Knowledge-Based Systems 193 (2020) 105418

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Community-oriented attributed network embedding[☆]

Yuan Gao, Maoguo Gong[*], Yu Xie, Hua Zhong

School of Electronic Engineering, Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an, Shaanxi Province 710071, China

A R T I C L E I N F O

Article history:
Received 8 May 2019
Received in revised form 17 December 2019
Accepted 19 December 2019
Available online 23 December 2019

Keywords:
Representation learning
Attributed network embedding (ANE)
Community detection
Topic model

A B S T R A C T

Network embedding aims to map vertices in a complex network into a continuous low-dimensional vector space. Meanwhile, the original network structure and inherent properties must be preserved. Most of the existing methods merely focus on preserving local structural features of vertices, whereas they largely ignore the community patterns and rich attribute information. For example, the title of papers in an academic citation network could imply their research directions, which are potentially valuable in seeking more meaningful representations of these papers. In this paper, we propose a Community-oriented Attributed Network Embedding (COANE) framework, which can smoothly incorporate the community information and text contents of vertices into network embedding. We design a margin-based random walk procedure on the network coupled with flexible margins among communities, which limit the scope of random walks. Inspired by the analogy between vertex sequences and documents, the statistical topic model is adopted to extract community features in the network. Furthermore, COANE integrates textual semantics into representations through the topic model while preserving their structural correlations. Experiments on real-world networks indicate that our proposed method outperforms six state-of-the-art network embedding approaches on network visualization, vertex classification and link prediction.

1. Introduction

Information networks, such as academic networks, social networks, and encyclopedia networks, are ubiquitous in a variety of real-world systems. How to effectively extract the features of the information network is considered sophisticated and challenging. Due to the sparsity of the adjacency matrix, the underlying structural features cannot be well reflected, especially when networks are large-scale. Recently, network embedding has aroused considerable research interests, which aims to learn low-dimensional dense vectors for vertices in the network while preserving their properties [1–4]. Benefited from this, various network analysis tasks, such as link prediction [5] and vertex classification [6], can be conducted efficiently through existing machine learning algorithms.

In the last few years, numerous methods have been proposed to learn network representations. The vast majority of models tend to extract contextual information of vertices and then capture the low-order or high-order proximity among them [7,8]. For instance, DeepWalk [9] generates random walks for each vertex and treats them as contextual information to learn vertex representations. Node2Vec [10] further extends DeepWalk by employing two preset parameters to control the random walk procedure, which provide a trade-off between breadth-first and depth-first graph searches. Both DeepWalk and Node2Vec face the problem of insufficient sampling on dense networks, hence some local patterns will not be reflected. Besides, there are some works presented to incorporate attribute features of vertices, such as text [11,12] or labels [13,14], into network embedding. AANE [15] learns representations based on the decomposition of attribute affinity and the embedding difference between adjacent vertices. SemNE [16] integrates vertex order information into the embedding process, then modifies representations and makes them better fit the annotation data. Essentially, these research works mainly focus on preserving the microscopic structure of networks while ignoring global patterns, even though attribute or label information is taken into account. As a consequence, the learned representations cannot be well adapted to diverse network analysis tasks.

Real-world networks usually contain rich community information, which is of paramount importance in community-level applications, such as network visualization. Inspired by this, some researchers introduced modularity-based algorithms to preserve community information [17–19]. However, modularity optimization will cause the problem of resolution limitation [20], which

[☆] No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105418.

[*] Corresponding author.
E-mail address: gong@ieee.org (M. Gong).

https://doi.org/10.1016/j.knosys.2019.105418
0950-7051/© 2019 Elsevier B.V. All rights reserved.

makes it hard to explore more detailed communities. Moreover, text contents of vertices are also essential for network analysis, which are taken into account in attributed network embedding. Most of the methods directly perform the dimensionality reduction on TF–IDF matrixes [21], while this simple method can only measure the text similarity rather than the semantic similarity of words [22]. In addition, vertices in the same community usually share some common attribute information. For example, papers in the same research field contain similar titles and abstracts. These similarities are expected to be extracted to facilitate exploring the community structure of the network and obtaining effective representations. Therefore, it is necessary to integrate textual semantics and community patterns into network embedding simultaneously.

