Heterogeneous Graph Propagation Network

Houye Ji, Xiao Wang, Chuan Shi*, *Member, IEEE*, Bai Wang and Philip S. Yu, *Fellow, IEEE*

Abstract—Graph neural network (GNN), as a powerful graph representation technique based on deep learning, has shown superior performance and attracted considerable research interest. Recently, some works attempt to generalize GNN to heterogeneous graph which contains different types of nodes and links. Heterogeneous graph neural networks (HeteGNNs) usually follow two steps: aggregate neighbors via single meta-path and then aggregate rich semantics via multiple meta-paths. However, we discover an important *semantic confusion* phenomenon in HeteGNNs, i.e., with the growth of model depth, the learned node embeddings become indistinguishable, leading to the performance degradation of HeteGNNs. We explain semantic confusion by theoretically deriving that HeteGNNs and multiple meta-paths based random walk are essentially equivalent. Following the theoretical analysis, we propose a novel **H**eterogeneous graph **P**ropagation **N**etwork (HPN) to alleviate the semantic confusion. Specifically, the semantic propagation mechanism improves the node-level aggregating process via absorbing node's local semantic with a proper weight, which makes HPN capture the characteristics of each node and learn distinguishable node embedding with deeper HeteGNN architecture. Then, the semantic fusion mechanism is designed to learn the importance of meta-path and fuse them judiciously. Extensive experimental results show the superior performance of the proposed HPN over the state-of-the-arts.

Index Terms—Heterogeneous graph, graph neural network, representation learning, deep learning.

1 Introduction

Graph representation learning which can learn the representation of graph-structured data has been widely used in various graph mining tasks [1], [2], [3], [4], [5]. Among different graph representation learning methods, graph neural network (GNN) is one of the most competitive deep learning techniques and attracts considerable attention [6], [7], [8], [9]. Basically, the current GNNs follow the message-passing framework which receives messages from neighbors and applies neural network to learn node representations. The promising performance of GNNs has been demonstrated by various graph applications [1], [2], [8], [10].

However, previous GNNs mainly focus on homogeneous graph, while in reality, the real-world graph usually comes with multiple types of nodes or edges, which is widely known as heterogeneous information network or heterogeneous graph [11]. Heterogeneous graph contains rich semantics and has been widely used in modeling complex relational data. As shown in Figure 1, academic graph ACM contains three types of nodes including papers, authors and subjects and their complex relations. Meta-path [12], a composite relation connecting a sequence of nodes, has been widely used to capture the rich semantics (e.g., Paper-Author-Paper describes the co-author relation and Paper-Subject-Paper means two papers belong to the same subject). Obviously, nodes connected via different meta-paths show diverse similarities in multiple aspects which can be viewed as different semantic informations. Several heterogeneous graph neural networks (HeteGNNs) have been proposed to better analyze such heterogeneous graph [13], [14], [15]. HeteGNNs usually follow two step aggregating process in a hierarchical manner: aggregate neighbors via single meta-path in node-level and then aggregate rich semantics via multiple meta-paths in semantic-level. For example, HAN [13] leverages hierarchical attention for better aggregation. The highly practical value enables it widely used diverse applications, especially in industrial applications [14], [16], [17].

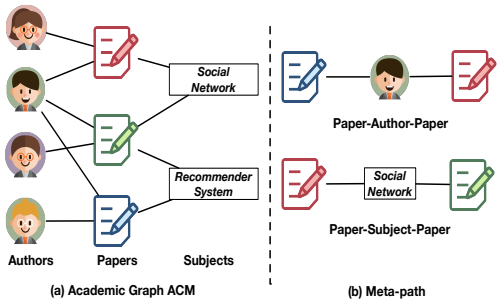
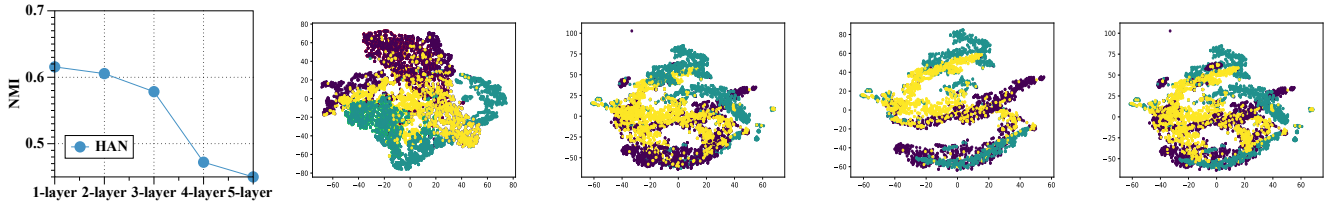


Fig. 1: An illustrative example of academic graph ACM. (a) A heterogenous graph ACM consists three types of nodes (i.e., author, paper, subject) and two types of connections. (c) Two meta-paths involved in ACM (i.e., Paper-Author-Paper and Paper-Subject-Paper).

When applying HeteGNNs in practice, we find an important phenomenon, called *semantic confusion*. Similar to over-smoothing in homogeneous GNNs [18], [19], semantic confusion means HeteGNNs inject confused semantics extracted via multiple meta-paths into node embedding, which makes the learned node embedding indistinguishable and leads to worse performance with more hidden layers. Figure 2(a) shows the clustering performance of HAN on ACM academic graph [13]. It clearly displays that with the growth of model depth, the performance of HeteGNNs is getting worse and worse. Furthermore, we visualize the learned paper embeddings via HAN in Figure 2(b)(c)(d)(e).

- *H. Ji, X. Wang, C. Shi (corresponding author) and B. Wang are with the Beijing Key Lab of Intelligent Telecommunications Software and Multimedia, Beijing University of Posts and Telecommunications, China.*
- *P. S. Yu is with University of Illinois at Chicago, IL, USA and Institute for Data Science, Tsinghua University, Beijing, China.*



(a) Cluster results of HAN with 1, 2, 3, 4, 5 layers. (b) Paper embedding via 1-layer HAN (c) Paper embedding via 2-layer HAN (d) Paper embedding via 3-layer HAN (e) Paper embedding via 4-layer HAN

Fig. 2: The clustering results and visualization of paper embeddings via HAN with different layers. Each point denotes one paper and corresponding color indicates the label (i.e., research areas). With the growth of model depth, semantic confusion happens which means the learned node embeddings become indistinguishable. For example, the paper embeddings belonging to different research areas which are learned via 1-layer HAN located in different positions, while the paper embeddings learned via 4-layer HAN mixed together.

As can be seen, HAN with 1-layer is able to learn distinguishable paper embedding, i.e., papers with different research areas are located in different positions, while HAN with 4-layer makes them less distinguishable. Note that neither overfitting nor vanishing gradient causes the degradation of HAN because we use LeakyReLU/ELU [20] as activation functions to alleviate the gradient vanishing and vary the hyper-parameters (e.g., regularization coefficient) with early-stopping to avoid overfitting. Actually, the reasons of semantic confusion are two folds: First, with the growth of model depth, different nodes will connect to the same meta-path based neighbors, which means the meta-path fails to capture the meaningful information for each node. Second, multiple meta-path combinations in semantic-level aggregating actually fuse multiple indistinguishable semantics, so the fused semantic remains indistinguishable. That is, the semantics extracted via multiple meta-paths are still indistinguishable. So even HeteGNNs are able to inject rich semantics into node embedding in a hierarchical manner, the confused semantics still happens, making the learned node embedding indistinguishable. Semantic confusion makes HeteGNNs hard to become a really deep model, which severely limits their representation capabilities and hurts the performance of downstream tasks. Alleviating the semantic confusion phenomenon to build a more powerful deeper HeteGNNs is an urgent problem.

In this paper, we theoretically analyze the semantic confusion in HeteGNNs and prove that HeteGNNs and multiple meta-paths based random walk [21] are essentially equivalent, which inspires us to alleviate the semantic confusion from the perspective of single meta-path aggregating in node-level or multiple meta-paths fusion in semantic-level. Then we propose a novel **H**eterogeneous Graph **P**ropagation **N**etwork (HPN) to alleviate semantic confusion from the perspective of multiple meta-paths based random walk, especially improving the single meta-path aggregating process in node-level. The proposed HPN contains two parts: semantic propagation mechanism and semantic fusion mechanism. Besides aggregating information from meta-path based neighbors, the semantic propagation mechanism also absorbs node's local semantics with a proper weight. So even with more hidden layers, semantic propagation mechanism can capture the characteristics of each node rather than inject confused semantics into node embedding. And thus it alleviates semantic confusion and builds a deeper HeteGNN. The semantic fusion mechanism aims to learn the importance of meta-paths and fuses them for comprehensive node embedding. Lastly, the whole model can be optimized via back-propagtation in an end-to-end manner.

