Hindawi
Computational Intelligence and Neuroscience
Volume 2019, Article ID 8106073, 15 pages
https://doi.org/10.1155/2019/8106073

Research Article

Context Attention Heterogeneous Network Embedding

Wei Zhuo,[1,2] Qianyi Zhan,[1,2] Yuan Liu,[1,2] Zhenping Xie,[1,2] and Jing Lu[1,2]

[1]*School of Digital Media, Jiangnan University, Wuxi 214122, China*
[2]*Jiangsu Key Laboratory of Media Design and Software Technology, Wuxi 214122, China*

Correspondence should be addressed to Yuan Liu; lyuan1800@jiangnan.edu.cn

Received 9 April 2019; Revised 11 July 2019; Accepted 17 July 2019; Published 21 August 2019

Academic Editor: Juan Carlos Fernández

Network embedding (NE), which maps nodes into a low-dimensional latent Euclidean space to represent effective features of each node in the network, has obtained considerable attention in recent years. Many popular NE methods, such as DeepWalk, Node2vec, and LINE, are capable of handling homogeneous networks. However, nodes are always fully accompanied by heterogeneous information (e.g., text descriptions, node properties, and hashtags) in the real-world network, which remains a great challenge to jointly project the topological structure and different types of information into the fixed-dimensional embedding space due to heterogeneity. Besides, in the unweighted network, how to quantify the strength of edges (tightness of connections between nodes) accurately is also a difficulty faced by existing methods. To bridge the gap, in this paper, we propose CAHNE (context attention heterogeneous network embedding), a novel network embedding method, to accurately determine the learning result. Specifically, we propose the concept of node importance to measure the strength of edges, which can better preserve the context relations of a node in unweighted networks. Moreover, text information is a widely ubiquitous feature in real-world networks, e.g., online social networks and citation networks. On account of the sophisticated interactions between the network structure and text features of nodes, CAHNE learns context embeddings for nodes by introducing the context node sequence, and the attention mechanism is also integrated into our model to better reflect the impact of context nodes on the current node. To corroborate the efficacy of CAHNE, we apply our method and various baseline methods on several real-world datasets. The experimental results show that CAHNE achieves higher quality compared to a number of state-of-the-art network embedding methods on the tasks of network reconstruction, link prediction, node classification, and visualization.

1. Introduction

Nowadays, information networks are ubiquitous in our daily life, for example, social and communication networks, citation networks, and co-occurrence networks. At most of the time, the scales of real-world networks are very large. Thus, analyzing large-scale networks has attracted considerable research attention in recent years. Network embedding (NE), also known as network representation learning, aims to generate informative numerical representations for nodes in the network to preserve network structures and further alleviates the inconveniences caused by sparsity. Network embedding methods are demonstrated to be effective in many network analysis tasks including link prediction [1], node classification [2], and clustering [3].

Many approaches have been proposed toward this goal, such as DeepWalk [4], LINE [5], Node2vec [6], and PPNE [7]. Particularly, network embedding aims to project the network into a low-dimensional space, where each node is represented using a corresponding embedding vector, and the relativity among nodes is preserved. The nodes with "high similarity" are mapped onto adjacent points ("high similarity" means nodes have similar properties and are more likely to have edges between them). The embedding vectors contain the semantic information transcribed from the network structure and can be applied in various network mining applications easily. However, most of the existing NE methods take the network structure as input to learn representations for nodes without considering any other information.

In reality, a network usually has rich heterogeneous information, such as text descriptions and other metadata. For instance, Wikipedia (https://www.wikipedia.org/) entries connect with each other and build an encyclopedia network. Simultaneously, each entry as a node has substantial text information such as keywords and introduction, which describe a node in detail and more comprehensively. Furthermore, in the real-world social network like Twitter (https://twitter.com) shown in Figure 1, users as nodes also have their own text descriptions, which may reflect the properties of each node. Hence, text information is typical and critical heterogeneous semantic information widely existing in real-world networks. However, most NE models treat all networks as homogeneous networks. In other words, most works learn representations only from network structures ignoring text information. Because of heterogeneity in networks, we put forward an idea to embed a network from both network structures and text information.

To this end, a direct way is to learn representations from text information of nodes and network structures independently, which can be called text-aware embedding. However, this way ignores the complicated interactions between network structures and text information, which leads to invalidity. CANE [8] is an efficient method to capture the correlation between the text feature of a node and its neighbors' in a network, which achieves the purpose we stated before. However, CANE only preserves the local relations in a network, while we need to take the global network structures into consideration rather than node pairs independently. For example, in Figure 1, Bob may have connections to other NLP researchers who are also his colleagues and Alice has not followed these researchers, so there may be potential relationships between these researchers and Alice in the text aspect because they have similar properties, but CANE cannot capture these relationships. Thus, how to satisfy the compatibility between network structures and text information in the network should be exploited to better represent nodes.

In addition to the problem stated above, typical NE methods are insensitive to the strength of the relationship between nodes in unweighted networks. As an intuitive example, we show some relationships from the real-world online networks in Figure 1. In Twitter, Trump is a celebrity who has plenty of followers, and each follower links to him by an edge. Alice and Bob are ordinary users, and they link with each other because they are colleagues. They also follow Trump just because they are Americans. In this case, the strength of the relationship between Alice and Bob should be stronger than that between Alice and Trump. As shown in Figure 1, we use dotted lines and solid lines to describe the strength of relationships (edges). Strong connection means high similarity between pairwise nodes, and weak connection means low similarity. In unweighted networks, classical NE methods generally treat the weight of the edge between nodes as a binary variable and ignore the rich semantics of edges we illustrated before. Therefore, the strength of connections is underlying structural information we need to take into consideration when learning network representations in real-world networks, which remains a great challenge.

From the aforementioned problems, the heterogeneity and structural complexity in real-world networks pose specific hurdles for network representation learning. Fortunately, in this paper, we propose a context attention heterogeneous network embedding (CAHNE) method with an emphasis on leveraging the rich and intrinsic information in heterogeneous networks. Specifically, CAHNE reconstructs the classical network represented as $G = (V, E)$ to form the heterogeneous text network denoted as $G = (V, E, T)$. We can extract a context node sequence for each node by breadth-first search (BFS) on the redesigned network, and the root node can be deemed the anchor node. Through a series of specific operations that we will give a detailed elaboration in the later section, combining the text information in a sequence, we can obtain a representation for the anchor of the context node sequence, which is the context embedding of the anchor node. Therefore, CAHNE integrates text information into the global structures of the network to learn the potential intertextual associations in the network. Moreover, the influence of context nodes on the anchor node can vary with different anchor nodes, and thus, we further the adopt attention mechanism to enhance the expressiveness of the influence from the context nodes on the specific anchor node. Besides, for unweighted networks, CAHNE is expected to preserve the underlying structural information on the strength of edges. Based on this idea, we give the definition of node importance that quantifies the strength of the relationship between nodes and integrate it into the network embedding method to learn a structure-based representation for each node. Finally, we concatenate the context embedding and the structure-based embedding of the node as the complete representation for the node. Empirically, we apply CAHNE to four network analysis tasks, i.e., network reconstruction, link prediction, node classification, and visualization, using seven real-world networks as datasets. Experimental results demonstrate that our method learns better nodes embeddings when compared to a variety of state-of-the-art baselines in the field of NE.

The main contributions of our method are summarized as follows:

(i) We propose a novel network embedding model, namely, CAHNE. The method is able to learn comprehensive representations for different types of real-world networks, which confirms the flexibility and robustness of our model.

(ii) We provide a key insight regarding the strength of relationships in unweighted real-world networks. We thereby propose the definition of node importance for optimizing the objective, which more closely shows the actual situations of the network.

(iii) We integrate heterogeneous information into network representation and mitigate the incompatibility between network structures and text information by extracting context node sequences accompanied by the attention mechanism to learn context embeddings.