To address the aforementioned challenges, we propose a novel Community-oriented Attributed Network Embedding (COANE) method. As it has been proven in [9] that words in the text and vertices in random walk sequences follow the same power-law distribution, we assume the role which vertices play in communities is much like that of words in topics, and then the statistical topic model is applied to vertex sequences to obtain the structure-based community distribution of each vertex. Subsequently, there will be margins among communities according to the distribution. Margins are geometric measures in support vector machine for evaluating the confidence of a classifier with respect to its decision [23], while in our approach they are used to evaluate the confidence of the community division. Different from the rigid random walk [9] or semi-supervised walk [10], our customized random walk procedure is flexible and self-adaptive. The scope of random walks is limited by margins, thus more explicit communities can be detected, and the SkipGram model is adopted to maximize the conditional likelihood function for vertices in the same community. We aggregate text content of a vertex and its neighbors into a single document and apply the other statistical topic model to the document set as well, hence the adjacent vertices share the similar text-based community distribution. The modeling process for vertex sequences and documents are similar, whereas the diverse sequences generated through random sampling could help the topic model learn more stable topic distribution. Specifically, our major contributions are as follows:

- We propose a novel attributed network embedding method named COANE and introduce topic models to vertex sequences generated through random walk to capture community information of the network. Compared with traditional network embedding algorithms, our approach can efficiently extract the community information and attribute features to learn more effective and meaningful representations.
- We design a flexible community-oriented random walk strategy without the need to detect communities in advance. Furthermore, we introduce margins among communities, which are able to control the scope of random walks adaptively, and it solves the problem that random walk based methods cannot sample sufficient walk sequences on dense networks.
- We evaluate COANE on several real-world networks. Experimental results demonstrate that the proposed algorithm outperforms other community-aware or text-associated baselines on network visualization, vertex classification and link prediction tasks with competitive time consumption.

The remainder of this paper is organized as follows. Section 2 briefly presents the related backgrounds and algorithms about the random walk strategy, communities in the network and the topic model. In Section 3, we formally define notations and the problem of network embedding, then the details of COANE are described. Section 4 shows the extensive experiments to validate the effectiveness, and we analyze the parameter sensitivity of the proposed method. Finally, we conclude with a discussion of COANE framework and summarize the future work in Section 5.

2. Related works

Representation learning [24] has been widely used in different fields such as computer vision [25,26] and natural language processing [27,28]. With the rapid development of the Internet, a mass of data are produced by multifarious complex networks. The kind of unstructured data attracts the interests of many researchers in network embedding, and various methods are proposed to learn network representations.

Recently, deep learning techniques are introduced into network analysis to obtain feature vectors of vertices. Perozzi et al. [9] verified the analogy among vertices in random walk sequences and words in the text. They proposed the DeepWalk model, which proceeds by adopting the SkipGram architecture to the random walk sequences generated on networks. As a language model, SkipGram is able to learn continuous low-dimensional representations of words by optimizing a neighborhood preserving likelihood objective using stochastic gradient descent with negative sampling [29]. To be specific, a vertex sequence $s = (v_1, v_2, \dots, v_{|s|})$ obtained by a random walk over the network is treated as a word sequence, and each vertex in the sequence is considered to be a word. Next, DeepWalk can obtain network representations following SkipGram, which seeks to maximize the average log-probability of observing the context of a vertex v_i conditioned on its feature representation:

$$\max_{\Phi} \quad \frac{1}{|s|}\sum_{i=1}^{|s|} logPr\left(\{v_{i-w}, \dots, v_{i-1}, v_{i+1}, \dots, v_{i+w}\} \mid v_i\right) \tag{1}$$