The main contributions are summarized as follows:

- We first discover an important phenomenon in HeteGNNs, named semantic confusion. Semantic confusion means the growth of model depth indistinguishes the node embeddings learned via heterogeneous GNNs, leading to the degeneration of model performance.
- To the best of our knowledge, this is the first attempt to explain why semantic confusion happens by theoretically proving heterogeneous GNNs and multiple meta-paths based random walk are essentially equivalent.
- We propose a novel deep heterogeneous graph propagation network, called HPN, which mainly consists of the semantic propagation mechanism and the semantic fusion mechanism. Comparing to the previous HeteGNNs (e.g., HAN), the proposed HPN is able to alleviate the semantic confusion in node-level and learn more representative node embedding with more hidden layers. Specifically, semantic propagation mechanism of HPN absorbs node's local semantics and inject distinguishable semantics into node embedding in node-level aggregating, while semantic fusion mechanism of HPN learns the importance of meta-paths and fuse them properly .
- We conduct extensive experiments to evaluate the proposed HPN and show its superiority in comparison with the state-of-the-arts. We also show the characteristics of meta-paths in the analysis of the semantic propagation mechanism and semantic fusion mechanism. By analyzing the learned importance of different meta-paths, the proposed HPN demonstrates its potentially good interpretability for heterogeneous graph analysis.

2 Preliminary

Definition 1. ***Heterogeneous Graph*** *[22]. A heterogeneous graph, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consists of an object set $\mathcal{V}$ and a link set $\mathcal{E}$. A heterogeneous graph is also associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and a link type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{R}$. $\mathcal{A}$ and $\mathcal{R}$ denote the sets of predefined object types and link types, where $|\mathcal{A}| + |\mathcal{R}| > 2$.*

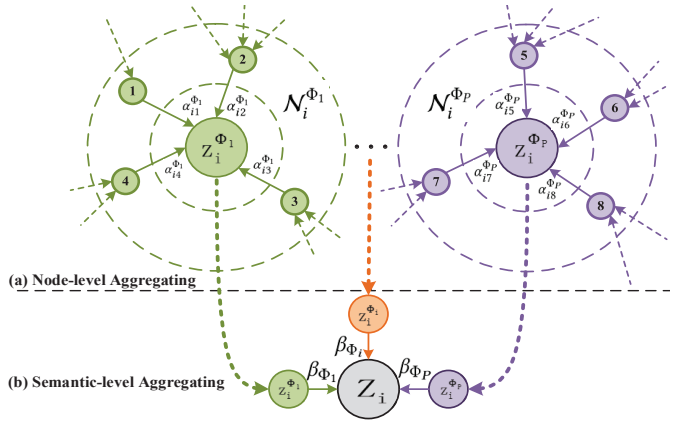


Fig. 3: Typical node- and semantic-level aggregating process in HeteGNNs, exemplified by HAN [13].

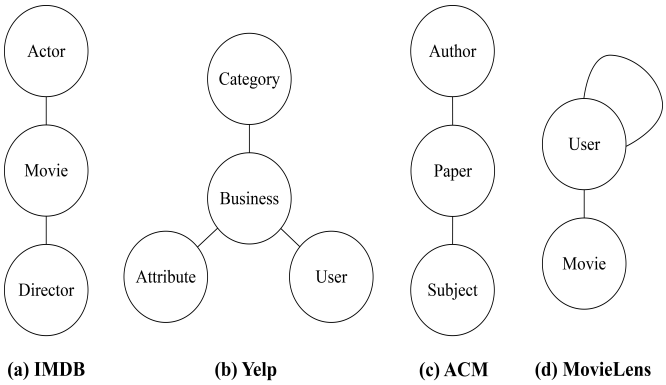


Fig. 4: Network schemas of heterogeneous graphs. (a) IMDB. (b) Yelp. (c) ACM. (d) MovieLens.

Definition 2. ***Meta-path*** *[12]. A meta-path Φ is defined as a path in the form of $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} A_{l+1}$ (abbreviated as $A_1A_2\cdots A_{l+1}$), which describes a composite relation $R = R_1 \circ R_2 \circ \cdots \circ R_l$ between objects A_1 and A_{l+1}, where $\circ$ denotes the composition operator on relations.*

Definition 3. ***Heterogeneous Graph Representation Learning*** *[15]. Given a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, heterogeneous graph representation learning aims to learn a mapping function (e.g., neural network) that projects the nodes $i \in \mathcal{V}$ into a d-dimensional space where $d \ll |V|$. The learned representation of node i, denoted as $\mathbf{z}_i \in \mathbb{R}^d$, is able to preserve rich heterogeneous information and can be applied to the downstram tasks.*

Definition 4. ***Semantic Confusion*** *Given a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a k-layer HeteGNN, if we take the limit $k \to \infty$, all node embedding $\mathbf{z}_i \in \mathbb{R}^d, i \in \mathcal{V}$ will converge to the same embedding $\mathbf{z}^{lim}$, which means the learned node embeddings become indistinguishable. We call it semantic confusion.*

$$\lim_{k\to\infty} \mathbf{z}_i = \mathbf{z}^{lim}, \forall\, i \in \mathcal{V}. \tag{1}$$

Example. Taking Figure 2 as an example, with the growth of k (e.g., 1,2,3,4,5), the nodes belonging to different classes mixed together. It means all node embeddings learned via HeteGNN (e.g., HAN) gradually become indistinguishable and the semantic confusion happens.

3 Semantic Confusion Analysis

In this section, we first give a brief review of HeteGNNs, and then prove that HeteGNNs and multiple meta-paths based random walk are essentially equivalent. Lastly, we explain why semantic confusion happens from the perspective of limit distribution of multiple meta-paths based random walk.

3.1 Heterogeneous Graph Neural Network

As shown in Figure 3, HeteGNNs (e.g., HAN) usually aggregate information from multiple meta-paths and update node embedding in both node-level and semantic-level. In particular, as shown in Figure 3(a), given a meta-path Φ_1 and node i, node-level attention in HAN aggregates the meta-path Φ_1 based neighbors $\{1,2,3,4\}$ with attentions $\{\alpha_{i1}^{\Phi_1}, \alpha_{i2}^{\Phi_1}, \alpha_{i3}^{\Phi_1}, \alpha_{i4}^{\Phi_1}\}$ to learn the semantic-specific node embedding $\mathbf{z}_i^{\Phi_1}$ for node i. Formally, given one meta-path Φ, the node-level aggregating is defined as:

$$\begin{aligned} &\mathbf{Z}^{\Phi,0} = \mathbf{X}, \\ &\mathbf{Z}^{\Phi,1} = \sigma\left(\boldsymbol{\alpha}^{\Phi,0} \cdot \mathbf{Z}^{\Phi,0}\right), \\ &\cdots, \\ &\mathbf{Z}^{\Phi} = \mathbf{Z}^{\Phi,k} = \sigma\left(\boldsymbol{\alpha}^{\Phi,k-1} \cdot \mathbf{Z}^{\Phi,k-1}\right), \end{aligned} \tag{2}$$

where $\mathbf{X}$ denotes node feature matrix, where the i-th row corresponds to the i-th node. And σ is an activate function, the element $\alpha_{ij}^{\Phi,k}$ of $\boldsymbol{\alpha}^{\Phi,k}$ denotes the learned attention weight between meta-path based node pair (i,j) via node-level attention by the k-th layer. Note that $\boldsymbol{\alpha}^{\Phi,k}$ is a (row-normalized) probability matrix and $\mathbf{Z}^{\Phi,k}$ denotes the learned embedding matrix by the k-th layer, where the i-th row corresponds to the i-th node. As shown in Figure 3(b), given a node i and a set of meta-paths $\{\Phi_1, \Phi_2, \cdots, \Phi_P\}$, semantic-level aggregating in HAN fuses P semantic-specific node embeddings $\left\{\mathbf{z}_i^{\Phi_1}, \cdots, \mathbf{z}_i^{\Phi_P}\right\}$ with attentions $\{\beta_{\Phi_1}, \cdots, \beta_{\Phi_P}\}$ to get the final embedding $\mathbf{z}_i$ for node i. The semantic-level aggregating is shown as follows:

$$\mathbf{Z} = \sum_{p=1}^{P} \beta_{\Phi_p} \cdot \mathbf{Z}^{\Phi_p}, \tag{3}$$

where $\mathbf{Z}$ denotes the final node embedding. In summary, HeteGNNs inject rich semantics into node embedding via aggregating process in both node- and semantic-level.

Here we take HAN as a representative HeteGNN to explain the hierarchical aggregating process and such aggregating process is also ubiquitous in HeteGNNs, such as [5], [14], [23], [24]. The difference among them is how to design different aggregating functions. So, we can analyze these HeteGNNs in a general framework (i.e., the hierarchical aggregating process in both node- and semantic-level) and give a unified explanation of semantic confusion phenomenon in HeteGNNs in the next section.