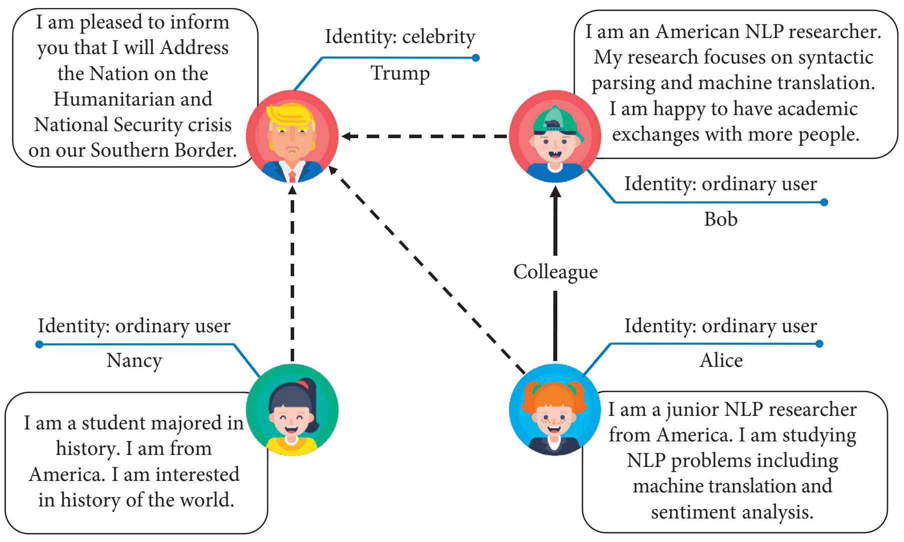

I am pleased to inform you that I will Address the Nation on the Humanitarian and National Security crisis on our Southern Border.
Identity: celebrity
Trump
I am an American NLP researcher. My research focuses on syntactic parsing and machine translation. I am happy to have academic exchanges with more people.
Identity: ordinary user
Bob
Colleague
Identity: ordinary user
Nancy
I am a student majored in history. I am from America. I am interested in history of the world.
Identity: ordinary user
Alice
I am a junior NLP researcher from America. I am studying NLP problems including machine translation and sentiment analysis.

Figure 1: Example of relationships between users in Twitter and the content of their tweets. Dotted lines and solid lines represent weak connections and strong connections, respectively.

The source code is available at https://github.com/zhuo931077127/CAHNE.

2. Related Works

Network representation learning (NRL) has been well researched for many years, for example, in earlier works such as Isomap [9], multidimensional scaling (MDS) [10], and Laplacian eigenmap (LE) [11]. These approaches represent the network as an affinity graph by using the feature vectors of the network nodes. For a given large-scale information network, e.g., social network and citation network, these methods are less efficient and inflexible to generate node representations.

In recent years, inspired by the development of the machine learning and word embedding method Word2vec [12], many NRL methods have been proposed for large-scale information network representation. For example, DeepWalk [4] proposes to perform random walks on the graph to obtain sequences of nodes. It introduces the Skip-Gram model to achieve vertex representations. Based on DeepWalk, Node2vec [6] defines a flexible notion of a node's network neighborhood and designs a biased random walk procedure to explore the network structure more efficiently. Some other methods focus on finding multivariate structure features in the network. For example, LINE [5] embeds the network into a low-dimensional latent space to approximate the first-order proximity and second-order proximity of the network. Nevertheless, most of these network embedding models only focus on homogeneous networks, without taking heterogeneous information into consideration.

Different from homogeneous networks, heterogeneous networks consist of complex node and edge attributes. Several attempts have been done on heterogeneous information network (HIN) embedding and achieved promising performance in various tasks. Hin2Vec [13] learns the embeddings of a HIN by conducting multiple prediction training tasks jointly. CANE [8] learns network embeddings from network structures and text descriptions with mutual relations of pairwise nodes. ANRL [14] proposes a neighbor enhancement autoencoder to incorporate both the network structure and node attribute information in a principled way. Paper2vec [15] aims to learn the paper node embeddings from the paper citation network.

In summary, existing methods in homogeneous network embedding use either affinity matrix models or deep models to preserve network structural features in a low-dimensional space. And existing HIN embedding methods focus on different types of heterogeneous information. They have been proven useful on network analysis, but they cannot maintain the sophisticated interaction between network structures and heterogeneous information (in this paper, we consider text information). Additionally, to the best of our knowledge, all existing NE models ignore the important relationship information between nodes in unweighted real-world networks we proposed before. In contrast, our proposed model CAHNE can learn more comprehensive information than existing methods.

3. Preliminaries

In this section, we introduce basic definitions and formalize the problem of context attention heterogeneous network embedding.

3.1. Context Node Sequence (CNS). Forming a context node sequence for the anchor node in the network can be viewed as a sampling process of detecting nodes that most likely have impact on the anchor node. Figure 2 shows the process of obtaining a context node sequence. Concretely, we first perform breadth-first search (BFS) on the original graph G starting from a node $v_i \in V$, and we regard v_i as an anchor node, which provides us with a BFS tree x_i rooted at v_i. x_i can be considered the unique relational tree of v_i. Context nodes are not only the neighborhood of the anchor node but also

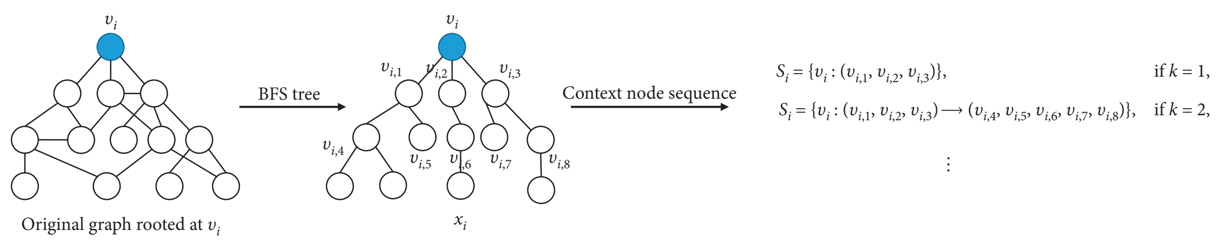

v_i
BFS tree
v_i
$v_{i,1}$
$v_{i,2}$
$v_{i,3}$
$v_{i,4}$
$v_{i,5}$
$v_{i,6}$
$v_{i,7}$
$v_{i,8}$
Context node sequence
$S_i = \{v_i : (v_{i,1}, v_{i,2}, v_{i,3})\}$, if $k = 1$,
$S_i = \{v_i : (v_{i,1}, v_{i,2}, v_{i,3}) \longrightarrow (v_{i,4}, v_{i,5}, v_{i,6}, v_{i,7}, v_{i,8})\}$, if $k = 2$,
$\vdots$
Original graph rooted at v_i
x_i

Figure 2: Example of the generating strategy for a context node sequence. The blue node is an anchor node v_c.

deeper layer nodes. Hence, we control the number of layers by setting the parameter k to sample context nodes. Furthermore, the value of k is uncertain and determined by the type of the given network. At last, for a given node v_i, we can obtain its context node sequence $S_i = \{v_i : (v_{i,1}, v_{i,2}, \cdots, v_{i,m}) \longrightarrow (v_{i,m+1}, \cdots, v_{i,m+n}) \longrightarrow \cdots\}$, where m and n are the number of context nodes in the first layer and second layer, respectively, and so on. v_i can also be treated as $v_{i,0}$. It is worth noting that each node can only appear once or 0 times in a context node sequence and building BFS trees for all nodes is not computationally expensive because of the sparsity of real-world networks.

3.2. Problem Formulation. Now, we formally define the problem of CAHNE. Compared to conventional homogeneous network embedding such as DeepWalk and Node2-vec, which only focus on a single network structure, our goal is to learn a representation for each node in a network graph with convergence of more heterogeneous associated information. Text information is widely available in real-world networks, e.g., social networks and citation networks, so we integrate it into the traditional graph definition ($G = (V, E)$) [16]. We first define a heterogeneous text network as follows.

Definition 1 (heterogeneous text network (HTN)). The HTN is denoted as $G = (V, E, T)$, where $V = \{v_1, \ldots, v_V\}$ represents the set of nodes, $E = \{e_{ij}\}_{i,j=1}^{|V|}$ represents the set of edges, and e_{ij} is the relationship between two nodes (v_i, v_j) linked with each other, with an associated weight w_{ij} (in this paper, we only consider unweighted networks). $T = \{t_1, t_2, \ldots, t_{|V|}\}$ denotes the text information of nodes. For the text information of a specific node $v_c \in V$, we can represent it as a word sequence $t_c = \{w_1, w_2, \ldots, w_{n_c}\}$, where $n_c = |t_c|$ denotes the number of words in t_c.

Noticing the difference between the definition of the heterogeneous text network $G = (V, E, T)$ and conventional network $G = (V, E)$, the heterogeneous text network contains richer information. Empirically, weight often indicates the strength of the edge between two nodes. In practice, for unweighted real-world network datasets, weights are only formed as binary variables. For example, if v_i has a neighbor v_j, the weight of the edge between them is 1; otherwise, it is 0. However, we expect to measure the strength of the relations more in line with the actual situations of real-world online networks. Thus, we propose the definition of node importance as follows.

Definition 2 (node importance). Node importance is denoted as NI, which is a quantitative representation for each node in the network. It measures the strength of the edge between a given node and its neighbors. For an anchor node $v_i \in V$, $\mathrm{NI}(v_i)$ is the value of node importance for v_i.

In real-world networks such as citation networks and social networks, each node has its own context node sequence. We can integrate all nodes' CNSs and get a global sequence for G, $S_G = (S_1, S_2, \ldots, S_{|V|})$. The more the CNSs a node consists of, in other words, the more the times a node appears in S_G, the less the importance for this node to its neighbors. For instance, in Twitter, a celebrity has thousands of followers, which means this celebrity consists of abundant CNSs. However, for ordinary users, the importance of the relationship with a celebrity is less than that with their real friends who have relationships with them.