where the context is composed of the words on both sides of the given word within a window size w. Inspired by DeepWalk, some constrained random walk methods emerge with different graph exploration strategies. Grover et al. [10] generated vertex sequences using Breadth-first Sampling and Depth-first Sampling strategies through two parameters p and q, in order to learn the homophily or structural equivalence of the input network. Yang et al. [30] proved that DeepWalk is equivalent to factorizing the matrix whose element M_{ij} can be interpreted as Pointwise Mutual Information (PMI) of a word-context pair (v_i, v_j). They also incorporated text contents of vertices into the matrix factorization framework. Chen et al. [31] presented the side information network embedding, which defines the semantical neighborhood to model the inscape of each node, then the random walk is applied to explore this neighborhood. Wang et al. [32] proposed a deep model with a semi-supervised architecture named SDNE, which maps the data to a highly non-linear latent space and is able to simultaneously optimizes the first-order and second-order proximity. Note that these methods only try to preserve the microscopic structure of networks though text information is integrated. Since they maximize the conditional likelihood function for context vertices, representations of vertices are only relevant to whether they appear in the same window of a sequence, but the community information which preserves the global patterns of the network is ignored [33]. Therefore, the representation vectors of vertices are more likely to be close when they are in different communities with the low-order proximity, while those in the same community with a weak relationship in microscopic structure can be far apart.

In addition to the microscopic structure, the community is one of the significant properties of complex networks as well.

Although some approaches preserve global features of a network, the community information has not been fully exploited in network embedding. Communities can impose constraints on vertex representation at a higher structural level, which make representations of vertices more similar when they are in the same community. Even though they may have a weak relationship in microscopic structure, communities could limit the distance of their representations from being far apart. Therefore, the community information needs to be integrated into network embedding to learn more discriminative vertex representations. Li et al. [34] presented a network embedding method based on evolutionary algorithms that can preserve both the local proximity of vertices and the community structure of the network by optimizing a multi-objective function. Chen et al. [35] proposed a method with valuable group information for large-scale networks by taking both the inner structures of the groups and the information across groups into account. Wang et al. [18] introduced a modularized non-negative matrix factorization (M-NMF) model to preserve both microscopic and mesoscopic structure. The performance of existing methods rely heavily on the accuracy of community detection, which is a tough but important question.

In recent years, statistical topic models, which aim to use the observed document to infer the hidden topic distribution, have been successfully adopted to mine topics in the intricate text [22,36,37]. Some researchers employed topic models to co-author networks to infer research community [38,39], whereas they did not leverage the network topological structure among authors, which is extremely useful for refining topics. Mei et al. [40] proposed a general solution of text mining with network structure called NetPLSA, which optimizes the topic smoothness on the graph. It regards links among documents as a network regularization, such that linked documents could share similar topic distributions. To the best of our knowledge, NetPLSA is the only attempt to leverage the text information of the network through the topic model. However, the model focuses on extracting community features from text attributes, while the network topological structure is only used as auxiliary, hence it cannot obtain robust representations while handling noisy or incomplete text information.

There are other works on complex network representations. Shi et al. [41] introduced a network diffusion embedding method to solve the limitations that random walks are likely to bias to high-degree vertices, and neglect the global structure information in highly biased networks. Chen et al. [42] presented a generalizable model that utilizes both linkage information and centrality information to learn low-dimensional vector representations, which can preserve different centrality information of vertices. Zhao et al. [43] proposed a unified framework for social and behavior recommendations with network embedding, and a joint network embedding approach is introduced as a pre-training step for users' latent representations. Li et al. [44] presented an unsupervised network embedding model to encode edge relationship information, thus feature representation of vertices can be further captured. Wu et al. [45] introduced a multi-task dual attention LSTM model to learn network representations for specific tasks. The model can capture structure, content and label information, then adjust vertex representations according to the downstream task.

Unlike the above research, we propose a novel community-oriented random walk procedure. Moreover, the topic model is utilized to explore community structure in networks. The proposed method can preserve both microscopic structure and community characteristics simultaneously to learn more meaningful and discriminative representations.

3. Methodology

In this section, we will give a detailed description for the statistical topic model and the proposed algorithm. Before introducing the whole approach, we start with discussing the statement of the network embedding problem.

3.1. Problem statement

Assume that there is an attribute network $G = (V, E, T)$, where V is a set of vertices; $E \subseteq V \times V$ are edges that denote the relations between vertices; and T represents the text contents of vertices. Specifically, the text information of a vertex $v \in V$ corresponds to a word sequence $T_v = (w_1, w_2, \dots, w_{n_v})$, in which $n_v = |T_v|$. Network embedding attempts to build a low-dimensional feature matrix denoted as $\Phi \in \mathbb{R}^{|V|\times d}$ for a network, where $d \ll |V|$ specifies the dimension of the latent representation space, such that these representations can be fed into further network analysis applications.