3.2 Relationship between HeteGNNs and Multiple Meta-paths based Random Walk

As a classical heterogeneous graph algorithm, multiple meta-paths based random walk [21] mainly contains: single meta-path based random walk and multiple meta-path combinations. Given a meta-path Φ, we have the meta-path

based probability matrix $\mathbf{M}^{\Phi}$ whose element $\mathbf{M}^{\Phi}_{ij}$ denotes the transition probability from node i to j via meta-path Φ. Then, the *k*-step single meta-path based random walk is defined as:

$$\boldsymbol{\pi}^{\Phi,k} = \mathbf{M}^{\Phi} \cdot \boldsymbol{\pi}^{\Phi,k-1}, \tag{4}$$

where $\boldsymbol{\pi}^{\Phi,k}$ denotes the distribution of *k*-step single meta-path based random walk. Considering a set of meta-paths $\{\Phi_1, \Phi_2, \cdots, \Phi_P\}$ and their weights $\{w_{\Phi_1}, w_{\Phi_2}, \cdots, w_{\Phi_P}\}$, the *k*-step multiple meta-paths based random walk is defined as:

$$\boldsymbol{\pi}^{k} = \sum_{p=1}^{P} w_{\Phi_p} \cdot \boldsymbol{\pi}^{\Phi_p,k}, \tag{5}$$

where $\boldsymbol{\pi}^{k}$ denotes the distribution of *k*-step multiple meta-paths based random walk. For *k*-step single meta-path based random walk:

Theorem 1. *Assuming a heterogeneous graph is aperiodic and irreducible, if we take the limit $k \to \infty$, then k-step meta-path based random walk will converge to a meta-path specific limit distribution $\boldsymbol{\pi}^{\Phi,\lim}$ which is independent of nodes:*

$$\boldsymbol{\pi}^{\Phi,\lim} = \mathbf{M}^{\Phi} \cdot \boldsymbol{\pi}^{\Phi,\lim}. \tag{6}$$

Proof. Since $\mathbf{M}^{\Phi}$ is a probability matrix, so the meta-path based random walk is a Markov chain. The convergence of Markov chain shows that $\boldsymbol{\pi}^{\Phi,k}$ will converge to a limit distribution $\boldsymbol{\pi}^{\Phi,\lim}$ if we take the limit $k \to \infty$. Obviously, $\boldsymbol{\pi}^{\Phi,\lim}$ only depends on $\mathbf{M}^{\Phi}$ and is independent of nodes. □

Different nodes connected via some relationships will influence each other and [19] demonstrates the influence distribution between two nodes is proportional to random walk distribution, shown as the following theorem:

Theorem 2 ([19])**.** *For the aggregation models (e.g., graph neural networks) on homogeneous graph, if the graph is aperiodic and irreducible, then the influence distribution I_i of node i is equivalent, in expectation, to the k-step random walk distribution.*

By Theorems 1 and 2, we conclude the influence distribution revealed by single meta-path based random walk is independent of nodes. Comparing Eq. 2 with Eq. 4, we find they both propagate and aggregate information via meta-path Φ. The difference is that $\boldsymbol{\alpha}^{\Phi,k}$ is a parameter matrix learned via node-level attention, while $\mathbf{M}^{\Phi}$ is a predefined matrix. Since $\mathbf{M}^{\Phi}$ and $\boldsymbol{\alpha}^{\Phi,k}$ are both probability matrix, they are actually meta-path related Markov Chain. So we find that node-level aggregation in HeteGNNs is essentially equivalent to meta-path based random walk if activate function is a linear function. Based on the above analysis, we find that if we stack infinite layers in node-level aggregating, the learned node embeddings $\mathbf{Z}^{\Phi}$ will only be influenced by the meta-path Φ and therefore are independent of nodes. So the learned node embeddings cannot capture the characteristics of each node and therefore are indistinguishable. For *k*-step multiple meta-paths based random walk, we have:

Theorem 3. *Assuming k-step single meta-path based random walk is independent of each other, if we take the limit $k \to \infty$, then the limit distribution of k-step multiple meta-paths based random walk is a weighted combination of single meta-path based random walk limit distribution, shown as follows:*

$$\boldsymbol{\pi}^{\lim} = \sum_{p=1}^{P} w_{\Phi_p} \cdot \boldsymbol{\pi}^{\Phi_p,\lim}. \tag{7}$$

Proof. Since *k*-step single meta-path based random walk is independent of each other, according to the properties of limits including Sum Rule and Constant Multiple Rule [25], we have:

$$\begin{aligned} \boldsymbol{\pi}^{\lim} &= \lim_{k\to\infty} \sum_{p=1}^{P} w_{\Phi_p} \cdot \boldsymbol{\pi}^{\Phi_p,k} = \sum_{p=1}^{P} w_{\Phi_p} \cdot \lim_{k\to\infty} \boldsymbol{\pi}^{\Phi_p,k} \\ &= \sum_{p=1}^{P} w_{\Phi_p} \cdot \boldsymbol{\pi}^{\Phi_p,\lim}. \end{aligned} \tag{8}$$

It shows that the meta-path combination can only change the position of limit distribution, but convergence of limit distribution remains unchanged. □

By Theorems 2 and 3, we conclude the influence distribution revealed by multiple meta-paths based random walk is also independent of nodes although they are connected via multiple meta-paths. Comparing Eq. 3 with Eq. 5, we can see that they both combine multiple meta-paths according to their weights. The difference is that semantic-level aggregating in HAN leverages neural network to learn the weight of meta-path β_{Φ_p}, while multiple meta-paths based random walk assigns predefined weight w_{Φ_p} to meta-path Φ_p by hand. Recall that in node-level aggregation, the node embeddings learned via single meta-path cannot capture the characteristics of each node and therefore are indistinguishable. In semantic-level aggregation, HeteGNNs fuse multiple node embeddings learned via multiple node-level aggregations with semantic-wise weights. Please semantic-wise weights are independent of each node. Synthesizing the above analysis, we conclude that the final node embeddings learned via both node- and semantic-level only influenced by a set of meta-paths and still remain indistinguishable. Since current HeteGNNs usually follow the hierarchical aggregation including both node- and semantic-level, we think it is the critical limitation of previous HeteGNNs and leads to semantic confusion. Based on the above analysis, to alleviate the semantic confusion phenomenon, we may improve the current HeteGNN architectures in node-level or semantic-level.

4 The Proposed Model

In this section, we propose a novel heterogeneous graph propagation network (HPN) which is able to alleviate the semantic confusion phenomenon in node-level based on the theoretical analysis. The proposed HPN mainly consists of semantic propagation mechanism and semantic fusion mechanism. Inspired by meta-path based random walk with restart, the proposed semantic propagation mechanism emphasizes node's local semantics in node-level aggregating process, alleviating the semantic confusion in node-level. Semantic fusion mechanism is able to learn the importance of meta-paths and get the optimal weighted combination of semantic-specific node embedding for the specific task.

4.1 Semantic Propagation Mechanism

Given one meta-path Φ, the semantic propagation mechanism $\mathcal{P}_\Phi$ first projects node into semantic space via semantic projection function f_Φ. Then, it aggregates information from meta-path based neighbors via semantic aggregation function g_Φ to learn semantic-specific node embedding, shown as follows:

$$\mathbf{Z}^\Phi = \mathcal{P}_\Phi(\mathbf{X}) = g_\Phi(f_\Phi(\mathbf{X})), \tag{9}$$

where $\mathbf{X}$ denotes initial feature matrix and $\mathbf{Z}^\Phi$ denotes semantic-specific node embedding. To handle heterogeneity graph, the semantic projection function f_Φ projects node into semantic space, shown as follows:

$$\mathbf{H}^\Phi = f_\Phi(\mathbf{X}) = \sigma(\mathbf{X}\cdot\mathbf{W}^\Phi + \mathbf{b}^\Phi), \tag{10}$$

where $\mathbf{H}^\Phi$ is the projected node feature matrix, $\mathbf{W}^\Phi$ and $\mathbf{b}^\Phi$ denote weight matrix and bias vector for meta-path Φ, respectively. Note that $\mathbf{H}^\Phi$ can also be viewed as the *0*-order node embedding $\mathbf{Z}^{\Phi,0}$, revealing the characteristics of each node. To alleviate semantic confusion, we design semantic aggregation function g_Φ, shown as follows:

$$\mathbf{Z}^{\Phi,k} = g_\Phi(\mathbf{Z}^{\Phi,k-1}) = (1-\gamma)\cdot\mathbf{M}^\Phi\cdot\mathbf{Z}^{\Phi,k-1} + \gamma\cdot\mathbf{H}^\Phi, \tag{11}$$

where $\mathbf{Z}^{\Phi,k}$ denotes node embedding learned by *k*-th layer semantic propagation mechanism and we take it as the semantic-specific node embedding $\mathbf{Z}^\Phi$. Note that $\mathbf{H}^\Phi$ reflects the characteristics of each node in meta-path Φ (also can be viewed as $\mathbf{Z}^{\Phi,0}$) and $\mathbf{M}^\Phi\cdot\mathbf{Z}^{\Phi,k-1}$ means aggregating information from meta-path based neighbors. Here γ is a weight scalar which indicates the importance of characteristic of node in aggregating process.