Definition 3 (network embedding). Given a heterogeneous text network denoted as $G = (V, E, T)$, network embedding aims to map the network data into a low-dimensional latent space, where each node $v \in V$ can learn a low-dimensional embedding $v \in \mathbb{R}^d$ according to its graph structure and other information. Note that $d \ll |V|$ is the dimension of the latent embedding space.

Embedding a network into a low-dimensional space is helpful for many analysis tasks. In this process, the structures and properties of the network are preserved and encoded. In a heterogeneous text network, structure-based network embedding is not enough and the heterogeneous information is usually highly correlated with the network structure. Thus, we further propose the definition of context embedding.

Definition 4 (context embedding). Aiming to learn a vector representation for the text information of each node in an HTN, context embedding learns a mapping function $f : t_i \longrightarrow t_i \in \mathbb{R}^{d_c}$ for a node $v_i \in V$, where d_c is the dimension of context embedding.

It is worth mentioning that more than integrating text features of the anchor node, it also takes the context node sequence into consideration. For instance, the context embedding of the anchor node v_c is determined by its CNS S_c and its own text description t_c. In this paper, our method CAHNE introduces the attention mechanism to weight the context nodes for each anchor node so that we can mitigate the incompatibility between network topologies and text features to obtain more comprehensive and accurate representations for the network.

4. CAHNE: The Proposed Method

In this section, we will give a detailed introduction to our method CAHNE.

4.1. Overall Framework. For CAHNE, we need to take full use of network structures and associated text information. We propose two types of embedding for a node $v \in V$, i.e., structure-based embedding v^s and context embedding v^c. Structure-based embedding can capture the network structural information, which incorporates node importance, while context embedding can capture the textual meanings of anchor nodes accompanied by their context node sequences' text information. We concatenate two types of embeddings and obtain the overall node embedding for a node as follows:

$$v = v^s \oplus v^c, \tag{1}$$

where $\oplus$ indicates the concatenation operation. In the following sections, we will give a detailed introduction to the two types of embeddings, respectively.

4.2. Structure-Based Embedding. Without loss of universality, we assume the heterogeneous text network is directed. For the undirected network, we consider two directed edges with opposite directions and equal weights. And then, CAHNE fuses node importance as the weight for each node in the network.

4.2.1. Node Importance. As noted in Definition 2, in a realistic network, the more the times a node appears in sequence S_G, the less the importance to its neighbors. The quantitative representation of the importance of a node is the product of two statistics, node frequency (NF) and inverse CNS frequency (ICF). The node frequency refers to the frequency of a given node that appears in a context node sequence, which is a binary variable. In order to get the node frequency of v_i, first we denote f_{ij} as whether v_i constitutes S_j, where $v_j \in V$:

$$f_{ij} = \begin{cases} 0, & v_i \notin S_j, \\ 1, & v_i \in S_j. \end{cases} \tag{2}$$

We denote f_{S_j} as the total number of nodes in the sequence S_j. And then, we define $\mathrm{NF}(i, j)$ as the node frequency of v_i in S_j, which can be formulated as $\mathrm{NF}(i, j) = f_{ij}/f_{S_j}$.

ICF can be considered a measure of the universal importance of a node because it captures the distribution of importance in real-world networks. For a given node v_i, we can denote $\mathrm{ICF}(i)$ as the inverse CNS frequency as follows:

$$\mathrm{ICF}(i) = \log \frac{|V|}{\{j : v_i \in S_k\}}, \tag{3}$$

where $k \in \{1, 2, \ldots, |V|\}$. After incorporating the mentioned node frequency and inverse CNS frequency, the node importance (NI) of a given node v_i can be measured as

$$\mathrm{NI}(i) = \log \frac{\sum_j^{|V|} (\mathrm{NF}(i, j) \cdot \mathrm{ICF}(i))}{|V|}. \tag{4}$$

Note that NI is a context-based measure for each node in the network, and it extends TF-IDF thinking to network node analysis. Compared with the degree-based PageRank [17], NI incorporates richer contextual semantic structures rather than pairwise nodes, which enables our model to measure the importance of a node in the high-order neighborhood [18].

For a node v_i in an unweighted network, $\mathrm{NI}(i)$ can be served as the weights of edges starting from v_i. We can also consider NI as the ranking of node popularity in the network. The smaller the value, the higher the prevalence of a node. After obtaining the quantitative representations of NI in a given network, we can simply obtain the empirical distribution of the network, which can be defined as follows:

$$\widehat{p}(i) = \frac{\mathrm{NI}(i)}{\sum_{v_j \in V} \mathrm{NI}(j)}. \tag{5}$$

4.2.2. Structure-Based Objective. Formally, we model the conditional probability of v_j generated by v_i as

$$p(v_j \mid v_i) = \frac{\exp(v_j^s \cdot v_i^s)}{\sum_{v_z \in V} \exp(v_z^s \cdot v_i^s)}. \tag{6}$$

This equation can be interpreted as the probability of detecting the edge from v_i to v_j, which denotes the reconstructed distribution.

With the empirical distribution of the coincident probability between nodes and the reconstructed distribution, to preserve the node importance and network structures, a straightforward way is to minimize the following objective function:

$$O = \mathrm{distance}(\widehat{p}(\cdot), p(\cdot)), \tag{7}$$

where $\mathrm{distance}(\cdot, \cdot)$ is the distance between the two distributions. We choose KL divergence of two probability distributions to measure the difference between distributions. Thus, replacing $\mathrm{distance}(\cdot, \cdot)$ with KL divergence, we can obtain the following objective:

$$\begin{aligned} \mathscr{L}_s(e_{ij}) = \mathrm{KL}(p \parallel \widehat{p}) &= \sum_{(v_i, v_j) \in E} p(v_j v_i) \log\left(\frac{p(v_j \mid v_i)}{\widehat{p}(i)}\right) \\ &\propto - \sum_{(v_i, v_j) \in E} \mathrm{NI}(i) \log p(v_j \mid v_i). \end{aligned} \tag{8}$$

With this formulation, we can minimize the objective equation (8) to obtain vectors $\{v_i\}_{i=1..|V|} \in \mathbb{R}^{d_s}$ that represent nodes in the d_s-dimensional latent space based on the network structure. We summarize the structure-based embedding method in Algorithm 1.

4.3. Context Embedding. CAHNE is expected to integrate typical heterogeneous information like text features in the

```
Input: network $G$, context node sampling parameter $k$, dimensionality $d_s$, and learning rate $\eta$
Output: $d_s$-dimensional embedding results $H$
(1) Initialize nodes' relational trees $\{x_i\}_{i=1}^{|V|}$ by performing BFS on $G$ starting from each node;
(2) Obtain a context node sequence $S$ by sampling context nodes layer by layer for each anchor node according to $k$;
(3) for $i=1$ to $|V|$ do
(4) Calculate NI$(i)$ by equation (4);
(5) end for
(6) while not convergence do
(7) Update the value of loss function equation (8) and node representations $H$ by the Adam algorithm with learning rate $\eta$;
(8) end while
(9) Return $H$;
```

Algorithm 1: Structure-based embedding with node importance.

network. A straightforward way is to learn representations from text information of nodes and network structures independently. However, it ignores the complex interactions and associations between topological structures and heterogeneous information. To bridge this gap, we introduce context embedding to fuse information of context nodes for an anchor in the network so that we can overcome the incompatibility problem.

As shown in Figure 2, we sample context nodes for the anchor node v_i and obtain a context node sequence S_i when setting k as 2. In a CNS, text features of different context nodes have various impacts on the anchor node. Thus, we expect to give a weight to each context node in a CNS, and the weights can reflect the impact trend of context nodes. To this end, we introduce exponentially weighted moving average [19].

