Information Sciences 600 (2022) 287–300

Contents lists available at ScienceDirect

Information Sciences

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

Approximate personalized propagation for unsupervised embedding in heterogeneous graphs

Yibi Chen [a,b], Yikun Hu [a,*], Keqin Li [a,c], Chai Kiat Yeo [b], Kenli Li [a]

[a] College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
[b] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[c] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

ARTICLE INFO

Article history:
Received 1 July 2021
Received in revised form 15 February 2022
Accepted 2 April 2022
Available online 5 April 2022

Keywords:
Approximate personalized propagation
Heterogeneous graphs
Neighborhood information
Robustness
Throwedge

ABSTRACT

Graphs are effective for representing various relationships in the real world and have been successfully applied in many areas, such as publication citations and movie networks. Compared to homogeneous graphs (i.e., nodes and edges of a single relation type), heterogeneous graphs have heterogeneity and richer information (i.e., nodes and edges of different relation types). How to tackle complex non-pairwise graph-structured data and model various relation-types is a daunting challenge for heterogenous graphs. However, the existing unsupervised methods focus on node attribute learning, while node neighborhood information utilizes very limited because they only consider node propagation that is within few steps. In this paper, we propose an unsupervised method, called APPTE, that models adequate node neighborhood information in local context, and captures the global neighborhood information. Meanwhile, our method considers the robustness and generalization ability. Specifically, we construct approximate personalized propagation in local context to utilize an infinite number of neighborhood aggregation layers for extending node neighborhood propagation range, and then fuse these local context to capture global neighborhood information. Additionally, we improve the robustness and generalization ability of model, employing throwedge to increase the randomness and diversity of the graph connections by randomly deleting a part of edges. The experimental results on three benchmark datasets containing heterogeneous graphs demonstrate that our proposed method is superior to the available state-of-the-art methods.

1. Introduction

1.1. Motivation

Deep learning approaches [1,2] have been successful in many applications and have obtained impressive results in processing Euclidean data. For example, a convolutional neural network (CNN) can effectively extract image and video data (i.e., the spatial organization of pixels) [3,4]. Extending Euclidean data [5] processing to non-Euclidean data processing is the development trend of deep learning technology. Typical non-Euclidean data can be represented by graph-structured data

[*] Corresponding author.
E-mail addresses: chenyibi@hnu.edu.cn (Y. Chen), yikunhu@hnu.edu.cn (Y. Hu), lik@newpaltz.edu (K. Li), asckyeo@ntu.edu.sg (C.K. Yeo), lkl@hnu.edu.cn (K. Li).

https://doi.org/10.1016/j.ins.2022.04.002
0020-0255/© 2022 Published by Elsevier Inc.

[6–8], which are usually irregular because they contain complex structural relationships. A graph contains nodes and edges and can establish various relation types in the real world, such as in social networks [9,10] and publication citations [11,12].

Heterogeneous graphs [13,14] have nodes and edges of different relation-types, and the nodes are related to each other in various ways. Taking the Association for Computing Machinery (ACM) dataset as an example, two papers written by the same author have established connections by choosing the metapath as Paper-Author-Paper; two papers have established connections based on the same subject by choosing the metapath as Paper-Subject-Paper. Various types of relationships can be used to generate different graphs through different metapath forms. These graphs are usually related, and they can help each other to achieve multiple downstream tasks (i.e., node classification, similarity searches and node clustering). Furthermore, each node has a potential relationship with its neighbors and can be affected by its close and distant neighbors at any time. Generally, the smaller the distance between a node and its neighbors is, the greater influence the neighbors have on the node. The propagation distance between a node and a neighboring node affects the amount of information obtained by the neighboring node, especially for different types of nodes with different characteristics and neighborhood relations. Based on the above analysis, the great challenge is to fully extract node neighborhood information and model various relation-types.

Existing supervised methods have performed graph representation learning in heterogeneous graphs. Sankar et al. [15] established a model for determining the key attributes based on the spatial convolution idea. Chen et al. [16] established a heterogeneous information network model to tackle the potential geometrical inflexibility in metric learning. Wang et al. [17] adopted a semi-supervised approach that processes node-level and semantic-level information separately through an attention mechanism and then performs information fusion. However, these methods apply node labels during training, but these labels are expensive.

Recently, several unsupervised methods have been put forward to solve above problem. Velickovic et al. [18] proposed to learn node attributes representations based on maximizing the mutual information (MI) between local patches and graph summaries, but are limited in single network. Follow that, Park et al. [19] established an attributes multiplex network, the differences among node relationship types based on local structure features are minimized through a consensus regularization, and real samples were distinguished by discriminators without considering the relationship types. Although the above methods have achieved good results, they are aimed at local and global attribute information. However, these methods are limited in the following respects. 1) They only consider the one-hop propagation of nodes. Therefore, it limits the propagation range of nodes and causes neighborhood information of nodes in a few utilized, especially for unsupervised method. 2) Heterogeneous graphs have heterogeneity and richer information based on different relation-types so that it are closer to dynamic changes in the real world. But, above methods does not consider the robustness and generalization ability.

1.2. Our contributions

To address these limitation, we develop an unsupervised method called APPTE that capture abundant node neighborhood information to achieve high-quality embeddings. Meanwhile, APPTE considers robustness and generalization ability. More precisely, we model the local context (i.e., each type) by personalized propagation which utilizes an infinite number of neighborhood aggregation layers. Then, approximate personalized propagation adopts an approximate method (i.e., power iteration) so that the computational complexity of the personalized propagation process is linear and the node neighborhood information is sufficient obtained. Next, we fuse these local context to capture global neighborhood information. Inspired by the above linear complexity of approximate personalized propagation that some edges information can be reduced, throwedge randomly deletes a part of edges to increase the randomness and diversity of the graph connections so that the model obtains better robustness and generalization ability. Moreover, APPTE requires only a few parameters to propagate information to more distant neighbors. Our method is validated on three benchmark heterogeneous graph datasets and is superior to the state-of-art baseline.

We summarize the main contributions in this paper as follows.

- We propose an unsupervised method called APPTE for heterogeneous graphs. This method enlarges the node neighborhood propagation range to capture adequate node neighborhood information.
- APPTE has an end-to-end structure, in which a part of edges are deleted to increase the randomness and diversity of the graph connections so that the model obtains better robustness and generalization ability.
- Our approach requires only a few parameters to propagate information to more distant neighbors.
- The experimental results on three benchmark datasets containing heterogeneous graphs demonstrate that our proposed APPTE method is superior to the available state-of-the-art methods.

1.3. Organization

The rest of this paper is organized as follows. Section 2, we review development of various graph technologies. Section 3 presents the preliminary notations and the task. In Section 4, we propose an unsupervised method and describe in detail our proposed APPTE framework. In Section 5, we perform extensive experiments and show the results. The conclusions are presented in Section 6.

2. Related work

With the development of various graph technologies, we review previous approaches, such as GCNs, message passing and multiplex network embedding.