Why semantic aggregation function g_Φ works. Here we establish the relationship between semantic aggregation function g_Φ and *k*-step meta-path based random walk with restart. *k*-step meta-path based random walk with restart for node *i* is defined as:

$$\boldsymbol{\pi}^{\Phi,k}(\boldsymbol{i}) = (1-\gamma)\cdot\mathbf{M}^\Phi\cdot\boldsymbol{\pi}^{\Phi,k-1}(\boldsymbol{i}) + \gamma\cdot\boldsymbol{i}, \tag{12}$$

where $\boldsymbol{i}$ is a one-hot vector of node i, γ means the restart probability. For *k*-step meta-path based random walk with restart:

Theorem 4. *Assuming a heterogeneous graph is aperiodic and irreducible, if we take the limit $k\to\infty$, then* k*-step meta-path based random walk with restart will converge to $\boldsymbol{\pi}^{\Phi,\lim}(\boldsymbol{i})$ which is related to the start node i:*

$$\boldsymbol{\pi}^{\Phi,\lim}(\boldsymbol{i}) = \gamma\cdot(\mathbf{I}-(1-\gamma)\cdot\mathbf{M}^\Phi)^{-1}\cdot\boldsymbol{i}. \tag{13}$$

Proof. If we take the limit $k\to\infty$, we have:

$$\boldsymbol{\pi}^{\Phi,\lim}(\boldsymbol{i}) = (1-\gamma)\cdot\mathbf{M}^\Phi\cdot\boldsymbol{\pi}^{\Phi,\lim}(\boldsymbol{i}) + \gamma\cdot\boldsymbol{i}. \tag{14}$$

Solving Eq. 14, we have:

$$\boldsymbol{\pi}^{\Phi,\lim}(\boldsymbol{i}) = \gamma\cdot(\mathbf{I}-(1-\gamma)\cdot\mathbf{M}^\Phi)^{-1}\cdot\boldsymbol{i}. \tag{15}$$

Obviously, $\boldsymbol{\pi}^{\Phi,\lim}(\boldsymbol{i})$ is related to node i. □

By Theorems 2 and 4, we conclude that the influence distribution revealed by meta-path based random walk with restart is related to nodes. Comparing Eq. 11 to Eq. 12, we find they both emphasis node's local semantics with a proper weight γ. By Theorem 4, we can see that the semantic aggregation function g_Φ absorbs node's local semantics and makes semantic-specific node embedding $\mathbf{Z}^{\Phi,k}$ distinguish from each other even if we take the limit $k\to\infty$. So semantic propagation mechanism can alleviate the semantic confusion. So semantic propagation mechanism can alleviate the semantic confusion in node-level via a well-designed semantic aggregation function.

4.2 Semantic Fusion Mechanism

Generally, every node in a heterogeneous graph contains multiple types of semantic information and semantic-specific node embedding can only reflect node from one aspect. To describe node more comprehensively, we leverage multiple meta-paths to capture rich semantics and describe node from different aspects.

Given a set of meta-paths $\{\Phi_1,\Phi_2,\cdots,\Phi_P\}$, we have P group semantic-specific node embeddings $\{\mathbf{Z}^{\Phi_1},\mathbf{Z}^{\Phi_2},\cdots,\mathbf{Z}^{\Phi_P}\}$. Then, we propose the semantic fusion mechanism $\mathcal{F}$ to fuse them for the specific task. Taking P groups of semantic-specific node embeddings learned from semantic propagation mechanism as input, the final node embedding $\mathbf{Z}$ learned via semantic fusion mechanism $\mathcal{F}$, shown as follows:

$$\mathbf{Z} = \mathcal{F}(\mathbf{Z}^{\Phi_1},\mathbf{Z}^{\Phi_2},\cdots,\mathbf{Z}^{\Phi_P}). \tag{16}$$

Intuitively, not all meta-paths should be treated equally. So semantic fusion mechanism should be able to tell the difference of meta-paths and assign different weights to them. To learn the importance of meta-paths, we project each semantic-specific node embedding into the same latent space and adopt semantic fusion vector $\mathbf{q}$ to learn the importance of meta-paths. The importance of meta-path Φ_p, denoted as w_{Φ_p}, is defined as:

$$w_{\Phi_p} = \frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\mathbf{q}^{\mathrm{T}}\cdot\tanh(\mathbf{W}\cdot\mathbf{z}_i^{\Phi_p}+\mathbf{b}), \tag{17}$$

where $\mathbf{W}$ and $\mathbf{b}$ denote weight matrix and bias vector, respectively, which are shared for all meta-paths. Note that all parameters in semantic fusion mechanism are shared for all nodes and semantics. After obtaining the importance of meta-paths, we normalize them via softmax function to get the weight of each meta-path. The weight of meta-path Φ_p, denoted as β_{Φ_p}, is defined as:

$$\beta_{\Phi_p} = \frac{\exp(w_{\Phi_p})}{\sum_{p=1}^{P}\exp(w_{\Phi_p})}. \tag{18}$$

Obviously, the higher β_{Φ_p}, the more important meta-path Φ_p is. With the learned weights as coefficients, we can fuse P semantic-specific embeddings to obtain the final embedding $\mathbf{Z}$ as follows:

$$\mathbf{Z} = \sum_{p=1}^{P}\beta_{\Phi_p}\cdot\mathbf{Z}^{\Phi_p}. \tag{19}$$

Then we can optimize the whole model for the specific task and learn the final node embedding. Note that semantic fusion mechanism is quite flexible and be optimized for various types of tasks. For different tasks, each semantic may make different contribution which means β_{Φ_p} may change a lot. Note that multiple semantics fusion is a special form of multi-view learning [26], [27].

4.3 Loss Functions

For semi-supervised node classification, we calculate Cross-Entropy and update parameters in HPN:

$$\mathcal{L} = -\sum_{l \in \mathcal{Y}_L} \mathbf{Y}_l \cdot \ln(\mathbf{Z}_l \cdot \mathbf{C}), \tag{20}$$

where $\mathbf{C}$ is a projection matrix which projects the node embedding as a node label vector, $\mathcal{Y}_L$ is the set of labeled nodes, $\mathbf{Y}_l$ and $\mathbf{Z}_l$ are the label vector and embedding of the labeled node l, respectively.

For unsupervised node recommendation, we leverage BPR loss with negative sampling [28], [29] to update parameters in HPN:

$$\mathcal{L} = -\sum_{(u,v)\in\Omega} \log \sigma\left(\mathbf{z}_u^\top \mathbf{z}_v\right) - \sum_{(u,v')\in\Omega^-} \log \sigma\left(-\mathbf{z}_u^\top \mathbf{z}_{v'}\right), \tag{21}$$

where $(u, v) \in \Omega$ and $(u, v') \in \Omega^-$ denote the set of observed (positive) node pairs and the set of negative node pairs sampled from all unobserved node pairs, respectively.

4.4 Model Analysis

We first analyze the relationship between the proposed HPN and previous models (i.e., HAN and ResNet [30]). Both HAN and HPN are HeteGNNs, but they are still different as follows: (1) Different motivations. HAN aims to learn the importance of neighbors in aggregating process, while HPN aims to alleviate the semantic confusion and deepen the HeteGNNs. (2) Different architectures. HPN removes the time-consuming self-attention and designs a particular aggregating model which can emphasize the local semantics in aggregating process. If we set $\gamma = 0$ which means HPN cannot absorb the local semantic, then the semantic confusion will happen in both HAN and HPN. Although both ResNet [30] and HPN make neural network much deeper than before, they are still different as follows: (1) Different motivations. ResNet aims to avoid gradient vanish and make the learning process easier, while semantic propagation mechanism aims to alleviate the semantic confusion. (2) Different methods. Residual connection $F(x) + x$ connects two hidden layers, while semantic propagation mechanism emphasizes local semantic with weight γ.

The proposed HPN has no learnable parameters in aggregating process. So even if we stack semantic propagation mechanism for multiple layers, the number of total parameters in HPN remains unchanged, making HPN to be a space-efficient model with lower parameter complexity. Besides, the proposed HPN is also time-efficient and can be easily parallelized. Given a meta-path Φ, the time complexity of semantic propagation mechanism is $(V_\Phi S_\Phi + E_\Phi)$, where V_Φ is the number of nodes, E_Φ is the number of meta-path based node pairs and S_Φ is the size of hidden layer of semantic-specific transformation function. The overall complexity of HPN is linear to the number of nodes and meta-path based node pairs. The proposed HPN has potentionally good interpretability for the specific task which is a big advantage for heterogeneous graph analysis. Benefitting from the semantic fusion mechanism, the proposed HPN is able to learn the importance of meta-paths for the specific task. By analyzing the learned attention values of meta-paths, we can check which meta-paths make the higher (or lower) contributions in the specific task, which is also verified by the experimental results in Section 5.7.

5 Experiments

5.1 Datasets and Baselines

We conduct experiments on real-world heterogeneous graphs. The detailed descriptions are shown in Table 1.