4.3.1. Exponentially Weighted Moving Average (EWMA). Moving average (MA) is a calculation to analyze sequential data which reflect the changing trend in the sequence. Based on MA, exponentially weighted moving average (EWMA) applies weighting factors which decrease exponentially. The older data are attached with lower weights, but weights never reach zero. The EWMA for a sequence Y can be formulated recursively:

$$\begin{aligned}\mathrm{EWMA}(t) &= \gamma \mathrm{EWMA}(t-1) + (1-\gamma)y(t) \\ &= (1-\gamma)y(t) + \gamma^2 \mathrm{EWMA}(t-2) \\ &\quad + \gamma(1-\gamma)y(t-1) \\ &\quad \vdots \\ &= \sum_{i=1}^{t} \gamma^{t-i}(1-\gamma)y(i),\end{aligned} \tag{9}$$

where γ is a parameter that represents the degree of weight decrease and $0 \le \gamma < 1$. $y(t)$ is the current data, and EWMA(t) represents the EWMA value of the current data. In the tree x_i, the deep layer nodes need to be given small weights because they are farther away from the anchor node. As a result, we can attach weight for each context node in S_i. However, the nodes in the same layer need to be sorted first. For consistency, we sort the same layer nodes according to their NI values. And then, a normalized context node sequence can be generated for the anchor node v_i as $S_i = \{v_i : (v_{i,1}, v_{i_2}, \cdots, v_{i,e})\}$, where $(v_{i,1}, \cdots, v_{i,e})$ are sampled context nodes of v_i. Afterwards, we apply EWMA on the context nodes from $v_{i,1}$ as follows:

$$\mathrm{EWMA}\left(v_{i,t}\right) = \gamma \mathrm{EWMA}\left(v_{i,t+1}\right) + (1-\gamma)v_{i,t}. \tag{10}$$

As the similarity we introduced EWMA, we treat $\gamma^{t-1}(1-\gamma)$ as the weight of the context node $v_{i,t}$, which is denoted as $\mathscr{W}_{i,t}$.

4.3.2. Text Information Representation. With the development of deep learning, there are many neural network models to learn text representations, e.g., convolutional neural network (CNN) [8, 20, 21], recurrent neural network (RNN) [22], long short-term memory (LSTM) [23], and gated recurrent units (GRUs) [24]. In this paper, we investigate different Word2vec models and find the CNN has the best performance on our tasks, which can capture comprehensive semantics in the heterogeneous text network.

In Figure 3, we show the framework of a generating process of context embedding. Given a normalized context node sequence S_i rooted at v_i, we take the word sequence of each node in S_i as the input, and the CNN obtains text embedding through three layers, i.e., encoder and looking-up, convolution, and mean-pooling. And then, we adopt weighted summations for the representation vectors of the anchor node and its context nodes to obtain context embedding v_i^c for v_i.

(1) Encoder and Looking-Up. First, we map all words in the heterogeneous text network to a sequence of word IDs. Hence, we can obtain an ID sequence for $t \in T$. And then, the looking-up layer transforms each word $w \in t$ into a vector $w \in \mathbb{R}^{d_w}$, where d_w is the dimension of word embeddings. Finally, we can obtain an embedding sequence $W_i = (w_1, \ldots, w_{n_i})$ for v_i. As is shown in Figure 3, after the encoder and looking-up layer, we can get a matrix sequence $P(i) = (P_i, \ldots P_{i,m}, \ldots, P_{i,n})$, and P_i is equivalent to $P_{i,0}$.

(2) Convolution. After the encoder and looking-up layer, we use the convolution layer to extract the features of the input matrix sequence $P(i)$. We perform convolution operation by a kernel $K \in \mathbb{R}^{d_t \times (1 \times d_w)}$ to slide row by row in $P_{i,x}$ $(x \in \{0, \cdots, n\})$ as follows:

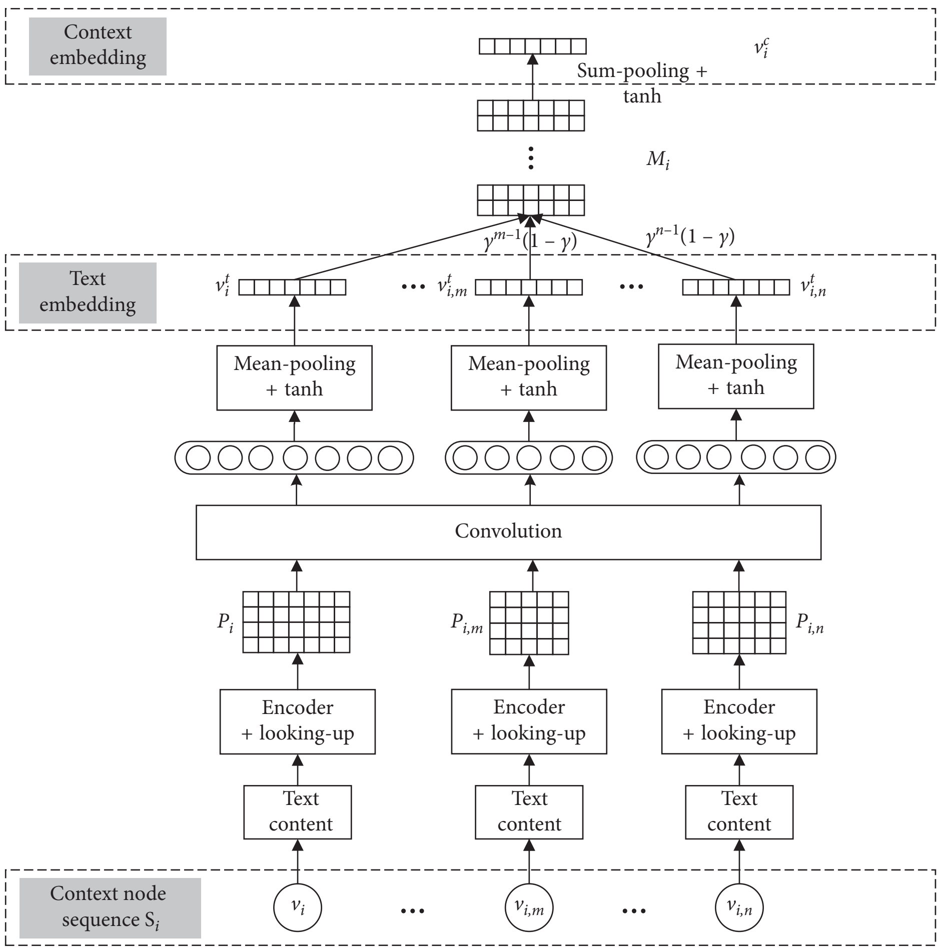


FIGURE 3: An illustration of context embedding for the anchor node v_i.

$$y_{i,x} = K \cdot P_{i,x} + b, \tag{11}$$

where $y_{i,x} = [y_1^x, \ldots, y_{n_x}^x]$ denotes the feature vector of $P_{i,x}$, in which n_x is the number of words in $t_{i,x}$ (the text of $v_{i,x}$), and b is the bias vector.

(3) Mean-Pooling. We test different pooling regulations. To get full-scale features of the text information for a node, we perform mean-pooling to get the text embedding v^t. Then, we choose tanh as the nonlinear activation function over $y_{i,x}$, which is

$$a_j = \tanh\left(\mathrm{mean}\left(y_1^x, \ldots, y_{n_x}^x\right)\right), \tag{12}$$

where $j \in \{1, 2, \ldots, d_t\}$, in which d_t is the dimension of text embedding. At last, we can get the embedding of the text information for $v_{i,x}$ as $v_{i,x}^t = [a_1, \ldots, a_{d_t}]$.

So far, we have obtained text embedding by the CNN for each node in a context node sequence. Following this, we do weight summations on the context node embeddings $(v_{i,1}^t, \cdots, v_{i,n}^t)$, and this operation is sum-pooling in Figure 3. The strategy of generating context embedding for v_i is as follows:

$$v_i^c = \tanh\left(v_i^t + \sum_{j=1}^{n} \mathscr{W}_{i,j} v_{i,j}^t\right). \tag{13}$$

Through the method stated, we establish correlations between the anchor node and its context nodes in terms of representation vectors and maintain text relevance. Eventually, we can get context embedding for a given node v_i, and the whole representation of v_i is bespoken as $v_i = v_i^s \oplus v_i^c$.

The text embedding part of the context embedding framework shown in Figure 3 looks like the convolution method of CANE. The difference is that the input of our model is the CNS of a node, while the input of CANE is a pair of nodes. In addition, we sort the nodes in the CNS according to NI and weight each node in CNS with EWMA values, as shown in equation (13).

4.3.3. *Context Embedding Objective.*

Context embedding objective aims to measure the log-likelihood of a given directed edge $(v_i, v_j) \in E$ as

$$\mathscr{O} = \log \frac{\exp\left(v_i^c \cdot v_j^c\right)}{\sum_{v_z \in V} \exp\left(v_i^c \cdot v_z^c\right)}. \tag{14}$$

Thus, the loss function of generating context embedding can be represented as $\mathscr{L}_c(e_{ij}) = -\mathscr{O}$. With above formulations, CAHNE aims to minimize the overall loss function as

$$\mathcal{L} = \sum_{e_{ij}\in E}\left(\mathcal{L}_s\left(e_{ij}\right) + \mathcal{L}_c\left(e_{ij}\right)\right). \tag{15}$$

At last, the workflow of the context embedding method is summarized in Algorithm 2.