2.1. Graph convolutional network

A GCN extracts node features and local neighbor information for graph-structured data. Bruna et al. [20] proposed domain-based hierarchical clustering on general graphs with local connection and pooling operations to reduce parameters and the graph Laplacian spectrum, which produces a convolution operator for global graph structures. Defferrard et al. [21] created a fast localized convolution filters for graphs based on the idea of a CNN. Kipf et al. [22] devised a spectral graph convolution method for semisupervised classification to select a local first-order approximation convolution structure to scale the number of graph edges and to represent the local graph structure and node features in hidden layers. Zhuang et al. [23] constructed a dual CNN that simultaneously considers both the local consistency and global consistency of knowledge. Guo et al. [24] designed a three-objective optimization scheme (i.e., partitioning-updating-tracking) for regions of interest. Rong et al. [25] proposed some edges are removed in homogeneous graph, which solves over-smoothing and over-fitting in deep GCN. Hamilton et al. [26] presented a method for learning node features and local neighborhoods on large graphs to obtain node embeddings of unknown graphs. Velivckovic et al. [18] described a method to maximize MI between patch representations and advanced graph summaries based on a GCN to enhance the correlations among the information, but this method is limited to a single network.

2.2. Message passing

Message passing is how nodes share information with their neighbors. Li et al. [27] introduced a combined message passing algorithm with cotraining and self-training in semisupervised learning to tackle the over-smoothing problem. Ying et al. [28] investigated combining random walk and graph convolution to obtain a node embedding, and then integrated it into the MapReduce model. Xu et al. [29] designed a jumping knowledge network, which aims at the different neighborhood range of each node to obtain a better graph structure representation. Kawamoto et al. [30] pointed out that whether a graph neural network obtains higher accuracy is determined by the backpropagation result or the architecture, and the mean value theory of the minimum graph neural network was designed. Guo et al. [31] proposed a multiobjective optimization method to improve the robustness of the model to explore Pareto-optima with time change. Chen et al. [32] employed the nonbacktracking operator to enhance a graph neural network to obtain the loss values at the global and local minima. Klicpera et al. [33] developed a message passing algorithm to separate a GCN and PageRank for semi-supervised classification. Wang et al. [17] adopted an attention mechanism to obtain the node-level and semantic-level features of heterogeneous graph and then fused them; however, this approach fails to consider mutual information.

2.3. Multiplex network embedding

Multiplex network embedding is composed of multiple relation-type information formed among single-type nodes to map an embedding space. Zhang et al. [34] performed high-dimensional embedding and low-dimensional embedding for each relation type and then exploited a network embedding model to learn multiple pieces of information. Fu et al. [35] proposed performing multiple prediction training tasks on a target relationship set to learn potential node information. Ma et al. [36] proposed to obtaining independent information in each dimension and relevant information across dimensions to learn the hierarchical representation of a multidimensional network embedding. Dong et al. [37] proposed constructing node neighborhoods on heterogeneous graphs with the random walk, and then simultaneously modeling both the structural and semantic correlations in heterogeneous networks. Park et al. [19] mentioned that the differences among node relationship types based on local structure features are minimized through a consensus regularization, and real samples were distinguished by discriminators without considering the relationship types, but the limited node propagation range leads to insufficient utilization of node neighborhood information.

In summary, we emphasize that APPTE is very different from the abovementioned literature. A novel heterogeneous graph architecture are designed to extract adequate node neighborhood information in local context, and capture the global neighborhood information. The robustness and generalization ability are considered in our model.

3. Preliminary

We define some concepts and task descriptions for heterogeneous graphs.

Definition 1. (Multiplex network): A multiplex network defined as $G = \left\{G^1, G^2, \ldots, G^{|\mathrm{K}|}\right\} = \{V, \xi, \mathrm{X}\}$. V is the node set, ξ is the edge set and X is the node feature matrix. $X \in \mathbb{R}^{n\times f}$ includes n nodes and each node has f feature information. $G^k = \left\{V, \xi^{(k)}, \mathrm{X}\right\}$ is a relation-type graph, $k \in \mathrm{K}, |\mathrm{K}| = 1$ is a single network, and $|\mathrm{K}| > 1$ is a multiplex network. The adjacency matrix is denoted as $A = \left\{A^{(1)}, \ldots, A^{(|K|)}\right\}$, and $A^{(k)} \in \{0,1\}^{|V|\times|V|}$ represents the adjacency matrix of network G^k.

Definition 2. (Metapath): A metapath P is described as $P_1 \xrightarrow{R_1} P_2 \xrightarrow{R_2} \cdots \xrightarrow{R_n} P_{n+1}$, where $R = R_1 \circ R_2 \cdots \circ R_n$ illustrates the composite relation between P_1 and P_{n+1} and $\circ$ is a composition operator that describes these relations.

Task 1 (Heterogeneous graph unsupervised embedding): Given metapath P, multiplex network G and adjacency matrix A, the task of heterogeneous graphs unsupervised embedding aims to learn the d-dimensional representation of each node $v_i \in V$ without use of labeled data.

4. The proposed APPTE framework

In this section, we describe in detail our proposed APPTE framework from three modules: Heterogeneous graph structured-data extraction mechanism, Mutual information and Consensus regularization. Fig. 1 presents the our method architecture.

We learn a high-quality embedding in heterogeneous graphs through APPTE. In Fig. 1, the purple line represents the process from the initial network to the corrupted network that initial node features X (i.e., positive samples) are destroyed to generate corrupted node features $\widetilde{X}$ (i.e., negative samples). Heterogeneous graph structured-data extraction mechanism consists of throwedge, GCN and approximate personalized propagation to obtain each type of node embedding matrix $m^{(k)}$, and then we fuse node embedding matrix of these type to generate the relation-type node embedding matrix $M'^{(k)}$ (i.e., global neighborhood information). Mutual information contains readout function and discriminator, in which the readout function calculates $M'^{(k)}$ to obtain the global summary representation, and then discriminator which represents yellow D computes the MI between the positive sample pairs and negative sample pairs for maximization, respectively. Consensus regularization establishes the multiple relation-type embedding matrix of the positive sample (i.e., green cuboid) and negative sample (i.e., purple cuboid) to minimize disagreements. Finally, we employ Adam optimizer to optimize the model and calculate the sum of the loss values to obtain objective function.

4.1. Heterogeneous graph structured-data extraction mechanism

The idea of single network [18] and multiplex network [19] both extracts node attributes to obtain node embedding matrix $h^{(k)}$ and $\widetilde{h}^{(k)}$ via a GCN, which is described in Section 4.1.1. They only consider one-hop neighbors so that node neigh-

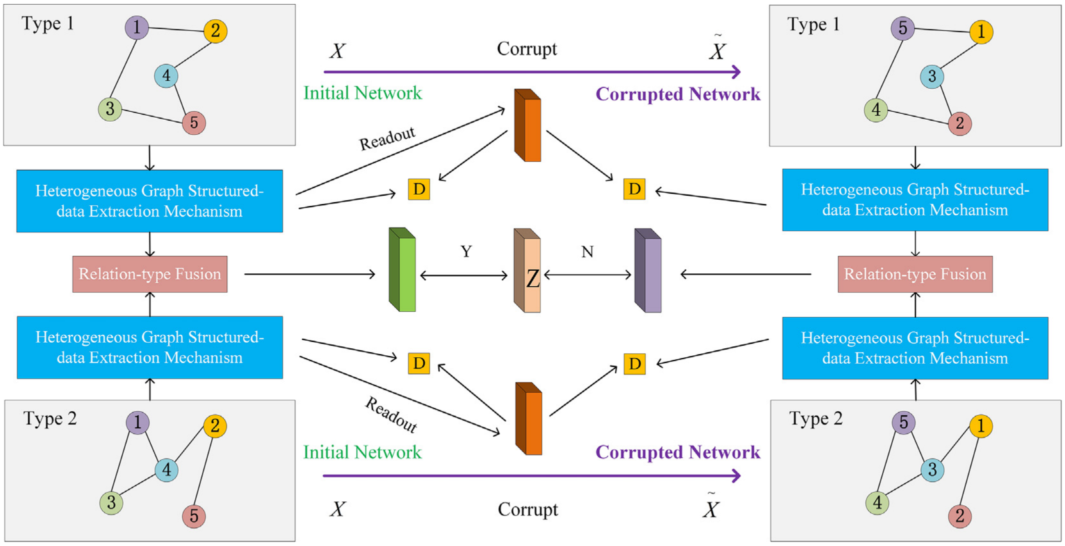


Fig. 1. Architecture of APPTE.

