *SIGKDD*, 2015, pp. 1165–1174.

[7] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *arXiv preprint arXiv:1610.09769*, 2016.

[8] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *SIGKDD*, 2017, pp. 135–144.

[9] L. Xu, X. Wei, J. Cao, and P. S. Yu, "Embedding of embedding (eoe): Joint embedding for coupled heterogeneous networks," in *WSDM*, 2017, pp. 741–749.

[10] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding." in *AAAI*, 2017, pp. 203–209.

[11] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han, "Heterogeneous network representation learning: Survey, benchmark, evaluation, and beyond," *arXiv preprint arXiv:2004.00216*, 2020.

[12] Y. Zhang, Y. Xiong, X. Kong, S. Li, J. Mi, and Y. Zhu, "Deep collective classification in heterogeneous information networks," in *WWW*, 2018, pp. 399–408.

[13] Y. Jacob, L. Denoyer, and P. Gallinari, "Learning latent representations of nodes for classifying in heterogeneous social networks," in *WSDM*, 2014, pp. 373–382.

[14] Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han, "Co-author relationship prediction in heterogeneous bibliographic networks," in *ASONAM*, 2011, pp. 121–128.

[15] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, and Q. Liu, "Shine: Signed heterogeneous information network embedding for sentiment link prediction," in *WSDM*, 2018, pp. 592–600.

[16] C. Shi, B. Hu, W. X. Zhao, and S. Y. Philip, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, pp. 357–370, 2018.

[17] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model," in *SIGKDD*, 2018, pp. 1531–1540.

[18] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, and P. S. Yu, "Heterogeneous graph attention network," in *WWW*, 2019, pp. 2022–2032.

[19] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, pp. 992–1003, 2011.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[21] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE transactions on Big Data*, 2018.

[22] D. Zhou, J. He, H. Yang, and W. Fan, "Sparc: Self-paced network representation for few-shot rare category characterization," in *KDD*, 2018, pp. 2807–2816.

[23] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *SIGKDD*, 2014, pp. 701–710.

[24] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *SIGKDD*, 2016, pp. 855–864.

[25] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*, 2015, pp. 1067–1077.

[26] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *SIGKDD*, 2016, pp. 1225–1234.

[27] S. Cao, W. Lu, and Q. Xu, "Grarep: Learning graph representations with global structural information," in *CIKM*, 2015, pp. 891–900.

[28] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *SIGKDD*, 2016, pp. 1105–1114.

[29] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *IJCAI*, 2015.

[30] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Homophily, structure, and content augmented network representation learning," in *ICDM*, 2016, pp. 609–618.

[31] D. Zhou, L. Zheng, J. Han, and J. He, "A data-driven graph generative model for temporal interaction networks," in *KDD*, 2020, pp. 401–411.

[32] Z. Liu, D. Zhou, Y. Zhu, J. Gu, and J. He, "Towards fine-grained temporal network representation via time-reinforced random walk," in *AAAI*, vol. 34, no. 04, 2020, pp. 4973–4980.

[33] Z. Liu, D. Zhou, and J. He, "Towards explainable representation of time-evolving graphs via spatial-temporal graph attention networks," in *CIKM*, 2019, pp. 2137–2140.

[34] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NIPS*, 2016, pp. 3844–3852.

[35] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *ICLR*, 2017.

[36] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017, pp. 1025–1035.

[37] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *ICLR*, 2014.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[39] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *ICLR*, 2018.

[40] B. Hu, Z. Zhang, C. Shi, J. Zhou, X. Li, and Y. Qi, "Cash-out user detection based on attributed heterogeneous information network with a hierarchical attention mechanism," in *AAAI*, 2019, pp. 946–953.

[41] T.-y. Fu, W.-C. Lee, and Z. Lei, "Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning," in *CIKM*, 2017, pp. 1797–1806.