4.4. Optimization of CAHNE

4.4.1. Attention for Context Node Sequence. Noticing the context embedding-generating strategy in equation (13), the vector representation of the anchor node v_i is decomposed as the affinity between v_i^t and its context nodes' representations $\sum_{j=1}^{n}\mathcal{W}_{i,j}v_{i,j}^t$. Intuitively, the affinity between context nodes and the anchor nodes should depend on the specific anchor node. For instance, v_i and v_j are anchor nodes in a real-world network, but they have different properties; as a result, they have varied intensity of affinity with their context nodes. Therefore, it is a requisite to incorporate such characters of the anchor nodes in modeling the unique excitation effects α.

In line with the attention mechanism [25], a novel and popular model for machine translation, we define the weights between the anchor node and its context nodes with the softmax unit as follows:

$$\alpha_i = \frac{\exp\left(v_i^t + \sum_{j=1}^{n}\mathcal{W}_{i,j}v_{i,j}^t\right)}{\sum_{m'}\exp\left(v_i^t + \sum_{j=1}^{n'}\mathcal{W}_{m',j}v_{m',j}^t\right)}. \tag{16}$$

Therefore, equation (13) can be reformulated as

$$v_i^c = \tanh\left(v_i^t + \alpha_i\sum_{j=1}^{n}\mathcal{W}_{i,j}v_{i,j}^t\right). \tag{17}$$

4.4.2. Negative Sampling. For equation (8) and equation (14), CAHNE aims to maximize the conditional probability between v_i and v_j, which is computationally expensive because of the softmax function for all nodes. To address this problem, we adopt the method of negative sampling [26] to approximate the objective function as the following form:

$$\log\sigma\left(v_j^T\cdot v_i\right) + \sum_{c=1}^{n}E_{z\sim P(v)}\left[\log\sigma\left(-v_j^T\cdot z\right)\right], \tag{18}$$

where $\sigma(x) = 1/(1+\exp(-x))$ represents the logistic function and n is the number of randomly sampled vertices. We set $P(v)\propto d_v^{3/4}$, where d_v is the out-degree of v. At last, we adopt the Adam algorithm [27] for optimizing equation (18) and set the learning rate as 0.001.

5. Experiment

In this section, we empirically evaluate the performance of the proposed framework CAHNE.

5.1. Dataset Descriptions. In order to comprehensively evaluate the effectiveness of our model CAHNE, we use seven real-world datasets, including two social networks, two citation networks, one language network, one co-occurrence network, and one communication network, for four applications, i.e., network reconstruction, link prediction, node classification, and visualization. The detailed descriptions are listed as follows:

(i) Zhihu [28] is a network of social relationships which is an online Q&A platform in China. Users follow each other, asking and answering questions on Zhihu. The text information is concerned topics of each user, which is expressed as full text. We filter out 10000 users from Zhihu who have information on concerned topics. The size of the vocabulary is 9035, and the average length of the text is 89. We evaluate this dataset on the link prediction task.

(ii) HEP-TH [8] is a citation network from arXiv. After filtering out the papers without abstract, 1038 papers are preserved. The text information is expressed as full text. The size of the vocabulary is 2970, and the average length of the text is 54. We evaluate these data on the link prediction task.

(iii) Cora (https://linqs.soe.ucsc.edu/data) is also a citation network containing 2708 machine learning papers with text information classified into one of seven classes. The citation network consists of 5429 links. The text information is expressed as full text. The size of the vocabulary is 16426, and the average length of the text is 88. Cora is used for the link prediction task and node classification task.

(iv) BlogCatalog (http://leitang.net/social_dimension.html) is a large social network of online users listed on the BlogCatalog website. There are 39 different categories of labels for this dataset, and each label represents the metadata provided by a user. Since this dataset does not contain text information, it will be evaluated on the node classification task and network reconstruction for CAHNE (without context embedding).

(v) Wikipedia [29] is a co-occurrence network which contains 2045 nodes, 17981 edges, and 19 different labels. The tf-idf matrix of the Wikipedia dataset describes the text information for this dataset. There are 4973 columns that correspond to 4973 different words. This dataset will be evaluated on the node classification task.

(vi) 20-NewsGroup (http://qwone.com/~jason/20Newsgroups/) is a collection of approximately 20,000 newsgroup documents, partitioned (nearly) evenly across 20 different newsgroups. We choose the news documents labeled as comp.graphics, rec.sport.baseball, and talk.politics.gums to evaluate our model on the visualization task. There are 1720 pieces of news contained and expressed as full text. The size of the vocabulary is 30127, and the average length of the text is 206. Besides, 20-NewsGroup is a weighted network.

(vii) Email-Enron (https://snap.stanford.edu/data/email-Enron.html) is a communication network that covers the email communication within a dataset. Nodes are email addresses, and edges denote

```
    Input: network $G$, context node sequences $S$, dimensionality $d_t$, learning rate $\eta$, EWMA parameter $\gamma$, and NI values
    Output: $d_t$-dimensional embedding results $C$
(1) Normalize context node sequences $S$ layer by layer with NI values;
(2) Apply EWMA on normalized context nodes with parameter $\gamma$ to obtain a weight for each context node;
(3) Encode text contents of nodes in the context node sequence and input them into the CNN;
(4) while not convergence
(5) Update the value of loss function $\mathscr{L}_c$ and node representations $C$ by the Adam algorithm with learning rate $\eta$;
(6) end while
(7) Return $C$;
```

Algorithm 2: Generating strategy of context embedding.

interactions between emails. Text descriptions of this dataset are full email message text. The size of the vocabulary is 29523, and the average length of the text is 149. We filter 6820 nodes and 23968 edges from the original dataset.

The detailed statistics are summarized in Table 1.

Table 1: Statistics of the dataset.

Dataset	#Nodes	#Edges	#Labels
Zhihu	10000	43894	—
HEP-TH	1038	1990	—
Email-Enron	6820	23968	—
Cora	2708	5429	7
BlogCatalog	10312	333983	39
Wikipedia	2405	17981	19
20-NewsGroup	1720	Fully connected	3

5.2. Baselines. We consider the following six NE methods to demonstrate the effectiveness and robustness of CAHNE:

(i) DeepWalk [4]: it adopts truncated random walk and Skip-Gram model to learn node representations.

(ii) LINE [5]: it preserves the first-order and second-order proximity among nodes in the network.

(iii) Node2vec [6]: it proposes a biased random walk based on DeepWalk to learn node representations.

(iv) GraRep [30]: it integrates global structural information of the graph and uses SVD to train the model.

(v) Naive Combination: we directly concatenate the text feature embeddings learned by the CNN and node representations learned from LINE for network representation. We choose LINE to learn structure embedding because it can exploit both first-order and second-order proximity in the network, which is more comprehensive than DeepWalk and Node2vec.

(vi) TADW [29]: it integrates text features into network embedding by employing matrix factorization.

(vii) TENE [31]: it learns the representations of nodes under the guidance of both the proximity matrix which captures the network structure and the text cluster membership matrix derived from clustering for text information.

(viii) ASNE [32]: it learns representations of nodes by preserving both the structural proximity and attribute (text) proximity.

5.3. Experimental Settings. To be fair, we set the embedding dimension $d = 100$ for all methods on HEP-TH, Cora, Email-Enron, and 20-NewsGroup. And for Zhihu, BlogCatalog, and Wikipedia, we set $d = 200$. For DeepWalk, we set the window size as 10, the walk length as 80, and the number of walks for each node as 10. For LINE, we set the learning rate as 0.001 and the number of negative samples as 5. For Node2vec, we choose the hyperparameters p and q to obtain the best performance by grid search. For GraRep, we set the maximum matrix transition step s as 5. For TENE, we set the parameter of the contribution of text information $\alpha = 10$ and the parameter β to guarantee the accuracy of the text cluster membership matrix as 10^7.

For our model CAHNE, we set the number of negative samples as 5 to speed up the training process. Besides, we set $\gamma = 0.5$ and $k = 2$ for all datasets. Hereinafter, we use "CAHNE-a" to validate the effectiveness of our method with the attention mechanism, and "CAHNE(w/o context)" denotes CAHNE without incorporating context embedding.

5.4. Network Reconstruction. Reconstructing the network and preserving the original network structure are fundamental objectives for network embedding methods. Definitely, we train an NE method to obtain vector representations of nodes and rank pairwise nodes according to the inner product similarities of them. Since the larger similarities mean higher probabilities of existing edges between pairwise nodes, the top ranking pairwise nodes are used to reconstruct the network efficiently. The precision@k [33] is used as the evaluation metric, which is formulated as

$$\text{precision@}k = \frac{\sum_{i=1}^{k} \xi_i}{k}, \tag{19}$$

where k is the number of evaluated pairwise nodes and ξ is a binary variable. $\xi_i = 1$ denotes the i-th reconstructed pair of nodes is correct; otherwise, it is wrong.

We use a real-world social network BlogCatalog and a communication network Email-Enron as representatives.

The result on the precision@k is shown in Figure 4, from which we make the following observations:

(i) Figure 4 shows that the precision@k of our method CAHNE almost outperforms that of other methods with the increase of k, which verifies that CAHNE can perfectly preserve the network structure.