borhood information are limited. This motivates us to capture more neighborhood information by expanding node neighborhood propagation range. For multiplex network, expansion of the node neighborhood range directly on the embedding of multiple relation-types bring each node obtaining more redundancy information from it neighbors, this cause the model to not achieve high-quality embedding. Therefore, we model each graph represented by each type to expand node neighborhood range, which capture adequate neighborhood information from relative simple relation-type to avoid above redundancy information. Furthermore, we considered robustness and generalization ability based on the above scheme.

We describes the extraction mechanism and provides its mathematical definition and derivations. We explain how extending node neighborhood propagation range and enhancing model robustness and generalization ability. Extraction mechanism framework is shown in Fig. 2.

4.1.1. Graph convolutional network

In APPTE, the GCN performs convolution operations for each type $k \in K$ to obtain node embedding matrix $h^{(k)}$ of all nodes in $G^{(k)}$. The node feature matrix $X \in \mathbb{R}^{n \times f}$ and adjacency matrix $A^{(k)} \in \mathbb{R}^{n \times n}$ are used as the inputs of the GCN. The output of the GCN is represented as $\mathbb{R}^{n \times f} \rightarrow \mathbb{R}^{n \times d}$:

$$h^{(k)} = ReLU\left(\overline{D}_k^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}_k^{-\frac{1}{2}}XW^{(k)}\right), \tag{1}$$

where, $\overline{A}^{(k)} = A^{(k)} + I_n$ denotes the adjacency matrix with added self-loops, I_n is a unit matrix, $\overline{D}_{ii} = \sum_j \overline{A}_{ij}$ is diagonal degree matrix and $W^{(k)} \in \mathbb{R}^{f \times d}$ is a weight matrix.

The corrupted node feature matrix is obtained by shuffling the initial node feature matrix in a row-wise manner, it destroying the information of initial feature matrix, i.e., $X \rightarrow \widetilde{X}$. The GCN corrupted feature matrix as follows:

$$\widetilde{h}^{(k)} = ReLU\left(\overline{D}_k^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}_k^{-\frac{1}{2}}\widetilde{X}W^{(k)}\right). \tag{2}$$

4.1.2. ThrowEdge

We randomly deletes a part of the edge to disturb graph connections data for each type, this increase data the randomness and diversity so that the model obtains better robustness and generalization ability. This manner can be seen as a data augmentation technology especially for heterogeneous graphs, similar to traditional image augmentation technology such as rotation and cropping. Specifically, T_f non-zero elements of the adjacency matrix $A^{(k)}$ are randomly set to zeros, where T represents the number of edges and f represents the throwing rate. We obtain adjacency matrix A_{throw} as follows:

$$A_{throw} = A - A', \tag{3}$$

where A' represents a sparse matrix which is a random subset of the size T_f generated by the initial edge ξ. Furthermore, we obtain $\overline{A}_{throw}$ to perform the re-normalization operation on A_{throw}. Noted that throwedge was not used in the validation and testing process.

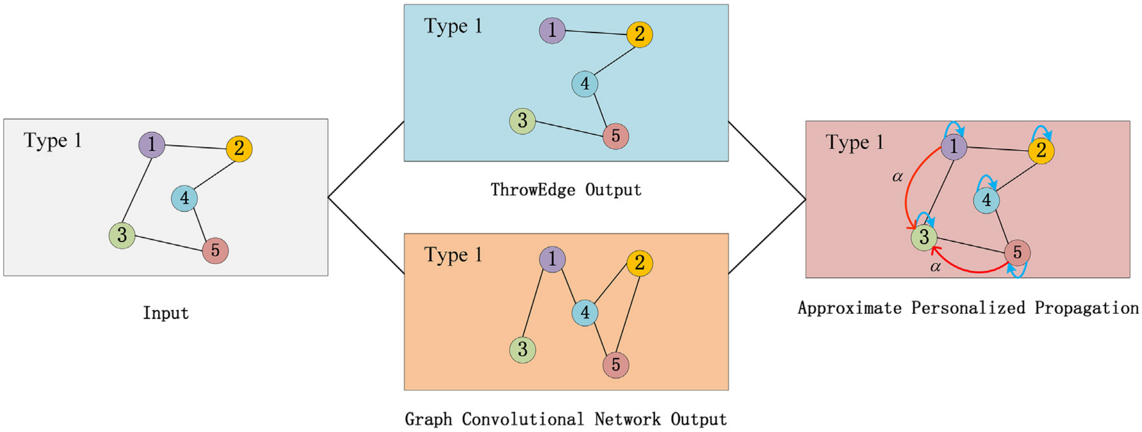

Type 1
1
2
4
3
5
ThrowEdge Output
Type 1
Input
Type 1
α
α
Approximate Personalized Propagation
Type 1
Graph Convolutional Network Output

Fig. 2. Flowchart of heterogeneous graph structured-data extraction mechanism.

4.1.3. Approximate personalized propagation

We firstly describe how personalized propagation obtains more node neighborhoods information, and then approximate personalized propagation applies an approximate method on personalized propagation to reduce the amount of calculation.

Personalized PageRank can aggregate an infinite number of node neighborhood layers. It considers the probability of teleporting back to every root node, and the neighborhood of each root node $v_j \in V$ is encoded based on the PageRank score. The root node v_j is determined by teleport vector I_{v_j} which preserves the neighborhood of the root node. Taking $G^{(k)}$ as an example, the recurrent equation realizes adaptation of personalized PageRank for root nodes; this is represented as follows:

$$\tau_{ppr}^{(k)}\left(I_{v_j}\right) = (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}^{-\frac{1}{2}}\tau_{ppr}^{(k)}\left(I_{v_j}\right) + \alpha I_{v_j}, \tag{4}$$

where $\alpha \in (0,1]$ is the teleport probability for adjusting the descending speed of a neighboring node as it moves away from the root node. In other words, it controls the neighborhood range of each node. By solving Equation (4), we can obtain:

$$\tau_{ppr}^{(k)}\left(I_{v_j}\right) = \alpha\left(I_n - (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}^{-\frac{1}{2}}\right)^{-1} I_{v_j}. \tag{5}$$

Each root node has different influence scores for its neighboring nodes. For example, $I(v_j, v_y)$ represents the influence score of the root node v_j on node v_y, which is proportional to the y-th element of personalized PageRank $\tau_{ppr}^{(k)}\left(I_{v_j}\right)$. Furthermore, the indicator vector I_{v_j} is replaced with identity matrix I_n to obtain a fully personalized PageRank matrix, which is described as follows:

$$\Pi_{ppr}^{(k)} = \alpha\left(I_n - (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}^{-\frac{1}{2}}\right)^{-1}, \tag{6}$$

$\Pi_{ppr\left(v_j v_y\right)}^{(k)} = \Pi_{ppr\left(v_y v_j\right)}^{(k)}$ indicates that the influence between node v_j and node v_y is the same, and $I(v_j, v_y) \propto \Pi_{ppr\left(v_j v_y\right)}^{(k)}$.