[42] H. Chen, H. Yin, W. Wang, H. Wang, Q. V. H. Nguyen, and X. Li, "Pme: projected metric embedding on heterogeneous networks for link prediction," in *SIGKDD*, 2018, pp. 1177–1186.

[43] B. Hu, Y. Fang, and C. Shi, "Adversarial learning on heterogeneous information networks," in *SIGKDD*, 2019, pp. 120–129.

[44] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *SIGKDD*, 2019, pp. 793–803.

[45] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *NIPS*, 2016, pp. 2172–2180.

[46] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," *ICLR*, 2017.
[47] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier gans," in *ICML*, 2017, pp. 2642–2651.
[48] H. Kim and A. Mnih, "Disentangling by factorising," *arXiv preprint arXiv:1802.05983*, 2018.
[49] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, "Deep convolutional inverse graphics network," in *NIPS*, 2015, pp. 2539–2547.
[50] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, pp. 1798–1828, 2013.
[51] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, "Deep variational information bottleneck," *arXiv preprint arXiv:1612.00410*, 2016.
[52] J. Ma, P. Cui, K. Kuang, X. Wang, and W. Zhu, "Disentangled graph convolutional networks," in *ICML*, 2019, pp. 4212–4221.
[53] Y. Liu, X. Wang, S. Wu, and Z. Xiao, "Independence promoted graph disentangled networks," *arXiv preprint arXiv:1911.11430*, 2019.
[54] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082*, 2014.
[55] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.
[56] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396*, 2016.
[57] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *NIPS*, 2014, pp. 3581–3589.
[58] R. Bekkerman and A. McCallum, "Disambiguating web appearances of people in a social network," in *WWW*, 2005, pp. 463–470.
[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[60] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.
[61] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML*, 2013.
[62] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, pp. 2579–2605, 2008.
[63] C. Shi, Z. Zhang, P. Luo, P. S. Yu, Y. Yue, and B. Wu, "Semantic path based personalized recommendation on weighted heterogeneous information networks," in *CIKM*, 2015, pp. 453–462.

Ruijia Wang received the B.S. degree from Beijing University of Posts and Telecommunications in 2019. She is currently a Ph.D. student in Beijing University of Posts and Communications. Her current research interest is graph analysis, especially graph neural networks.

Chuan Shi received the B.S. degree from the Jilin University in 2001, the M.S. degree from the Wuhan University in 2004, and Ph.D. degree from the ICT of Chinese Academic of Sciences in 2007. He joined the Beijing University of Posts and Telecommunications as a lecturer in 2007, and is a professor and deputy director of Beijing Key Lab of Intelligent Telecommunications Software and Multimedia at present. His research interests are in data mining and machine learning. He has published more than 100 papers in refereed journals and conferences.

Tianyu Zhao received the B.S. degree from Beijing University of Posts and Telecommunications in 2020. He is currently a master student in Beijing University of Posts and Communications. His current research interest is graph analysis, especially heterogeneous graph neural networks.

Xiao Wang is an Assistant Professor in the School of Computer Science, Beijing University of Posts and Telecommunications. He received his Ph.D. degree from the School of Computer Science and Technology, Tianjin University, in 2016. He was a postdoctoral researcher in Department of Computer Science and Technology, Tsinghua University. He got the China Scholarship Council Fellowship in 2014 and visited Washington University as a joint training student from 2014 to 2015. His current research interests include data mining, social network analysis and machine learning. Until now, he has published more than 50 papers in conferences, such as AAAI, IJCAI, KDD, and journals such as IEEE TKDE, IEEE Trans. on Cybernetics, etc.

Yanfang Ye is Theodore and Dana Schroeder Associate Professor at Case Western Reserve University. She received her Ph.D. in computer science from Xiamen University. She was an assistant professor and then associate professor in the department of computer science and electrical engineering (CSEE) at West Virginia University. Her research areas mainly include data mining, machine learning, cybersecurity, and health intelligence.