(ii) Because there is no text information in BlogCatalog, Figure 4(a) can clearly reveal that using node importance to weight edges is effective.

(iii) Figure 4(b) shows our method has comparable performance on Email-Enron. We can notice that methods integrating text information are obviously better than other methods, and CAHNE-a can have a relatively high position.

From the above observations, we regard that our method CAHNE and its expansion CAHNE-a achieve a significant advance in efficiency on the task of network reconstruction.

5.5. Link Prediction. For link prediction, we use AUC [34] to evaluate the performance, which means the probability that nodes in a random edge are higher than those in a casual nonexistent edge. In this task, as shown in Tables 2–4, we randomly hide certain percentages of edges, respectively, from 85% to 5% on HEP-TH, Cora, and Zhihu and use the left graph to train. We use the logistic regression method to predict the probability of a given pair of nodes has an edge between them.

From these tables, some observations can be listed:

(i) The results show that the fewer the training edges, the more the nodes are ignored and the lower the performances of all methods. The results on Zhihu are worse than those on other datasets probably because real-world social networks are often accompanied by more complex information from both structures and properties compared to citation networks. However, our proposed model CAHNE-a always achieves the best performances compared to all other baselines on all different datasets. Especially, when the ratio of training edges reaches 95% in Cora and HEP-TH, AUC values of CAHNE-a are higher than 95.

(ii) CAHNE(w/o context) performs better than other structure-only methods (DeepWalk, LINE, Node2vec, and GraRep). It demonstrates that merging node importance when learning network representation is valid and leads to better predicting power for new link formation.

(iii) TADW, TENE, ASNE, and CAHNE perform better than all other structure-only methods. It verifies our assumption that text information cannot be neglected in heterogeneous text networks. However, CAHNE cannot always perform better than TADW, such as shown in 15% in Table 2 and 15% in Table 3. We notice that this phenomenon occurs only when the training ratio is under 35%, which we believe is due to the fact that the CNS cannot contain most context nodes of the anchor node when the training ratio is too low. Also, if the CNS is too incomplete, it will lose a lot of information from the context. Table 5 shows the average length of CNSs when extracting different ratios of edges as training sets in three datasets. The completeness of CNSs will affect the effectiveness of CAHNE.

Thus, the results in tables can serve as evidence that CAHNE-a has a stable and best performance on all datasets and different training ratios. It demonstrates the flexibility and robustness of CAHNE, and the attention mechanism is significant when learning representations for real-world networks.

5.6. Node Classification. For this task, we choose BlogCatalog, Cora, and Wikipedia as training datasets in which each node is assigned a label. Given the node embeddings obtained by different NE methods as node features, we train a logistic regression classifier to predict the node labels. We use Macro-F1 and Micro-F1 as measurements to evaluate the performance. We vary the size of the training set from 50% to 90%, and the remaining nodes are the testing set. We repeat each classification experiment ten times and report the average performance in terms of both Macro-F1 and Micro-F1 scores. The results on BlogCatalog, Cora, and Wikipedia are shown and compared in Figure 5. Since BlogCatalog is without text information, we only consider CAHNE(w/o context) on this dataset.

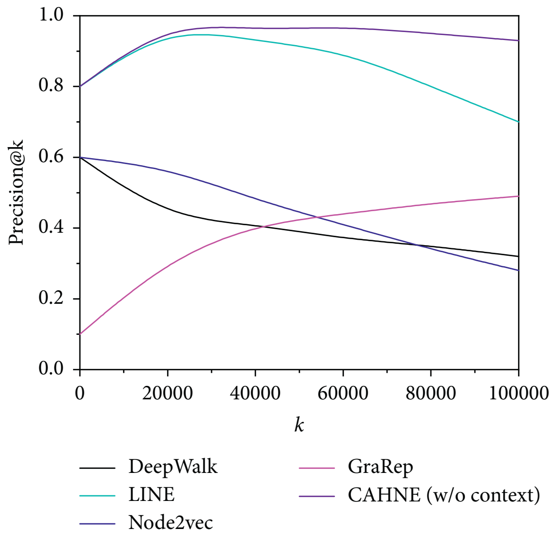
From the results, we obtain the following observations:

(i) The performances in BlogCatalog are worse than those in other datasets because of the complexity of social networks, and BlogCatalog has the most nodes which could reduce the capability of the classification task, but our proposed model CAHNE(w/o context) still obtains the most satisfactory results.

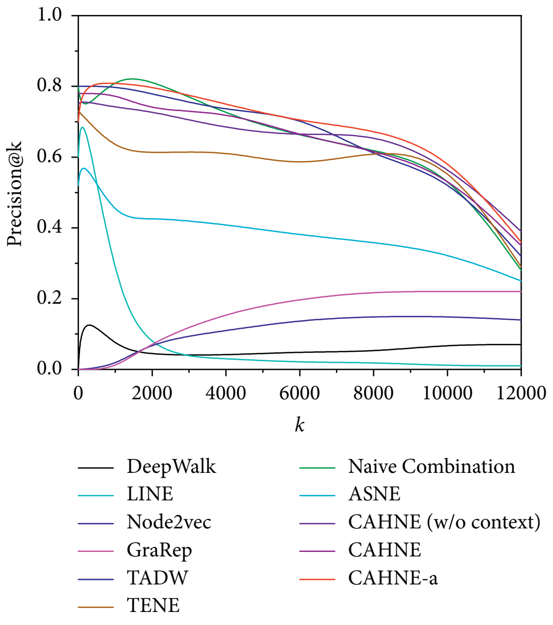
(ii) For structure-only methods, CAHNE(w/o context) has the best effectiveness on all datasets. It demonstrates that the network representations merging with node importance can be better generalized to the classification task.

(iii) CAHNE(w/o context) performs better than CAHNE and CAHNE-a on Wikipedia as measured by Macro-F1, which indicates this dataset is not sensitive to text information. We believe this is because the text descriptions between different entries vary widely.

5.7. Visualization. Another intuitive way to investigate the qualities of network embedding methods is visualization, and in this experiment, we reduce the dimensionality of each representation vector to 2. There are many ways to visualize high-dimensional vectors, e.g., PCA [35], Isomap [9], and t-SNE [36]. In this paper, we adopt t-SNE to achieve dimension reduction because t-SNE can preserve local and global structures of the data. Therefore, we use



(a)



(b)

FIGURE 4: precision@k on (a) BlogCatalog and (b) Email-Enron.

TABLE 2: AUC scores on HEP-TH.

% training edges	15%	25%	35%	45%	55%	65%	75%	85%	95%
DeepWalk	0.658	0.757	0.819	0.864	0.891	0.898	0.897	0.919	0.912
LINE	0.500	0.594	0.717	0.755	0.788	0.806	0.846	0.839	0.923
Node2vec	0.663	0.776	0.845	0.866	0.884	0.899	0.906	0.929	0.915
GraRep	0.628	0.735	0.776	0.841	0.853	0.872	0.885	0.896	0.914
Naive Combination	0.766	0.782	0.788	0.802	0.827	0.856	0.883	0.912	0.928
TADW	0.806	0.818	0.857	0.893	0.902	0.918	0.924	0.936	0.948
TENE	0.778	0.807	0.839	0.862	0.899	0.923	0.928	0.938	0.939
ASNE	0.783	0.802	0.833	0.869	0.893	0.905	0.918	0.926	0.938
CAHNE(w/o context)	0.730	0.796	0.854	0.894	0.893	0.913	0.916	0.921	0.923
CAHNE	0.786	0.818	0.860	0.896	0.902	0.928	0.935	0.937	0.954
CAHNE-a	**0.858**	**0.854**	**0.869**	**0.898**	**0.910**	**0.929**	**0.941**	**0.945**	**0.977**

TABLE 3: AUC scores on Cora.

% training edges	15%	25%	35%	45%	55%	65%	75%	85%	95%
DeepWalk	0.614	0.708	0.777	0.807	0.853	0.858	0.871	0.877	0.898
LINE	0.608	0.743	0.807	0.827	0.853	0.865	0.870	0.885	0.894
Node2vec	0.654	0.722	0.768	0.812	0.838	0.861	0.871	0.878	0.908
GraRep	0.589	0.732	0.786	0.826	0.852	0.874	0.897	0.898	0.914
Naive Combination	0.668	0.772	0.801	0.826	0.852	0.866	0.904	0.921	0.942
TADW	0.803	0.824	0.834	0.862	0.887	0.888	0.903	0.918	0.945
TENE	0.779	0.818	0.822	0.859	0.879	0.881	0.892	0.913	0.916
ASNE	0.718	0.742	0.809	0.832	0.849	0.870	0.902	0.921	0.933
CAHNE(w/o context)	0.654	0.747	0.803	0.843	0.877	0.885	0.901	0.909	0.915
CAHNE	0.793	0.805	0.828	0.863	0.892	0.898	0.908	0.925	0.954
CAHNE-a	**0.805**	**0.830**	**0.837**	**0.872**	**0.892**	**0.907**	**0.915**	**0.926**	**0.963**

baselines and our method CAHNE-a to learn representations of the 20-NewsGroup network and input them into t-SNE. From 20-NewsGroup, since all categories of graphs are full connection, to simplify the computational process and improve visualization performance, we filter three categories of news and their documents, comp.-graphics, rec.sport.baseball, and talk.politics.gums, as our training set.