•**Yelp**[1]**.** We extract businesses located in North Carolina (NC), Wisconsin (WI), Pennsylvania (PA) from Yelp. Then we construct a heterogeneous graph that comprises businesses (B), categories (C), attributes (A) and users (U). Business features actually are represented by their attributes. We employ the meta-path set {*BCB*, *BAB*, *BUB*} to perform experiments. Here we label the businesses based on their locations (i.e., their states).

•**ACM**[2]**.** We extract papers published in KDD, SIGMOD, SIGCOMM, MobiCOMM, and VLDB and divide the papers into three classes (*Database, Wireless Communication, Data Mining*). Then we construct a heterogeneous graph that comprises papers (P), authors (A) and subjects (S). Paper features correspond to elements of a bag-of-words represented of keywords. Note that we can adopt word2vec/doc2vec initialize paper features [15], [31] and further improve the learned node embedding. We employ the meta-path set {*PAP*, *PSP*} to perform experiments. Here we label the papers according to the conference they published.

•**IMDB**[3]**.** Here we extract a subset of IMDB which contains movies (M), actors (A) and directors (D). The movies are divided into three classes (*Action, Comedy, Drama*) according to their genre. Movie features correspond to elements of a bag-of-words represented of plots. We employ the meta-path set {*MAM*, *MDM*} to perform experiments.

•**MovieLens (ML for short)** [4]**.** Here we extract a subset of MovieLens which contains 1682 movies (M) and 943 users (U). Movie features correspond to their genres. We employ the meta-path set {*MU*, *UU*} to perform experiments.

We compare with some state-of-the-art baselines, including the (heterogeneous) network embedding and (heterogeneous) GNNs, to verify the effectiveness of the proposed HPN. Meanwhile, we also test two variants of HPN (i.e., HPN$_{pro}$ and HPN$_{fus}$) to verify the effectiveness of different parts in our model.

•metapath2vec (mp2vec for short) [32]/HERec [33]: Two classical heterogeneous graph embedding methods which perform meta-path based random walk and utilize skip-gram to embed the heterogeneous graphs.

•GCN [1]/GAT [34]/PPNP [35]: Three classical graph convolutional networks which are designed for the homogeneous graphs.

•MEIRec [14]: It is a heterogeneous graph neural network which is able to integrate rich semantics via multiple meta-paths.

1. https://www.yelp.com
2. http://dl.acm.org/
3. https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset
4. https://grouplens.org/datasets/movielens/

TABLE 1: Statistics of the datasets.

Dataset	A-B	#A	#B	#A-B	Fea.	Train	Valid	Test	Meta-paths/ Semantics
IMDB	Movie-Actor	4780	5841	14340	1232	300	300	2687	Movie-Actor-Movie (*MAM*) Two movies are played by the same actor.
	Movie-Director	4780	2269	4780					Movie-Director-Moive (*MDM*) Two movies,are directed by the same director.
ACM	Paper-Author	3025	5835	3025	1870	600	300	2125	Paper-Author-Paper (*PAP*) Two papers published,by the same author.
	Paper-Subject	3025	56	3025					Paper-Subject-Paper (*PSP*) Two papers belong to the same subject.
Yelp	Bus.-Category	4463	733	17123	144	600	300	3563	Business-Category-Business (*BCB*) Two business connected via the same category.
	Bus.-Attribute	4463	144	82705					Business-Attribute-Business (*BAB*) Two business have the same attribute.
	Bus.-User	4463	29383	44816					Business-User-Business (*BUB*) Two business connected via the same user.
ML	User-Movie	943	1682	100000	18	72000	8000	20000	User-Movie (*UM*) One user watch one movie.
	User-User	943	943	47150					User-User (*UU*) Two users are friends.

TABLE 2: Qantitative results (%) on the node clustering task. The larger values, the better performace.

Datasets	Metrics	mp2vec	HERec	GCN	GAT	PPNP	MEIRec	HAN	HGT	MG	HPN$_{pro}$	HPN$_{fus}$	HPN
Yelp	NMI	42.04	0.30	32.58	42.30	40.60	30.09	45.46	47.82	47.56	44.36	12.86	**48.90**
	ARI	38.27	0.41	23.30	41.52	37.72	27.88	41.39	42.91	43.24	42.57	10.54	**44.89**
ACM	NMI	21.22	40.70	51.40	57.29	61.68	61.56	61.56	60.89	64.12	65.60	67.55	**68.21**
	ARI	21.00	37.13	53.01	60.43	65.15	61.46	64.39	59.85	66.29	69.30	71.53	**72.33**
IMDB	NMI	1.20	1.20	5.45	8.45	10.20	11.32	10.87	11.59	11.79	9.45	12.01	**12.31**
	ARI	1.70	1.65	4.40	7.46	8.20	10.40	10.01	9.92	10.32	8.02	12.32	**12.55**

•HAN [13]: It is a heterogeneous graph neural network based solely on attention mechanism which employs node-level attention and semantic-level attention simultaneously.

•HGT [23]: It is a heterogeneous graph neural network which aggregates information via meta relation triplet based on heterogeneous mutual attention. We remove the relative temporal encoding in HGT, because our datasets are static heterogeneous graphs.

•MAGNN (MG for short) [24]: It is a heterogeneous graph neural network which leverages relational rotation encoder to aggregate semantic in complex space.

•HPN$_{pro}$: It is a variant of HPN, which set the restart probability $\gamma = 0$.

•HPN$_{fus}$: It is a variant of HPN, which takes the simple average over all meta-paths.

Here we randomly initialize parameters with the Gaussian distribution and leverage Adam [36] to optimize the model. For the proposed HPN, we set the learning rate to 0.01, the regularization parameter to 5e-3, the dimension of the semantic fusion attention vector $\mathbf{q}$ to 32 and the dropout rate to 0.5. We use early stopping with a patience of 100, i.e. we stop training if the validation loss does not decrease for 100 consecutive epochs. For GCN, GAT, PPNP, MEIRec and HAN, we optimize their parameters using the validation set. For single meta-path based methods including metapath2vec and HERec, we test their performance with different meta-paths and report the best performance of all meta-paths. For homogeneous graph neural networks including GCN, GAT and PPNP, we translate the original heterogeneous graph into several homogeneous graphs via different symmetric meta-paths and report the best performance of all meta-paths. For all GNNs including GCN, GAT, PPNP, HAN, HGT, MAGNN and MEIRec, we split exactly the same training set, validation set and test set to ensure fairness. Note that for all GNNs, we test their performance with different layers and report the best performance. The reason is that with the growth of model depth, the performance of some graph neural networks start to degenerate (e.g., HAN). For example, we use 1-layer HAN for all datasets (details are shown in Section 5.6). For random walk models(e.g., metapath2vec and HERec), we set window size to 5, walk length to 100, walks per node to 40, the number of negative samples to 5. For a fair comparison, we set the embedding dimension to 64 for all models. For node recommendation, we uniformly sample one negative sample to calculate BPR loss and set the drop rate to 0.5, the learning rate to 1e-5 and the regularization to 1e-5.

5.2 Node Clustering

In order to compare unsupervised models (i.e., metapath2vec and HERec) with semi-supervised models (i.e., GCN, GAT, PPNP, MEIRec, HGT, MAGNN, HAN and HPN), following the previous work [13], we get the learned node embeddings of all models via feed forward, and then leverage classical node clustering to test their effectivenesses. Here we utilize the K-Means to perform node clustering and the number of clusters K is set to the number of classes. We select *NMI* and *ARI* to evaluate the clustering task and report the averaged results of 10 runs in Table 2.

As can be seen, the proposed HPN performs significantly better than all baselines. It shows the importance of alleviating semantic confusion in HeteGNNs. We also find that graph neural networks always perform better than network embedding methods. Moreover, heterogeneous graph neural networks including HAN, MEIRec, HGT, MAGNN and HPN outperform homogeneous GNNs because they can capture rich semantics and describe the characteristic of node more comprehensively. Note that the performance of HPN$_{pro}$ and HPN$_{fus}$ both show different degradations,

which imply the importance of semantic propagation mechanism and semantic fusion mechanism. Based on the above analysis, we can find that the proposed HPN can propagate and fuse semantic information effectively and shows significant improvements.

5.3 Node Classification

Besides node clustering, node classification is also an effective method to evaluate the node embeddings. Once the graph neural network trained, we can get all the node embedding via feed forward. Here we employ the KNN classifier ($k = 5$) perform node classification and select *Macro-F1* and *Micro-F1* as the evaluation metrics. For a more stable comparison, we repeat the process for 10 times and report the averaged results in Table 3.

As shown in Table 3, the proposed HPN generally outperforms than baselines. As can be seen, graph neural network based methods which combine the structure and feature information, usually perform better than graph embedding methods (i.e., metapath2vec and HERec) which do not consider the feature information. With deeper insight into these models, attention based models (i.e., GAT, HAN, HGT, MAGNN and HPN) which are able to learn the weights of nodes or meta-paths usually perform better. Although MEIRec, HGT, MAGNN, HAN and HPN are designed for heterogeneous graph, HPN still show its superiority. The reason is that HPN can capture high-order semantics via stacking more layers and absorbs node's local semantics. Comparing with HPN, the performance of two variants of HPN (i.e., HPN$_{pro}$ and HPN$_{fus}$) show different degradations, indicating the importance of semantic propagation mechanism and semantic fusion mechanism. In summary, HPN usually outperforms all state-of-the-art baselines.