For personalized propagation, we encode the node feature matrix with GCN to obtain node embedding matrix of each type (i.e., $h^{(k)}$ and $\widetilde{h}^{(k)}$), and then fully personalized PageRank propagates $h^{(k)}$ and $\widetilde{h}^{(k)}$ to aggregate more node neighborhood information. Taking $h^{(k)}$ as an example, the equation is described as follows:

$$m^{(k)} = \alpha\left(I_n - (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}^{-\frac{1}{2}}\right)^{-1} h^{(k)}. \tag{7}$$

In Equation (7), we see that personalized propagation is used to calculate dense matrix $\mathbb{R}^{n\times n}$ to obtain memory requirement of $O(n^2)$. To tackle this problem, approximate personalized propagation utilizes power iterations for personalized propagation to obtain a linear computational complexity; we employ the adjacency matrix $\overline{A}_{throw}$ obtained in throwedge to represent the graph structure, and the matrix $\mathbb{R}^{n\times n}$ is never constructed. Different from the original PageRank method, which adopted the normal random walk, the power iteration of personalized propagation is connected to the restarting random walk method, which considers a process for teleporting back to the root nodes. Each power iteration process is described as follows::

$$\begin{aligned} m_{(0)}^{(k)} &= h^{(k)}, \\ m_{(n)}^{(k)} &= (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}_{throw}^{(k)}\overline{D}^{-\frac{1}{2}}m_{(n-1)}^{(k)} + \alpha h^{(k)}, \\ m'^{(k)} &= m_{(N)}^{(k)}, \end{aligned} \tag{8}$$

where $h^{(k)}$ serves as both the teleport set and the beginning vector, and N represents the number of power iterations (i.e., $n \in [0, N-2]$).

Compared with the GCN that needs to provide more parameters for each additional layer, approximate personalized propagation only requires a few parameters and no additional layers to propagate very far neighbors. In the propagation scheme, the gradient flows participate in the backpropagation process of infinite neighborhood aggregation layers, which greatly improves model accuracy. We fuse each type of neighborhood information to capture relation-type embedding matrix (i.e., $M'^{(k)} = \{m^{(1)}, m^{(2)}, .., m^{(k)}\}$).

4.2. Mutual information

MI between the local patches and the global summary representation is maximized to learn the graph-structured representation. We obtain local patches $\left\{m_1^{(k)}, m_2^{(k)}, .., m_i^{(k)}\right\}$ as follows:

$$\left\{m_1^{(k)}, m_2^{(k)}, .., m_i^{(k)}\right\} \in M'^{(k)}, \tag{9}$$

where $m_i^{(k)}$ represents the i-th row vector of the matrix $M'^{(k)}$.

Global summary representation $q^{(k)}$ is computed through a Readout function $\mathbb{R}^{n\times d} \to \mathbb{R}^d$ as follows:

$$q^{(k)} = Readout\left(M'^{(k)}\right) = Sigmoid\left(\frac{1}{N}\sum_{i=1}^{N} m_i^{(k)}\right). \tag{10}$$

Note that different pooling methods (e.g., SAGPool [38]) can also be applied in lieu of the Readout function.

Then, MI is maxmized for positive sample pairs (i.e., $m_i^{(k)}$ and $q^{(k)}$) and negative sample pairs (i.e., $\widetilde{m}_j^{(k)}$ and $q^{(k)}$). $m_i^{(k)}$ and $\widetilde{m}_j^{(k)}$ are generated in the initial network and corrupted network, respectively. We compute the binary cross entropy loss function between the local patches and global summary representation as follows:

$$L^{(k)} = \sum_{j=1}^{N} \log\left(1 - D\left(\widetilde{m}_j^{(k)}, q^{(k)}\right)\right) + \sum_{v_i \in V}^{N} \log D\left(m_i^{(k)}, q^{(k)}\right), \tag{11}$$

where discriminator $D : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ obtains the score of a patch summary pair, such as $\left(m_i^{(k)}, q^{(k)}\right)$. We employ a simple pattern (i.e., a bilinear scoring function) in the experiment:

$$D\left(m_i^{(k)}, q^{(k)}\right) = Sigmoid\left(m_i^{(k)T} J^{(k)} q^{(k)}\right), \tag{12}$$

where $J^{(k)} \in \mathbb{R}^{d\times d}$ is a scoring matrix that is shared among all relation types $k \in K$, (i.e., $J = J^{(1)} = \ldots = J^{(K)}$).

4.3. Consensus regularization

The embedding matrices of different relation types are based on node neighborhood information to achieve mutual helping by consensus regularization.

More precisely, all relation-type matrices can achieve consensus after the introduction of consensus matrix $Z \in \mathbb{R}^{n\times d}$. One regularizer minimizes the disagreement between the set of relation-type matrices $\left(M'^{(k)}, k \in K\right)$ and the consensus matrix Z in the initial network; another regularizer maximizes the disagreement between the set of relation-type matrices $\left(\widetilde{M}'^{(k)}, k \in K\right)$ and the consensus matrix Z in the corrupted network. The equation is as follows:

$$L_{cr} = \left(Z - \Omega\left(\left(M'^{(k)}, k \in K\right)\right)^2 - \left(Z - \Omega\left(\left(\widetilde{M}'^{(k)}, k \in K\right)\right)\right)^2. \tag{13}$$

Ω indicates that the aggregation function can combine the set of the multiple relation-type matrices into a single matrix. We employ a simple pattern to calculate a set of multiple relation-type matrices to improve model efficiency:

$$M = \Omega\left(M'^{(k)}\right) = \frac{1}{|K|}\sum_{k\in K} M'^{(k)}. \tag{14}$$

4.4. Optimization

We employ Adam optimizer to optimize the model, and calculate the sum of the loss values jointly with the relation-type loss in Equation (11) and the consensus regularization loss in Equation (13) to obtain objective function:

$$C = \sum_{k\in K} L^{(k)} + \lambda L_{cr} + \mu||\Psi||^2, \tag{15}$$

where λ adjusts the importance degree of consensus regularization and μ is a trainable parameter set that controls $L2$ regularization on $\Psi = \left\{W^{(k)}, J, Z\right\}$. We sumarize APPTE in Algorithm 1.