Table 4: AUC scores on Zhihu.

% training edges	15%	25%	35%	45%	55%	65%	75%	85%	95%
DeepWalk	0.469	0.472	0.497	0.507	0.533	0.537	0.556	0.574	0.587
LINE	0.521	0.569	0.618	0.624	0.655	0.636	0.646	0.676	0.698
Node2vec	0.488	0.482	0.507	0.505	0.552	0.546	0.558	0.582	0.590
GraRep	0.583	0.619	0.642	0.659	0.654	0.662	0.663	0.668	0.663
Naive Combination	0.524	0.553	0.579	0.618	0.653	0.672	0.689	0.705	0.703
TADW	0.558	0.576	0.593	0.625	0.655	0.697	0.696	0.723	0.729
TENE	0.551	0.549	0.607	0.622	0.660	0.666	0.668	0.692	0.711
ASNE	0.586	0.563	0.608	0.633	0.661	0.682	0.699	0.700	0.728
CAHNE(w/o context)	0.595	0.600	0.603	0.604	0.612	0.618	0.639	0.657	0.679
CAHNE	0.623	0.693	0.706	0.709	0.707	0.711	0.713	0.722	0.731
CAHNE-a	**0.631**	**0.707**	**0.721**	**0.724**	**0.723**	**0.727**	**0.736**	**0.748**	**0.759**

Table 5: Average length of context node sequences when extracting different ratios of edges.

% training edges	15%	35%	55%	75%	95%
HEP-TH	0.8	2.2	5.7	6.9	7.2
Cora	2.6	4.3	7.9	15.2	17.7
Zhihu	2.3	4.1	6.6	9.7	10.3

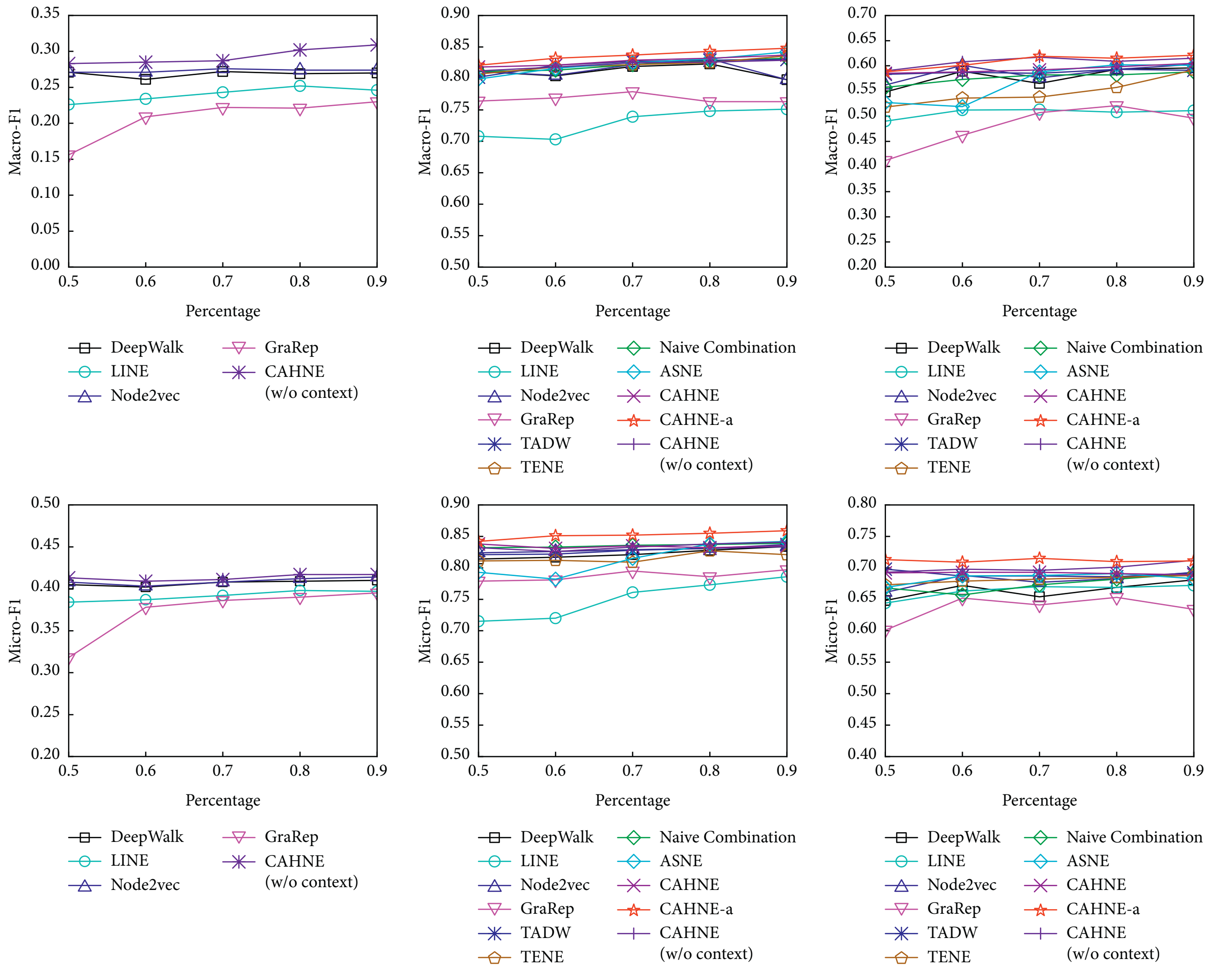


Figure 5: Macro-F1 and Micro-F1 on (a) BlogCatalog, (b) Cora, and (c) Wikipedia.