5.4 Node Recommendation

Following [24], [29], we test all models on unsupervised node recommendation task. Here we select four metrics including precise@10, recall@10, hit@10 and ndcg@10 to evaluate the recommendation results. The experimental results are shown in Table 4.

From Table 4, we can find the proposed HPN outperforms all baselines. Heterogeneous GNNs show their superiorities via capturing rich semantics. Attention based GNNs usually perform better via distinguishing the importance of nodes or meta-paths. By stacking multiple layers, HPN is capable of exploring the high-order semantics, which serves the crucial collaborative signal in the recommendation. Note that the performance of HPN$_{fus}$ is similar to HPN because all meta-paths make equal contributions.

5.5 Node Visualization

For a more intuitively comparison, we conduct the task of node visualization. Specifically, we use t-SNE [37] to project the learned node embedding into a 2-dimensional space and visualize them. Taking ACM as an example, we visualize the learned paper embedding in Figure 5. Each point indicates one paper and its color indicates the research area.

From Figure 5, we can observe that homogeneous GNNs fail to perform well. The papers belonging to different research areas are mixed with each other. Comparing with homogeneous GNNs, HAN performs slightly better. However, the boundary is still blurry. Based on the visualization results, we can find that the proposed HPN is able to learn more representative node embedding than all baselines, as papers belonging to different research areas located in different positions with clear boundaries. Benefitting from semantic propagation mechanism, the proposed HPN is able to caputure high-order semantic structure and learn more representative node embedding.

5.6 Robustness to Model Depth

A salient property of HPN is the incorporation of the semantic propagation mechanism which is able to alleivate the semantic confusion and build a deeper and more powerful HeteGNN. Comparing to the previous HeteGNNs (e.g., HAN), the proposed HPN can stack more layers and learn more representative node embedding. To show the superiority of semantic propagation in HPN, we test HAN and HPN with 1, 2, 3, 4, 5 layers, shown in Figure 6.

As can be seen, with the growth of model depth, the performance of HAN performs worse and worse on both ACM and IMDB. Recall the theoretical analysis in Section 2, we believe this phenomenon is the semantic confusion, leading to the degradation of previous heterogeneous GNNs (e.g., HAN). Obviously, semantic confusion makes HeteGNNs hard to become a really deep model, which severely limits their representation capabilities and hurts the performance of downstream tasks (e.g., node clustering). On the other hand, with the growth of model depth, the performance of the proposed HPN is getting better and better, indicating that semantic propagation mechanism is able to effectively alleviate the semantic confusion. So even stacking for more layers, the node embeddings learned via the proposed HPN are still distinguishable. In summary, the proposed HPN is able to capture high-order semantics and learns more representative node embedding with deeper architecture, rather than learning indistinguishable node embedding.

5.7 Analysis of Semantic Fusion Mechanism

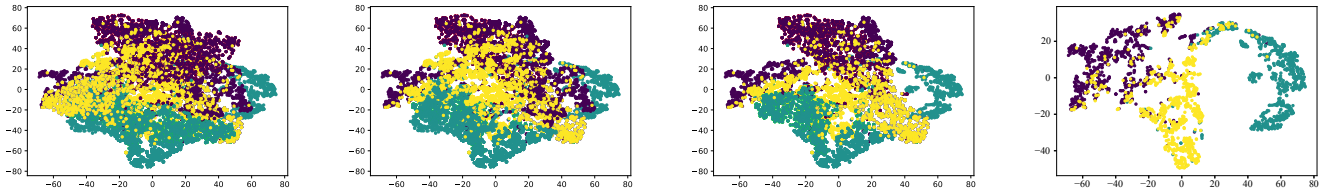
An interesting characteristic of HPN is the incorporation of semantic fusion mechanism which can learn the importance of meta-paths and fuse them for the specific task. To understand the importance of meta-paths, we provide a detailed analysis on the semantic fusion mechanism. Here we conduct two types of semantic fusion experiments, shown in Figure 7 and Figure 8. We first show the importance of meta-paths with the fixed number of layers. Then, we discover the propagation property of meta-path by showing how the importance of meta-paths changes with the growth of depth.

5.7.1 Semantic Fusion Mechanism with Fixed Depth.

Here we analyze semantic fusion mechanism on ACM dataset with 5-layer HPN. As shown in Figure 7, different meta-paths show different effectivenesses and semantic fusion mechanism assigns different attention weights to them. The higher *NMI*, the more useful meta-path is. The proposed HPN assigns higher weights to more userful meta-paths which means the semantic fusion mechanism is able to fuse multiple meta-paths properly for the specific task. In

TABLE 3: Qantitative results (%) on the node classification task. The larger values, the better performace.

Data	Metric	Train	mp2vec	HERec	GCN	GAT	PPNP	MEIRec	HAN	HGT	MG	HPN_{pro}	HPN_{fus}	HPN
Yelp	Ma-F1	20%	82.37	83.03	83.57	75.10	78.73	79.09	83.91	82.95	84.05	79.39	66.28	**84.47**
		40%	83.93	83.93	84.29	77.83	81.79	79.94	84.23	83.23	84.51	82.39	68.44	**85.56**
		60%	83.69	83.76	84.32	78.24	83.37	80.31	84.83	83.89	84.58	83.12	69.80	**86.55**
		80%	84.17	83.28	84.84	79.58	83.53	79.58	84.50	83.81	84.66	83.76	70.66	**85.75**
	Mi-F1	20%	82.43	83.08	83.61	75.37	79.10	79.20	83.89	82.94	84.10	79.84	66.70	**84.57**
		40%	83.98	83.97	84.3	77.88	81.88	79.95	84.23	83.26	84.49	82.54	68.61	**85.64**
		60%	83.74	83.79	84.33	78.43	83.43	80.39	84.78	83.81	84.59	83.34	69.92	**86.61**
		80%	84.25	83.38	84.81	79.52	83.56	79.68	84.87	83.79	84.67	83.77	71.16	**85.90**
ACM	Ma-F1	20%	65.09	66.17	86.81	86.23	88.62	88.89	89.40	89.12	89.01	88.28	89.38	**89.71**
		40%	69.93	70.89	87.68	87.04	88.99	89.12	89.79	**89.80**	89.76	88.80	89.59	89.68
		60%	71.47	72.38	88.10	87.56	89.58	89.61	89.51	89.58	89.81	88.86	89.75	**89.96**
		80%	73.81	73.92	88.29	87.33	90.09	89.84	90.63	90.31	90.85	89.86	90.12	**91.16**
	Mi-F1	20%	65.00	66.03	86.77	86.01	88.55	88.72	89.22	88.98	88.84	88.16	89.28	**89.58**
		40%	69.75	70.73	87.64	86.79	88.91	88.89	**89.64**	89.63	89.05	88.69	89.47	89.57
		60%	71.29	72.24	88.12	87.40	89.52	89.40	89.33	89.45	88.55	88.72	89.56	**89.78**
		80%	73.69	73.84	88.35	87.11	90.07	89.72	90.54	90.69	89.89	89.72	90.09	**91.11**
IMDB	Ma-F1	20%	41.16	41.65	45.73	49.44	48.73	50.12	50.00	51.13	**51.98**	48.96	50.25	51.36
		40%	44.22	43.86	48.01	50.64	49.93	52.36	52.71	52.07	52.55	50.61	52.63	**52.86**
		60%	45.11	46.27	49.15	51.90	51.18	53.21	**54.24**	54.19	54.11	52.78	53.31	54.15
		80%	45.15	47.64	51.81	52.99	53.98	53.68	54.38	54.68	54.59	53.30	55.71	**55.78**
	Mi-F1	20%	45.65	45.81	49.78	55.28	53.82	55.97	55.73	55.67	55.98	53.63	**56.80**	56.53
		40%	48.24	47.59	51.71	55.91	54.79	57.20	57.97	57.99	57.89	55.12	57.83	**58.25**
		60%	49.09	49.88	52.29	56.44	55.42	57.73	58.32	58.21	58.12	56.79	**58.85**	58.52
		80%	48.81	50.99	54.61	56.97	58.10	58.23	58.51	58.59	58.55	57.00	59.93	**60.07**

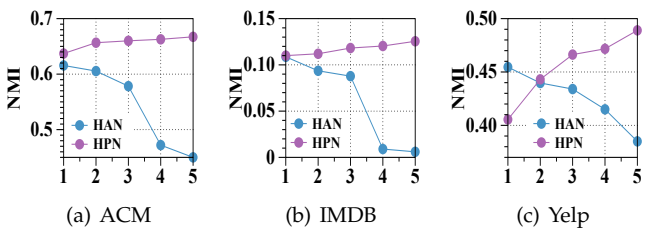


(a) GAT (b) PPNP (c) HAN (d) HPN

Fig. 5: Visualization paper embedding on ACM. Each point indicates one paper and its color indicates the research area.