Algorithm 1 APPTE Algorithm

```
Require: A multiplex network $G = \{V, \xi, \mathrm{X}\}$, adjacency matrix $A = \left\{A^{(1)}, \ldots, A^{(|K|)}\right\}$;
the metapath $P = \{P_1, \ldots, P_{n+1}\}$.
Ensure: Training the APPTE model.
  for $P_i \in P$ do
    for $k = 1 \ldots K$ do
      //From initial information to corrupted information;
      $\widetilde{X}^{(k)} \leftarrow X^{(k)}$;
      //Generating each type node embedding matrix;
      $h^{(k)} \leftarrow ReLU\left(\overline{D}_k^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}_k^{-\frac{1}{2}}XW^{(k)}\right)$;
      $\widetilde{h}^{(k)} \leftarrow ReLU\left(\overline{D}_k^{-\frac{1}{2}}\overline{A}^{(k)}\overline{D}_k^{-\frac{1}{2}}\widetilde{X}W^{(k)}\right)$;
      //ThrowEdge;
      $A_{throw} \leftarrow A - A'$;
      //Fully personalized PageRank;
      $\Pi_{ppr}^{(k)} \leftarrow \alpha\left(I_n - (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}_{throw}\overline{D}^{-\frac{1}{2}}\right)^{-1}$;
      //Approximate personalized propagation;
      for $n = 1 \ldots N$ do
        $m_{(0)}^{(k)} \leftarrow h^{(k)}$;
        $m_{(n)}^{(k)} \leftarrow (1-\alpha)\overline{D}^{-\frac{1}{2}}\overline{A}_{throw}^{(k)}\overline{D}^{-\frac{1}{2}}M_{(n-1)}^{(k)} + \alpha h^{(k)}$;
        $m'^{(k)} \leftarrow m_{(N)}^{(k)}$;
        $\widetilde{m}'^{(k)} \leftarrow \widetilde{m}_{(N)}^{(k)}$;
      end for
      //Relation-type fusion;
      $M'^{(k)} \leftarrow \{m^{(1)}, m^{(2)}, .., m^{(k)}\}$
      //Mutual information;
      $\left\{m_1^{(k)}, .., m_i^{(k)}\right\} \in M'^{(k)}$;
      $\left\{\widetilde{m}_1^{(k)}, .., \widetilde{m}_i^{(k)}\right\} \in \widetilde{M}'^{(k)}$;
      $q^{(k)} \leftarrow Readout\left(M'^{(k)}\right)$;
      Calculate maximizing the average MI with the cross entropy loss $L^{(k)}$;
      //Calculate consensus regularization;
      $L_{cr} \leftarrow \left(Z - \Omega\left(M'^{(k)}\right)\right)^2 - \left(Z - \Omega\left(\widetilde{M}'^{(k)}\right)\right)^2$;
    end for
    //Objective function;
    $C \leftarrow \sum_{k \in K} L^{(k)} + \lambda L_{cr} + \mu||\Psi||^2$;
end for
return
```

5. Experimental evaluation

In this section, our APPTE method is compared with other baseline methods on three benchmark datasets.

5.1. Datasets

We describe the datasets (i.e., ACM, DBLP and IMDB) in detail and summarize them in Table 1.

ACM. This dataset contains data from Knowledge Discovery and Data Mining (KDD), MobiCOMM and other journals. It contains 3025 papers (P), 5835 authors (A), and 56 subjects (S), corresponding to the relation type representation in Table 1; these papers are classified as databases, wireless communications, and data mining; the published papers are labeled. We choose the metapath set as $P = \{PAP, PSP\}$ in our experiment.

DBLP. This dataset contains 14328 papers (P), 4057 authors (A), 8789 terms (T) and 20 conferences (C), corresponding to the relation type representation in Table 1. The authors' research fields are labeled based on conferences they attended and

Table 1
Statistics of the datasets in the experiments.

Dataset	Relations(A-B)	Type A	Type B	Relations	Relation type	Node feature	Labeled data
ACM	Paper-Author	3025	5835	9744	PAP	1830	600
	Paper- Subject	3025	56	3025	PSP		
DBLP	Paper-Author	14328	4057	19645	APA	334	800
	Paper-Conf	14328	20	14328	APCPA		
	Paper-Term	14327	8789	88420	APTPA		
IMDB	Movie-Actor	3550	4441	10650	MAM	1007	300
	Movie-Director	3550	1726	3550	MDM		

are classified into databases, data mining, machine learning, and information retrieval. We choose the metapath set as $P = \{APA, APCPA, APTPA\}$ in our experiment.

IMDB. This dataset contains 3550 movies (M), 4441 actors (A) and 1726 directors (D), corresponding to the relation type representation in Table 1. Based on movie genre, they are divided into action, comedy, and drama. We choose the metapath set $P = \{MAM, MDM\}$ in our experiment.

5.2. Implementation details

5.2.1. Hyperparameters

We employ a random process to divide the dataset into a training set, validation set and test set. We set the learning rate to 0.005 and set the number of hidden dimension to 64. The throwedge rate f is 0.1. The teleport probability and power iteration steps are set to $\alpha = 0.6$ and $N = 10$, respectively. The consensus regularization coefficient λ is 0.001, the $L2$ regularization coefficient μ is 0.0001. Early stopping algorithms are applied to our model, and we set the patience threshold to 80 (e.g., when the loss value does not change for 80 consecutive epochs, the model stops training).

5.2.2. Evaluation metrics

We adopt three classical performance evaluation indicators (i.e., node classification, node clustering and similarity search) to evaluate the different baseline methods. For node classification, we calculate Macro-F1 and Micro-F1 on the test set; the logistic regression classifier is trained by learning embeddings on the training set for node evaluation on the test set. For the similarity search, we calculate the cosine similarity scores of the node embeddings among all node pairs, and rank the nodes based on the similarity score of each node. Then, the ratio of the top-5 nodes that belong to the same class is calculated and is called Sim@5. For node clustering, we adopt the k-means algorithm and evaluate the clustering results through normalized MI (NMI).

5.3. Comparision of the baseline methods

We compare some classic methods in graph representation learning that contain the state-of-art benchmarks to illustrate the effectiveness of our proposed APPTE method. We conduct our experiments on NVIDIA TITAN RTX GPU card with 24 GB memory.

5.3.1. Random walk-based methods

DeepWalk [39]. A truncated random walk method obtains the local information of homogeneous graphs based on the network embedding method. However, we perform DeepWalk on heterogeneous graphs without considering the heterogeneity of the nodes.

Metapath2vec [37]. This method constructs node neighborhoods on heterogeneous graphs by a random walk method, and then simultaneously models both the structural and semantic correlations in heterogeneous networks.

HERec [40]. This method employs a random walk method to generate node sequences on heterogeneous graphs. Node embeddings are converted by a fusion function and then transferred to extended matrix factorization.

5.3.2. Graph neural network-based methods

GCN [22]. A method which encodes local structure and node features to learn graph representations. We test the GCN and report its performance.

GAT [41]. A method which employs masked self-attentional layers in which a node participates in its neighborhoods' features and then assigns different weights to different nodes in the neighborhoods.

DGI [18]. An unsupervised method which learns node representations based on maximizing MI between local patches and graph summaries.

HAN [17]. A method which applies an attention mechanism to heterogeneous graphs and considers node level and semantic-level attention.

Table 2
Baseline comparison on node classification tasks.