3.2. LDA-based community detection

As mentioned before, the community plays a critical role in network embedding. To extract global community patterns, we design a margin-based random walk with a combination of the statistical topic model. The Latent Dirichlet Allocation (LDA) [22], a classic statistical topic model, is used in this paper to estimate community distributions. In order to better expound how LDA works in network community detection, the model will be described through vertices and communities instead of words and topics.

The generation process of each vertex sequence is based on the hypothesis that a network consists of several communities and each vertex in the network belongs to single or multiple communities with a certain probability. Under this assumption, a whole sequence can be regarded as a combination of different communities. In this way, LDA can learn the similarity of vertices by grouping them into unobserved communities, and a mixture of these communities constitutes the observable sequence [46]. The modeling process of LDA corresponds to the conditional distribution of hidden variables and observed variables below:

$$Pr(v_i|s) = \sum_{j=1}^{K} Pr(v_i|c_i = j)Pr(c_i = j|s) \tag{2}$$

where $Pr(v_i|s)$ is the probability of the vertex v_i appearing in the given sequence s. $Pr(v_i|c_i = j)$ represents the probability of the vertex v_i within the latent community j, and $Pr(c_i = j|s)$ indicates the probability of picking a vertex from the community j in the sequence s. LDA estimates the community-vertex distribution $Pr(v|c)$ and the sequence-community distribution $Pr(c|s)$ from a given number of communities k using Gibbs sampling. It randomly assigns a community c_i to each vertex v_i of the current sequence s. Then each vertex v_i is traversed and its community j is updated based on the probability $Pr(c_i = j|v_i, s, c_{-i})$ using the following equation, until the LDA model parameters converge [47].

$$Pr(c_i = j|v_i, s, c_{-i}) \propto \frac{N_{VC}(v_i, j) + \beta}{\sum_v N_{VC}(v, j) + |V|\beta} \cdot \frac{N_{SC}(s, j) + \alpha}{\sum_c N_{SC}(s, c) + k\alpha} \tag{3}$$

Here $N_{VC}(v, j)$ is a count of all community-vertex assignments, and $N_{SC}(s, c)$ indicates a count of the sequence-community assignments. Besides, c_{-i} represents all the above assignments except the current assignment t_i for vertex v_i, and α and β are the hyperparameters serving as smoothing factors for the Dirichlet priors. Therefore, the conditional distribution of $Pr(v_i|c_i = j)$ and $Pr(c_i = j|s)$ can be estimated as follows:

$$Pr(v_i|c_i = j) = \frac{N_{VC}(v_i, j) + \beta}{\sum_v N_{VC}(v, j) + |V|\beta} \tag{4}$$

$$Pr(c_i = j|s) = \frac{N_{SC}(s, j) + \alpha}{\sum_c N_{SC}(s, c) + k\alpha} \tag{5}$$

With these two conditional distributions, we can estimate the probability that a vertex in the random walk sequence belongs to each community, thereby the community structure of the network is initially detected.

3.3. Community-oriented attributed network embedding

In this paper, we propose a novel margin-based random walk strategy, which is controlled by margins among vertices and inclined to access a vertex with a smaller margin from the current one. Two LDA models with different hyper-parameters are used respectively to learn the structure-based and the text-based community distribution. Subsequently, the representations of vertices can be learned through the SkipGram model.

At the early stage, we employ the original random walk on the network, and vertex sequences generated during this procedure are input into the structure-based LDA model for continually updating. After a certain number of iterations, the preliminary global patterns and structure-based communities are learned, and there will be margins among vertices in different communities as visualized in Fig. 1. Then the margin-based random walk will be adopted in subsequent phases. During the iterative process, communities become more explicit, and the margins among vertices in different communities will also enlarge, so that random walks tend to access vertices within a particular community. When the vertices in a sequence belong to the same community, it will facilitate LDA learning the real community distribution according to Eq. (3), and make the representations of vertices in the same community more compact. In this mutually reinforcing process, SkipGram is applied to maximize the conditional likelihood function for vertices in the same community, thus representation vectors of vertices are obtained from these random walk sequences.