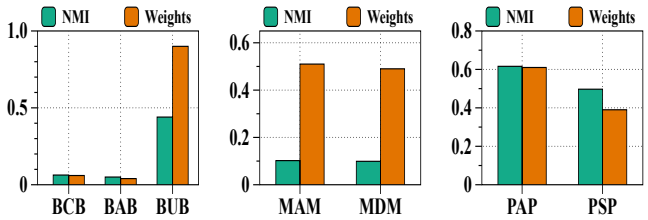
TABLE 4: Qantitative results (%) on the node recommendation. The larger values, the better performace.

Model	ML			
	recall@10	prec@10	hit@10	ndcg@10
GCN	20.09	32.08	89.28	40.06
GAT	20.78	32.66	89.88	40.79
PPNP	21.41	33.21	90.66	41.91
MEIRec	22.19	34.19	91.12	43.09
HAN	22.14	34.21	91.04	43.01
HGT	22.21	34.25	91.21	43.11
MG	22.29	34.28	91.12	43.04
HPN_{pro}	22.02	33.89	91.02	42.35
HPN_{fus}	22.71	**34.82**	91.91	**43.36**
HPN	**22.73**	34.81	**91.93**	43.34



(a) ACM (b) IMDB (c) Yelp

Fig. 6: Clustering results of HAN/HPN with 1,2,3,4,5 layers.



(a) Yelp (b) IMDB (c) ACM

Fig. 7: Performance and weights of different meta-paths.

Yelp, meta-path *BUB* is much more important than the rest. Since the importance of meta-paths can be quite different, if we treat these meta-paths equally (e.g., HPN_{fus}), the performance will drop significantly. For ACM, the proposed HPN gives *PAP* the larger weight, which means HPN considers the *PAP* as the more critical meta-path in identifying the paper's research area. We also find that although *PAP* shows superiority over *PSP*, the gap is not very large. It can explain why HPN_{fus} still works well on ACM with simple average operation on all meta-paths. We can find similar phenomenon on the IMDB. *MAM* performs slightly better than *MDM*, so HPN_{fus} also performs well by average operation as shown in Tables 3 and 2. In summary, semantic fusion mechanism can tell the difference of meta-paths and

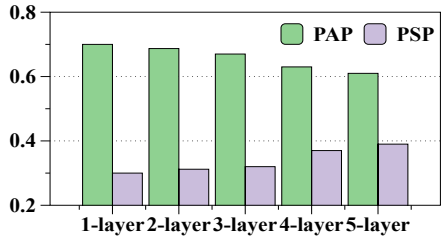
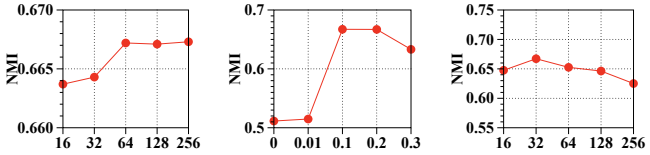


Fig. 8: The learned attention weights of different meta-paths with 1, 2, 3, 4, 5 layers on ACM via HPN.



(a) Dimension of the final embedding $\mathbf{Z}$. (b) Value of the restart probability γ. (c) Dimension of the semantic fusion vector $\mathbf{q}$.

Fig. 9: Parameter study of the proposed HPN on ACM.

assign proper weights to them.

5.7.2 *Semantic Fusion Mechanism with Different Depths.*

Then, we test semantic fusion mechanism with different depths (e.g., HPN with 1,2,3,4,5 layers). The propagation property of each meta-path can be quite different, i.e., after propagating for several steps, the importance of meta-paths may change a lot. Taking ACM as an example, we show how the weights of meta-paths change with different number of layers in Figure 8. As can be seen, after propagating for several steps, the attention weight of *PSP* becomes higher and higher, while the attention weight of *PAP* becomes lower and lower, which indicates *PSP* is getting more and more important. When deeper insight into such phenomenon, we find that *PAP* may connect two papers belonging to different research areas because some authors have diverse research interests and publish papers in different research areas. So semantic propagation process will introduce some noise via *PAP* and mixes papers in different research areas and makes their embeddings indistinguishable. On the contrary, papers connected via *PSP* always belong to the same research area due to the same subject. So even propagating for several steps, *PSP* still differentiates the characteristics of papers clearly. It explains why semantic fusion mechanism pays more attention to *PSP* with the growth of model depth. In summary, different meta-paths have different propagate properties and semantic fusion mechanism is able to assign suitable attention weights to them.

5.8 Parameters Experiments

In this section, we investigate the sensitivity of parameters and report the results of clustering (*NMI*) on ACM dataset with various parameters in Figure 9.

•**Dimension of the final embedding Z.** We first test the effect of the dimension of the final embedding $\mathbf{Z}$. The results are shown in Figure 9(a). We can see that with the growth of the embedding dimension, the performance of HPN raises first and then remains stable. The reason is that higher dimension can encode more semantics and a suitable dimension can capture all semantics. For ACM dataset, the proposed HPN achieves the best performance when the dimension of the final embedding $\mathbf{Z}$ is set to 64.

•**Value of restart probability γ.** In order to check the impact of restart probability γ, we explore the performance of the proposed HPN with various values of γ and show them in Figure 9(b). Note that the restart process is removed when the value of γ is set to 0. Here larger γ means the proposed HPN pays more attention to local semantics. We can find that the restart process ($\gamma > 0$) can improve the performance of HPN. However, the larger γ will lead to the worse performance. A suitable γ is able to balance local semantics and global semantics properly and improve performance. Here HPN achieves the best performance when restart probability γ is set to 0.1.

•**Dimension of semantic fusion vector q**. Since the ability of semantic fusion mechanism is affected by the dimension of attention vector $\mathbf{q}$, we change its dimension and test the performance of HPN. The results are shown in Figure 9(c). With the growth of dimension of $\mathbf{q}$, the performance of HPN increases in the beginning and decreases significantly when the dimension of $\mathbf{q}$ is larger than 32. One possible reason is that there are no learnable parameters in the semantic propagation mechanism, so the larger dimension of attention vector $\mathbf{q}$ will significantly increase the total number of parameters in HPN and lead to over-fitting.

6 Related Work

6.1 Graph Neural Network

Graph neural networks generalize deep learning to graph-structured data, which follows the message-passing framework to receive messages from neighbors and apply neural network to update node embedding. Michaël et al. [38] propose a spectural graph convolutional network which leverages *K*-order Chebyshev polynomials to approximate smooth filters. [1] proposes GCN via a localized approximation of spectral graph convolutions. [39] proposes a inductive GraphSAGE model which leverage neighbor sampling and flexible aggregating function to learn node embedding. To improving the neighbors aggregation scheme in GNNs, Veličković et al. [34] utilizes attention mechanism to learn the improtance of neighbors and aggregate them properly. DGI [40] tries to maximize mutual information between global and local representations and learn node embedding in an unsupervised manner. Some works [18], [35] try to give a theoretical analysis of graph convolutional network. [18] demonstrates GCN is a special form of Laplacian smoothing. Klicpera et al. [35] utilize the relationship between GCN and PageRank to derive an improved propagation scheme based on personalized PageRank. All the above graph neural networks focus on homogeneous graphs.

Some works [13], [14], [23], [24], [41], [42] extend GNNs to the heterogeneous graph. [41] proposes GraphInception to learn relational features for collective classification. [13] and [15] both leverage hierarchical aggregation to capture rich semantics. [23] adopts heterogeneous mutual attention to aggregate meta relation triplet, and [24] leverages relational rotation encoder to aggregate meta-path instances.

[42] learns a soft selection of edge types and generate meta-paths automatically, solving the problem of meta-path selection. Some works [14], [16], [17] leverage HeteGNNs to solve diverse recommendation tasks. Although there are several attempts in design HeteGNNs, the working mechanism of HeteGNNs still remains to be analyzed.

6.2 Network Embedding

Network embedding aims to project node into low-dimensional space while preserving the network structure and property. For example, the random walk based methods [43], [44], the deep neural network based methods [45], the matrix factorization based methods [46], and others, e.g., LINE [47]. [43] leverages random walk to get the context and adopts skip-gram model to learn node embedding. Node2vec [44] leverages BFS and DFS to get more flexible contexts. Wang et al. [45] introduce deep autoencoder to capture the non-linear graph structure. However, all above algorithms are proposed for the homogeneous graphs.

Heterogeneous graph embedding focuses on capture rich semantics via meta-path. Metapath2vec [32] utilizes meta-path based random walk and skip-gram to embed heterogeneous graph. HIN2Vec [48] captures the rich semantics by exploiting different types of relationships among nodes. [49] embeds heterogeneous graph into hyperbolic space to capture hierarchical structure. RHINE [50] designs a relation structure-aware heterogeneous graph embedding method which can distinguish the difference of affiliation relations and interaction relations. [51] propose PME to capture both first-order and second-order proximities in a unified way. HERec [33] captures rich semantics via meta-path based random walk and learn node embedding via skip-gram.