Method	ACM		DBLP		IMDB	
	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1
DeepWalk	72.63%	73.56%	76.71%	78.44%	51.21%	53.68%
Metapath2vec	68.17%	68.24%	88.13%	88.62%	52.53%	54.18%
HERec	69.28%	69.12%	89.67%	90.02%	52.68%	54.53%
GCN	85.79%	86.01%	90.06%	90.87%	59.24%	60.09%
GAT	85.62%	85.97%	90.71%	91.14%	59.81%	60.47%
DGI	86.53%	86.55%	88.37%	88.95%	60.37%	60.72%
HAN	88.29%	88.43%	91.88%	92.45%	61.99%	62.16%
MAGNN	89.46%	89.69%	91.95%	92.58%	62.91%	63.23%
HGT	88.52%	88.77%	91.74%	92.33%	62.38%	62.59%
DMGI	88.91%	88.92%	91.51%	92.27%	63.56%	63.87%
APPTE	**90.26%**	**90.25%**	**92.25%**	**92.99%**	**65.20%**	**65.22%**

Table 3
Baseline comparison on the similarity search and node clustering tasks.

Method	ACM		DBLP		IMDB	
	Sim@5	NMI	Sim@5	NMI	Sim@5	NMI
DeepWalk	69.59%	30.42%	75.66%	68.41%	49.21%	11.21%
Metapath2vec	66.97%	30.89%	67.43%	67.43%	49.53%	12.46%
HERec	67.28%	40.11%	86.32%	68.52%	50.01%	12.98%
GCN	84.59%	52.87%	86.89%	68.01%	56.96%	16.51%
GAT	85.12%	56.92%	87.04%	66.37%	57.19%	17.06%
DGI	86.53%	58.88%	88.95%	71.33%	57.88%	17.91%
HAN	87.89%	60.41%	89.33%	73.88%	59.07%	17.24%
MAGNN	88.81%	62.23%	90.54%	74.57%	59.85%	18.68%
HGT	88.26%	61.29%	90.41%	74.00%	59.51%	18.10%
DMGI	88.53%	61.78%	90.27%	74.05%	60.33%	19.33%
APPTE	**89.21%**	**63.31%**	**90.79%**	**75.18%**	**61.31%**	**20.18%**

MAGNN [42]. This method establishes node content transformation and multiple metapath aggregation to produce node embedding of heterogenous graph.

HGT [43]. This method utilizes a transformer mechanism to design a dedicated representation for nodes and edges in heterogenous graphs.

DMGI [19]. A method which minimizes disagreements among the node embeddings of relation types and distinguishes the real samples regardless of relation type. We test DMGI and report its performance.

5.4. Performance evaluation and analysis

In Tables 2 and 3, we compare the performance of different baseline methods. Our proposed APPTE is superior to the state-of-art benchmarks of different datasets. APPTE is an unsupervised method that does not require any labeled data.

We observe that the experiment results of the graph neural network-based methods (i.e., GCN, GAT, DGI, HAN, and DMGI) are generally better than those of the random walk-based methods (i.e., DeepWalk, Metapath2vec and HERec).

Random walk-based methods employ generated sequences to define the proximity relationship between nodes. With enough sampling, these methods can be used to well describe the proximity information between nodes. However, they do not aggregate node features and neighborhood information.

We conducted an in-depth analysis of graph neural network-based methods. The GCN can aggregate a node's features and its neighborhood information and can construct a multilayer network. However, it does not calculate the importance of neighboring nodes. GAT introduces a self-attention mechanism and utilizes the current node features and neighboring node features to obtain the importance of the neighboring nodes. However, it is not applied to heterogeneous graphs.

HAN applies an attention mechanism to heterogeneous graphs and establishes node level and semantic-level attention. However, MI is not considered in this method. MAGNN establishes node content transformation and multiple metapath aggregation to produce node embedding of heterogenous graph but fails to consider the robustness and generalization ability. HGT utilizes transformer mechanism to design a dedicated representation for nodes and edges in heterogenous graphs but fails to consider the consensus regularization. DGI learns node representations through MI between local patches and graph summaries, but it is limited to a single network. DMGI introduces a consensus regularization framework and universal discriminator on multiplex network embeddings to learn *d*-dimensional vector representations, but the limited node propagation range leads to insufficient utilization of node neighborhood information.

Our proposed APPTE explores more neighborhood information by expanding the node propagation range and randomly deletes a part of edges to obtain better robustness and generalization ability so that the unsupervised embeddings in heterogeneous graphs are achieved. On the ACM dataset, we remove 2525 edges; on the DBLP dataset, we remove 498891 edges; on the IMDB dataset, we remove 6597 edges. Therefore, APPTE obtains better performance, as shown in bold in Tables 2 and 3.

5.5. The performance of our proposed APPTE method

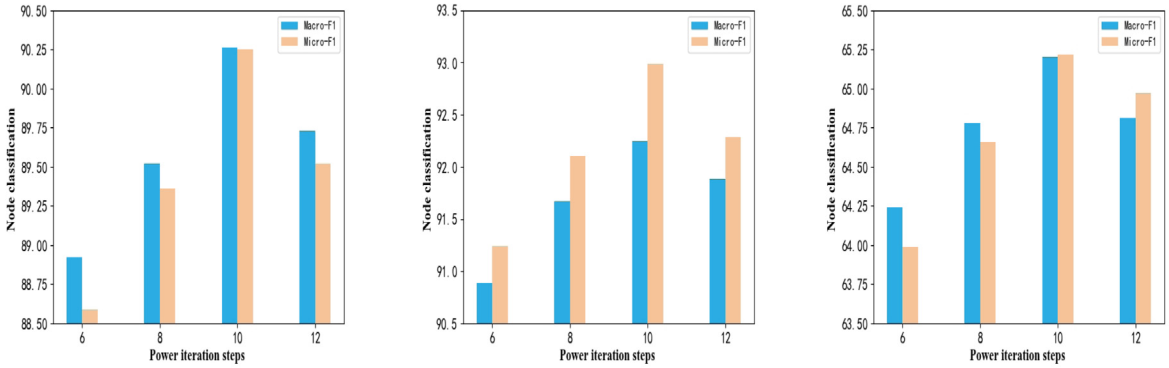
We evaluate the performance of APPTE on the datasets (i.e., ACM, DBLP and IMDB) through different numbers of power iterations steps and epochs values.

Under the different power iteration steps, Fig. 3 shows the evaluation indicators of Macro-F1 and Micro-F1 for node classification. We observe that before the power iteration step $N = 10$, the performance of APPTE increases as the propagation range expands, but there is a slight decrease when $N = 12$. Fig. 4 shows the evaluation indicators of NMI for node clustering. We can clearly see that the best results are achieved in the power iteration steps $N = 10$; the other results are also good. According to Figures 3 and 4, APPTE proves that expanding the node neighborhood range can improve the performance of node classification and node clustering.

Under the different epochs values, we observe the performance changes. Fig. 5 shows the changes in the Macro-F1 and Micro-F1 scores. We can clearly see that the ACM dataset and IMDB dataset reach convergence in 2000 epochs; the DBLP dataset reaches convergence in 3000 epochs. Fig. 6 shows the situational changes in NMI, and then we find that the ACM, DBLP and IMDB datasets converge in 2000 epochs. Based on Figs. 5 and 6, APPTE converges with fewer epochs than the state-of-art benchmark DMGI, which converges in 10000 epochs.