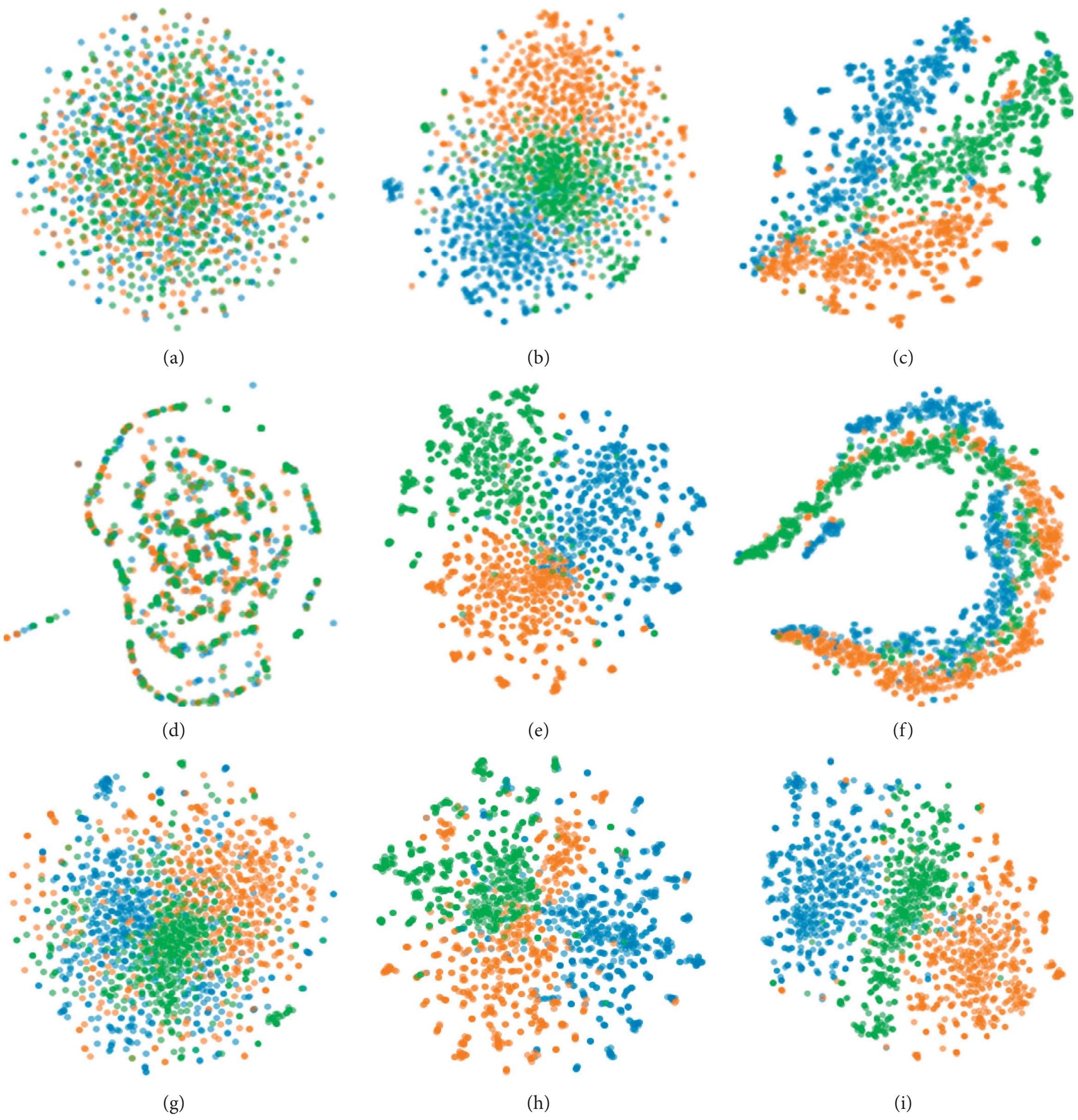


Figure 6: Visualization of the 20-NewsGroup network. Green represents the category of talk.politics.gums, orange represents the category of comp.graphics, and blue represents the category of rec.sport.baseball. (a) DeepWalk. (b) LINE. (c) Node2vec. (d) GraRep. (e) Naive Combination. (f) TADW. (g) TENE. (h) ASNE. (i) CAHNE-a.

The resulting visualizations with baselines and CAHNE-a are illustrated in Figure 6, from which we have the following observations:

(i) For DeepWalk and GraRep, all points of different categories are chaotic and mixed with each other. Since the network is weighted, DeepWalk cannot handle weighted networks when random walking, which leads to chaos. GraRep integrates weights of edges into representation learning by using E-SGNS, which is powerless to capture the nonlinear relationship between nodes.

(ii) For LINE, ASNE, TENE, and Naive Combination, we can intuitively find the clusters, but the boundary of each category is not clear.

(iii) For Node2vec, we can distinguish three categories more explicitly than for LINE because of a larger space between each cluster. However, the downsides of these clusters are not divisible.

(iv) For TADW, the shapes of clusters are not regular, and the blue points are not getting together.

Obviously, the visualization of our model CAHNE-a has a clear boundary, and the shapes of clusters are more regular than those reported in other baselines.

6. Conclusions

In this paper, we propose a novel method to learn node representations for heterogeneous networks, namely, CAHNE. By formulating the context node sequence for each node in a real-world network and redefining the conventional network to integrate text information, CAHNE achieves the learning of node embedding and captures the comprehensive semantic information, maintaining the compatibility between network structures and text information simultaneously. For the unweighted network, we analyze the strength of the relationship between nodes and propose the definition of node importance to quantify it as the weight between nodes. We integrate node importance into the learning process of structure-based embedding to explore the potential structural information in the network. Furthermore, by plugging an attention mechanism in the influence rate of the context nodes, CAHNE obtains the capacity to decide the influence degree from context nodes for different anchor nodes. Extensive experiments prove the competitiveness of CAHNE against baselines and demonstrate the flexibility, stability, and robustness of CAHNE. Future work includes incorporating more types of heterogeneous information like attributes of nodes and edges and optimizing the training process on larger networks.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Social Science Foundation of China under Grant 17CG209 and the National Natural Science Foundation of China under Grant 61872166. The work was also supported in part by the Natural Science Foundation of Jiangsu Province under Grant BK20180600 and Fundamental Research Funds for the Central Universities under Grant JUSRP11852.

References

[1] L. Lü and T. Zhou, "Link prediction in complex networks: a survey," *Physica A: Statistical Mechanics and Its Applications*, vol. 390, no. 6, pp. 1150–1170, 2011.

[2] C. Li, Z. Li, S. Wang, Y. Yang, X. Zhang, and J. Zhou, "Semi-supervised network embedding," in *Proceedings of the International Conference on Database Systems for Advanced Applications*, pp. 131–147, Springer, Suzhou, China, March 2017.

[3] Y. Jiang, F.-L. Chung, S. Wang, Z. Deng, J. Wang, and P. Qian, "Collaborative fuzzy clustering from multiple weighted views," *IEEE Transactions on Cybernetics*, vol. 45, no. 4, pp. 688–701, 2015.

[4] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701–710, ACM, New York, NY, USA, August 2014.

[5] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077, International World Wide Web Conferences Steering Committee, Florence, Italy, May 2015.

[6] A. Grover and J. Leskovec, "Node2vec: scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, ACM, San Francisco, CA, USA, August 2016.

[7] C. Li, S. Wang, D. Yang et al., "PPNE: property preserving network embedding," in *Proceeding of the International Conference on Database Systems for Advanced Applications*, pp. 163–179, Springer, Suzhou, China, March 2017.

[8] C. Tu, H. Liu, Z. Liu, and M. Sun, "CANE: context-aware network embedding for relation modeling," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, vol. 1, pp. 1722–1731, Vancouver, BC, Canada, August 2017.

[9] M. Balasubramanian and E. L. Schwartz, "The isomap algorithm and topological stability," *Science*, vol. 295, no. 5552, p. 7a, 2002.

[10] T. F. Cox and M. A. A. Cox, *Multidimensional Scaling*, Chapman and hall/CRC, Boca Raton, FL, USA, 2000.

[11] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 585–591, Vancouver, Canada, December 2002.

[12] Y. Goldberg and O. Levy, "Word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," 2014, http://arxiv.org/abs/1402.3722.

[13] T.-y. Fu, W.-C. Lee, and 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*, pp. 1797–1806, Singapore, November 2017.

[14] Z. Zhang, H. Yang, J. Bu et al., "Attributed network representation learning via deep neural networks," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 18, pp. 3155–3161, Stockholm, Sweden, July 2018.

[15] S. Ganguly and V. Pudi, "Paper2vec: combining graph and text information for scientific paper representation," in *Proceedings of the European Conference on Information Retrieval*, pp. 383–395, Springer, Aberdeen, Scotland, April 2017.

[16] F. B. Viégas and J. Donath, "Social network visualization: can we go beyond the graph," in *Proceedings of the CSCW Workshop on Social Networks*, vol. 4, pp. 6–10, Chicago, IL, USA, November 2004.

[17] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web," Technical report, Stanford InfoLab, Stanford, CA, USA, 1999.

[18] C. Yang, M. Sun, Z. Liu, and C. Tu, "Fast network embedding enhancement via high order proximity approximation," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 3894–3900, Melbourne, Australia, August 2017.

[19] J. M. Lucas and M. S. Saccucci, "Exponentially weighted moving average control schemes: properties and enhancements," *Technometrics*, vol. 32, no. 1, pp. 1–12, 1990.

[20] Y. Kim, "Convolutional neural networks for sentence classification," 2014, http://arxiv.org/abs/1408.5882.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in

Proceedings of the Advances in Neural Information Processing Systems, pp. 1097–1105, Lake Tahoe, NV, USA, December 2012.

[22] D. P. Mandic and J. A. Chambers, *Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability*, John Wiley & Sons, Hoboken, NJ, USA, 2001.

[23] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proceedings of the Thirteenth Annual Conference of the International Speech Communication Association*, Portland, Oregon, USA, September 2012.

[24] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, http://arxiv.org/abs/1412.3555.

[25] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 5998–6008, Long Beach, CA, USA, December 2017.

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3111–3119, Lake Tahoe, NV, USA, December 2013.

[27] D. P. Kingma and J. Ba. Adam, "A method for stochastic optimization," 2014, http://arxiv.org/abs/1412.6980.

[28] X. Sun, J. Guo, X. Ding, and T. Liu, "A general framework for content-enhanced network representation learning," 2016, http://arxiv.org/abs/1610.02906.

[29] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pp. 2111–2117, Buenos Aires, Argentina, July 2015.

[30] S. Cao, W. Lu, and Q. Xu, "GraRep: learning graph representations with global structural information," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 891–900, Melbourne, Australia, October 2015.

[31] S. Yang and B. Yang, "Enhanced network embedding with text information," in *Proceedings of the 24th International Conference on Pattern Recognition (ICPR)*, pp. 326–331, IEEE, Beijing, China, August 2018.

[32] L. Liao, X. He, H. Zhang, and T.-S. Chua, "Attributed social network embedding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 12, pp. 2257–2270, 2018.

[33] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1225–1234, ACM, San Francisco, CA, USA, August 2016.

[34] J. M. Lobo, A. Jiménez-Valverde, and R. Real, "AUC: a misleading measure of the performance of predictive distribution models," *Global Ecology and Biogeography*, vol. 17, no. 2, pp. 145–151, 2008.

[35] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1–3, pp. 37–52, 1987.

[36] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.