7 Conclusion

In this paper, we explore the semantic confusion phenomenon in HeteGNNs, i.e., with the growth of model depth, the node embeddings learned via HeteGNNs become indistinguishable, leading to the degradation of performance and limiting their representation capabilities and applications. We explain semantic confusion by demonstrating that HeteGNNs and multiple meta-paths based random walk are essentially equivalent. Then, we propose a novel heterogeneous graph propagation network (HPN) to alleviate the semantic confusion in node-level, which mainly consists of semantic propagation mechanism and semantic fusion mechanism. Specifically, the semantic propagation mechanism absorbs node's local semantic with a proper weight during node-level aggregating, alleviating the semantic confusion under deep HeteGNN architecture. And, the semantic fusion mechanism is designed to fuse rich semantics and comprehensively describe the node from different aspects. Experimental results show the superiority of the proposed HPN. More importanctly, we analyze the importance and propagation property of meta-path which may help to understand heterogeneous graph.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. U20B2045, 61772082, 61702296, 62002029, U1936104, 61972442), National Social Science Fund of China (18CSH019), Beijing Social Science Fund (20JCC096), and NSF under grants III-1763325, III-1909323, and SaTC-1930941.

References

[1] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.
[2] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *NIPS*, 2018, pp. 4800–4810.
[3] A. Li, Z. Qin, R. Liu, Y. Yang, and D. Li, "Spam review detection with graph convolutional networks," in *CIKM*, 2019, pp. 2703–2711.
[4] J. Klicpera, J. Groß, and S. Günnemann, "Directional message passing for molecular graphs," in *ICLR*, 2020.
[5] L. Hu, T. Yang, C. Shi, H. Ji, and X. Li, "Heterogeneous graph attention networks for semi-supervised short text classification," in *EMNLP*, 2019, pp. 4823–4832.
[6] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," in *ArXiv*, 2018.
[7] Z. Wu, S. Pan, F. Chen, G. Long, and P. S. Yu, "A comprehensive survey on graph neural networks," in *ArXiv*, 2019.
[8] J. You, R. Ying, and J. Leskovec, "Position-aware graph neural networks," in *ICML*, 2019, pp. 7134–7143.
[9] H. Pei, B. Wei, K. C.-C. Chang, Y. Lei, and B. Yang, "Geom-gcn: Geometric graph convolutional networks," in *ICLR*, 2020.
[10] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," in *KDD*, 2018.
[11] C. Shi, Y. Li, J. Zhang, Y. Sun, and P. S. Yu, "A survey of heterogeneous information network analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, pp. 17–37, 2017.
[12] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," in *VLDB*, 2011, pp. 992–1003.
[13] 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.
[14] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, and Y. Li, "Metapath-guided heterogeneous graph neural network for intent recommendation," in *KDD*, 2019, pp. 2478–2486.
[15] C. Zhang, D. Song, C. Huang, A. Swami, and C. V. Nitesh, "Heterogeneous graph neural network," in *KDD*, 2019, pp. 793–803.
[16] H. Ji, J. Zhu, C. Shi, X. Wang, B. Wang, C. Zhang, Z. Zhu, F. Zhang, and Y. Li, "Large-scale comb-k recommendation," in *WWW*, 2021.
[17] H. Ji, J. Zhu, X. Wang, C. Shi, B. Wang, X. Tan, Y. Li, and S. He, "Who you would like to share with? a study of share recommendation in social e-commerce," in *AAAI*, 2021.
[18] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *AAAI*, 2018, pp. 3538–3545.
[19] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-i. Kawarabayashi, and S. Jegelka, "Representation learning on graphs with jumping knowledge networks," pp. 5453–5462, 2018.
[20] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *AISTATS*, 2011.
[21] S. Lee, S. Park, M. Kahng, and S.-G. Lee, "Pathrank: Ranking nodes on a heterogeneous graph for flexible hybrid recommender systems," *Expert Systems with Applications*, vol. 40, no. 2, pp. 684–697, 2013.
[22] Y. Sun and J. Han, "Mining heterogeneous information networks: a structural analysis approach," *Acm Sigkdd Explorations Newsletter*, vol. 14, no. 2, pp. 20–28, 2013.
[23] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *WWW*, 2020.
[24] X. Fu, J. Zhang, Z. Meng, and I. King, "Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding," in *WWW*, 2020.
[25] D. Zill, W. S. Wright, and M. R. Cullen, *Advanced engineering mathematics*. Jones & Bartlett Learning, 2011.
[26] C. Zhang, H. Fu, Q. Hu, X. Cao, Y. Xie, D. Tao, and D. Xu, "Generalized latent multi-view subspace clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 1, pp. 86–99, 2018.

[27] C. Zhang, Y. Cui, Z. Han, J. T. Zhou, H. Fu, and Q. Hu, "Deep partial multi-view learning," *IEEE transactions on pattern analysis and machine intelligence*, 2020.
[28] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *SIGIR*, 2019, pp. 165–174.
[29] J. Shi, H. Ji, C. Shi, X. Wang, Z. Zhang, and J. Zhou, "Heterogeneous graph neural network for recommendation," in *ICML*, 2020.
[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.
[31] C. Zhang, A. Swami, and N. V. Chawla, "Shne: Representation learning for semantic-associated heterogeneous networks," 2019.
[32] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *KDD*, 2017.
[33] C. Shi, B. Hu, X. Zhao, and P. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, 2018.
[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *ICLR*, 2018.
[35] J. Klicpera, A. Bojchevski, and S. Günnemann, "Predict then propagate: Graph neural networks meet personalized pagerank," in *ICLR*, 2019.
[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.
[37] L. V. D. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2605, pp. 2579–2605, 2008.
[38] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *NIPS*, 2016.
[39] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017, pp. 1024–1034.
[40] P. Veličković, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *ICLR*, 2019.
[41] 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.
[42] S. Yun, M. Jeong, R. Kim, J. Kang, and H. Kim, "Graph transformer networks," in *NeurIPS*, 2019.
[43] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014.
[44] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, 2016, pp. 855–864.
[45] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *KDD*, 2016, pp. 1225–1234.
[46] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding." in *AAAI*, 2017.
[47] 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.
[48] 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.
[49] X. Wang, Y. Zhang, and C. Shi, "Hyperbolic heterogeneous information network embedding," in *AAAI*, vol. 33, 2019, pp. 5337–5344.
[50] Y. Lu, C. Shi, L. Hu, and Z. Liu, "Relation structure-aware heterogeneous information network embedding," in *AAAI*, vol. 33, 2019, pp. 4456–4463.
[51] 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 *KDD*, 2018, pp. 1177–1186.

Houye Ji is currently a PhD student in Beijing University of Posts and Telecommunications. His research interests are in heterogeneous graph and graph neural network. He has published several papers in WWW, AAAI, EMNLP, TOIS and PRICAI.

Xiao Wang received the PhD degree from the School of Computer Science and Technology, Tianjin University, in 2016. He is currently an assistant professor with the Beijing University of Posts and Telecommunications, Beijing, China. Prior to that, he was a postdoctoral researcher in Tsinghua University. His current research interests include data mining and social network analysis. Until now, he has published more than 30 refereed journals and conferences, such as SIGKDD, IJCAI, AAAI, IEEE TKDE and WWW.

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, machine learning, and evolutionary computing. He has published more than 100 papers in refereed journals and conferences, such as SIGKDD, IJCAI, CIKM, IEEE TKDE, WWWJ, and ACM TIST.

Bai Wang received the B.S. degree from the Xian Jiaotong University, Xian, China and Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China. And she is currently a professor of computer science in BUPT. She was the director of Beijing Key Lab of Intelligent Telecommunications Software and Multimedia.

Philip S. Yu received the B.S. Degree in E.E. from National Taiwan University, the M.S. and Ph.D. degrees in E.E. from Stanford University, and the M.B.A. degree from New York University. He is a Distinguished Professor in Computer Science at the University of Illinois at Chicago and also holds the Wexler Chair in Information Technology. Before joining UIC, Dr. Yu was with IBM, where he was manager of the Software Tools and Techniques department at the Watson Research Center. His research interest is on big data, including data mining, data stream, database and privacy. He has published more than 1,200 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. Dr. Yu is a Fellow of the ACM and the IEEE. Dr. Yu is the recipient of ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions on *mining, fusion and anonymization of big data*, the IEEE Computer Society's 2013 Technical Achievement Award for "*pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining and anonymization of big data*", and the Research Contributions Award from IEEE Intl. Conference on Data Mining (ICDM) in 2003 for his pioneering contributions to the field of data mining. He also received the ICDM 2013 10-year Highest-Impact Paper Award, and the EDBT Test of Time Award (2014). He was the Editor-in-Chiefs of ACM Transactions on Knowledge Discovery from Data (2011-2017) and IEEE Transactions on Knowledge and Data Engineering (2001-2004).