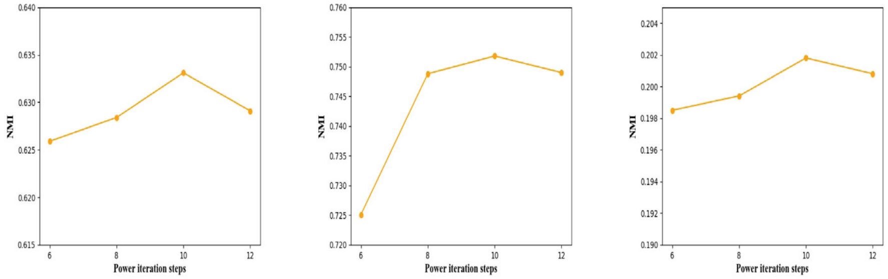
5.6. Ablation study

To evaluate robustness and generalization ability in APPTE, we employ ablation studies of multiple groups on the DBLP dataset for sensitivity analysis. According to different throwedge rates (i.e., $f = 0.1$ and $f = 0.2$), we set 3 random seeds to conduct performance tests shown as Table 4. When the throwedge is $f = 0.1$, we observe that APPTE's performance has only changed slightly with different random seeds, which indicates that our framework has good robustness and generalization



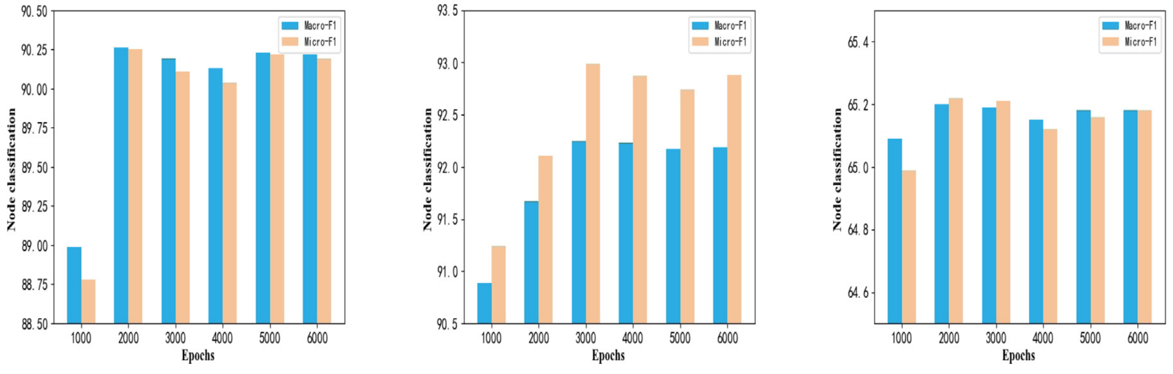
(a) ACM dataset (b) DBLP dataset (c) IMDB dataset

Fig. 3. Different power iteration steps for node classification.



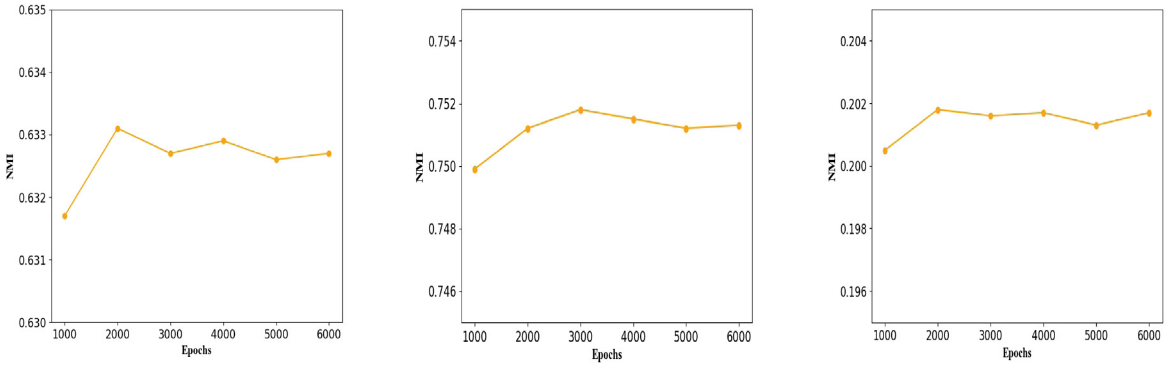
(a) ACM dataset (b) DBLP dataset (c) IMDB dataset

Fig. 4. Different power iteration steps for NMI.



(a) ACM dataset (b) DBLP dataset (c) IMDB dataset

Fig. 5. Different epochs for node classification.



(a) ACM dataset (b) DBLP dataset (c) IMDB dataset

Fig. 6. Different epochs for NMI.

Table 4
Performance comparison on node classification, similarity search and node clustering.

Throwedge Rate	Random Seed	DBLP			
		Macro-F1	Micro-F1	Sim@5	NMI
f = 0.1	1	92.26%	93.02%	90.78%	75.11%
	2	92.22%	92.97%	90.75%	75.05%
	3	92.21%	92.98%	90.77%	75.07%
f = 0.2	1	91.09%	91.88%	89.19%	73.89%
	2	91.08%	91.86%	89.15%	73.84%
	3	91.04%	91.89%	89.09%	73.86%

ability in small variance. Compared with a throwedge of $f = 0.1$, performance degrades when throwedge the is $f = 0.2$, indicating that a large amount of loss of edge information leads to the insufficient extraction of key information by the model.

We also construct different modules to illustrate their performance, as shown in Table 5. The first group of ablation experiments (i.e., APPTE-noTE) considers approximate personalized propagation, consensus regularization and MI, but not throwedge. The second group of ablation experiments (i.e., APPTE-noAPP) considers throwedge, consensus regularization and mutual information, but not approximate personalized propagation. The third group of ablation experiment (i.e., APPTE-noCR) is throwedge, approximate personalized propagation and mutual information, but not consensus regularization. From Table 5, we find that the performance of APP-noTE is slightly lower than that of APPTE because throwedge values in a reasonable range reduces the convergence speed of over-smoothing to reduce information loss. The performance of APPTE-noCR is lower than APP-noTE because consensus regularization aggregates the information of the initial and corrupted. Compared

Table 5
Performance of different modules on DBLP.

Method	DBLP			
	Macro-F1	Micro-F1	Sim@5	NMI
APPTE	92.25%	92.99%	90.79%	75.18%
APPTE-noTE	92.01%	92.69%	90.54%	74.85%
APPTE-noCR	91.79%	92.47%	90.30%	74.53%
APPTE-noAPP	91.63%	92.34%	90.18%	74.21%

with APPTE, we observe that the performance of APPTE-noAPP is significantly lower, indicating that the node neighborhood information has a greater impact on the model.

6. Conclusions

In this paper, we analyze and tackle the problem that limits the propagation range of node neighborhoods in heterogeneous graphs. Our APPTE framework is an unsupervised method that has an end-to-end structure to achieve multiple downstream tasks (i.e., node classification, similarity searches and node clustering). We construct model to adequate node neighborhood information in local context, and captures the global neighborhood information. Meanwhile, our method deletes a part of edges to increase the randomness and diversity of the graph connections so that the model obtains better robustness and generalization ability. The experimental results demonstrate that APPTE is superior to several state-of-the-art baseline methods.

In future work, we will explore more effective propagation schemes for more complex practical applications, such as drug-drug interactions (DDIs).

CRediT authorship contribution statement

Yibi Chen: Conceptualization, Methodology, Validation, Writing – review & editing. **Yikun Hu:** Methodology, Validation, Writing – review & editing. **Keqin Li:** Supervision, Software. **Chai Kiat Yeo:** Formal analysis, Writing – review & editing. **Kenli Li:** Supervision, Formal analysis, Writing – review & editing.

Declaration of Competing Interest

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

Acknowledgments

The research was partially funded by the Key Program of the National Natural Science Foundation of China (Grant Nos. 61133005, 61432005) and National Natural Science Foundation of China (Grant Nos. 62172151, 61876061).

References

[1] Q. Lin, J. Liu, Y. Pan, L. Zhang, X. Hu, J. Ma, Rule-enhanced iterative complementation for knowledge graph reasoning, Inf. Sci..
[2] G. Yang, Y. Kang, X. Zhu, C. Zhu, G. Xiao, Info2vec: an aggregative representation method in multi-layer and heterogeneous networks, Inf. Sci..
[3] L. Jiao, R. Zhang, F. Liu, S. Yang, B. Hou, L. Li, X. Tang, New generation deep learning for video object detection: A survey, IEEE Trans. Neural Networks Learn. Syst..
[4] S. Zhou, Q. Ou, X. Liu, S. Wang, L. Liu, S. Wang, E. Zhu, J. Yin, X. Xu, Multiple kernel clustering with compressed subspace alignment, IEEE Trans. Neural Networks Learn. Syst..
[5] S. Zhou, X. Liu, M. Li, E. Zhu, L. Liu, C. Zhang, J. Yin, Multiple kernel clustering with neighbor-kernel subspace segmentation, IEEE Trans. Neural Networks Learn. Syst. 31 (4) (2019) 1351–1362.
[6] U.S. Shanthamallu, J.J. Thiagarajan, H. Song, A. Spanias, Gramme: Semisupervised learning using multilayered graph attention models, IEEE Trans. Neural Networks Learn. Syst. 31 (10) (2019) 3977–3988.
[7] M. Shi, Y. Tang, X. Zhu, Mlne: Multi-label network embedding, IEEE Trans. Neural Networks Learn. Syst. 31 (9) (2019) 3682–3695.
[8] Y. Chen, X. Zou, K. Li, K. Li, X. Yang, C. Chen, Multiple local 3d cnns for region-based prediction in smart cities, Inf. Sci. 542 (2021) 476–491.
[9] S. Min, Z. Gao, J. Peng, L. Wang, K. Qin, B. Fang, Stgsn-a spatial-temporal graph neural network framework for time-evolving social networks, Knowl.-Based Syst. 106746 (2021).
[10] G. Cui, J. Zhou, C. Yang, Z. Liu, Adaptive graph encoder for attributed graph embedding, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 976–985.
[11] Z. Sun, C. Wang, W. Hu, M. Chen, J. Dai, W. Zhang, Y. Qu, Knowledge graph alignment network with gated multi-hop neighborhood aggregation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 222–229..
[12] X. Zou, K. Li, C. Chen, Multi-level attention based u-shape graph neural network for point clouds learning, IEEE Trans. Ind. Inf..
[13] D. Zhou, S. Zhang, M.Y. Yildirim, S. Alcorn, H. Tong, H. Davulcu, J. He, High-order structure exploration on massive graphs: A local graph clustering perspective, ACM Transactions on Knowledge Discovery from Data (TKDD) 15 (2) (2021) 1–26.
[14] S. Guo, Y. Lin, H. Wan, X. Li, G. Cong, Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting, IEEE Trans. Knowl. Data Eng. 01 (2021) 1.

[15] A. Sankar, X. Zhang, K.C.-C. Chang, Motif-based convolutional neural network on graphs, arXiv preprint arXiv:1711.05697..

[16] H. Chen, H. Yin, W. Wang, H. Wang, Q.V.H. Nguyen, X. Li, Pme: projected metric embedding on heterogeneous networks for link prediction, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1177–1186.

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

[18] P. Velickovic, W. Fedus, W.L. Hamilton, P. Liò, Y. Bengio, R.D. Hjelm, Deep graph infomax..

[19] C. Park, D. Kim, J. Han, H. Yu, Unsupervised attributed multiplex network embedding, AAAI (2020) 5371–5378.

[20] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, arXiv preprint arXiv:1312.6203..

[21] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Advances in neural information processing systems, 2016, pp. 3844–3852..

[22] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint arXiv:1609.02907..

[23] C. Zhuang, Q. Ma, Dual graph convolutional networks for graph-based semi-supervised classification, in: Proceedings of the 2018 World Wide Web Conference, 2018, pp. 499–508.

[24] Y.-N. Guo, X. Zhang, D.-W. Gong, Z. Zhang, J.-J. Yang, Novel interactive preference-based multiobjective evolutionary optimization for bolt supporting networks, IEEE Trans. Evol. Comput. 24 (4) (2019) 750–764.

[25] Y. Rong, W. Huang, T. Xu, J. Huang, Dropedge: Towards deep graph convolutional networks on node classification, arXiv preprint arXiv:1907.10903..

[26] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in neural information processing systems, 2017, pp. 1024–1034..

[27] Q. Li, Z. Han, X.-M. Wu, Deeper insights into graph convolutional networks for semi-supervised learning, arXiv preprint arXiv:1801.07606..

[28] R. Ying, R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 974–983.

[29] K. Xu, C. Li, Y. Tian, T. Sonobe, K.-I. Kawarabayashi, S. Jegelka, Representation learning on graphs with jumping knowledge networks, arXiv preprint arXiv:1806.03536..

[30] T. Kawamoto, M. Tsubaki, T. Obuchi, Mean-field theory of graph neural networks in graph partitioning, J. Stat. Mech: Theory Exp. 2019 (12) (2019) 124007.

[31] Y. Guo, H. Yang, M. Chen, J. Cheng, D. Gong, Ensemble prediction-based dynamic robust multi-objective optimization methods, Swarm Evol. Comput. 48 (2019) 156–171.

[32] Z. Chen, X. Li, J. Bruna, Supervised community detection with line graph neural networks, arXiv preprint arXiv:1705.08415..

[33] J. Klicpera, A. Bojchevski, S. Günnemann, Predict then propagate: Graph neural networks meet personalized pagerank, arXiv preprint arXiv:1810.05997..

[34] H. Zhang, L. Qiu, L. Yi, Y. Song, Scalable multiplex network embedding., in: IJCAI, Vol. 18, 2018, pp. 3082–3088..

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

[36] Y. Ma, Z. Ren, Z. Jiang, J. Tang, D. Yin, Multi-dimensional network embedding with hierarchical structure, in: Proceedings of the eleventh ACM international conference on web search and data mining, 2018, pp. 387–395.

[37] Y. Dong, N.V. Chawla, A. Swami, metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, 2017, pp. 135–144.

[38] J. Lee, I. Lee, J. Kang, Self-attention graph pooling, arXiv preprint arXiv:1904.08082..

[39] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014, pp. 701–710.

[40] C. Shi, B. Hu, W.X. Zhao, S.Y. Philip, Heterogeneous information network embedding for recommendation, IEEE Trans. Knowl. Data Eng. 31 (2) (2018) 357–370.

[41] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint arXiv:1710.10903..

[42] X. Fu, J. Zhang, Z. Meng, I. King, Magnn: Metapath aggregated graph neural network for heterogeneous graph embedding, Proceedings of The Web Conference 2020 (2020) 2331–2341.

[43] Z. Hu, Y. Dong, K. Wang, Y. Sun, Heterogeneous graph transformer, Proceedings of The Web Conference 2020 (2020) 2704–2710.