Moreover, the text content of a vertex is aggregated with that of several neighbors into a single document and is input to the text-based LDA model. Take the citation network as an example, each paper is viewed as a vertex and its title as the text content. Without loss of generality, we assume that there is an intrinsic connection between the title of a paper and that of papers it cites. These related titles are aggregated, then the text-based LDA can be leveraged to explore explicit topics, or communities, in there. Similarly, the probability of a neighbor being selected depends on the margin between it and the source vertex, and a vertex with a smaller margin from the source one is more likely to be selected, because vertices in the same community usually share similar text contents. The representation vector of the text information is represented by the text-based community distribution, which is eventually concatenated to the original structure-based representations. The framework of COANE is given in Algorithm 1.

The matrix $\mathcal{U}$ is generated randomly to initialize representation vectors of vertices in line 1 of Algorithm 1. M_{max} indicates the current maximum margin among vertices, which is initialized to 0 and has a maximum value of 1. When it equals 0, the margin-based random walk is equivalent to the traditional DeepWalk algorithm. As M_{max} increases, the probability of accessing vertices across different communities is gradually decreasing. When M_{max} reaches 1, the random walk can only be carried out within the same community. In this way, M_{max} can be considered as the

Algorithm 1 Framework of COANE

```
Input: graph: $G(V, E, T)$; window size: $w$; representation dimension: $d$; walks per vertex: $\gamma$; walk length: $l$; number of communities: $k$; margins appear moment: $M_m$; margins enlarge speed: $M_s$.
Output: matrix of network representations: $\Phi \in \mathbb{R}^{|V|\times(d+k)}$
1: Sample $\Phi$ from $\mathcal{U}^{|V|\times d}$
2: $M_{max} = 0$
3: for $i = 0$ to $\gamma$ do
4:   if $i > \gamma \times M_m$ then
5:     $M_{max} = min(1, M_{max} + \frac{1}{\gamma \times M_s})$
6:   end if
7:   $\mathcal{O}$ = Shuffle($V$)
8:   for each vertex $v \in \mathcal{O}$ do
9:     $\mathcal{S}_v$ = RandomWalk($G$, $v$, $Pr_s$, $M_{max}$, $l$)
10:    $\mathcal{D}_v$ = ContextAggregation($G$, $v$, $Pr_s$, $M_{max}$)
11:    SkipGram($\Phi$, $\mathcal{S}_v$, $w$)
12:  end for
13:  $Pr_s$ = LDA$_s$($\mathcal{S}$)
14:  $Pr_t$ = LDA$_t$($\mathcal{D}$)
15: end for
16: $\Phi = \Phi \oplus Pr_t$
17: return $\Phi$
```

confidence in community division. It is ever-increasing during the iterative process in lines 4–6 of Algorithm 1, and the process is controlled by two parameters M_m and M_s between 0 and 1. Among them, M_m denotes the margins appear moment, and M_s indicates the time required for M_{max} to reach 1. In other words, M_{max} will be 1 when the number of iterations reaches $\gamma \cdot (M_m + M_s)$. Through the margin-based random walk, which is described in detail in Algorithm 2, vertex sequences are generated and fed into the structure-based LDA model. Similarly, we aggregate text features of a vertex to that of its neighbors and input them into the other text-based LDA model. Afterward, the probability of vertices belonging to structure-based community Pr_s and that belonging to text-based community Pr_t can be learned. The former is used to guide the above generation process, while the latter is finally concatenated to the original representation vectors obtained by SkipGram in line 16.

Algorithm 2 Margin-based Random Walk

```
Input: graph: $G(V, E, T)$; walk length: $l$; number of communities: $k$; probability of vertices belonging to structure-based communities: $Pr_s$; max margin among vertices: $M_{max}$.
Output: a random walk sequence: $\mathcal{S}_v$
1: Initialize random walk
2: while $length(\mathcal{S}_v) < l$ do
3:   if current vertex $v$ has neighbors then
4:     for each neighbor vertex $u$ of $v$ do
5:       $p(u|v) = 1 - \frac{1}{2} \cdot \sum_{i=1}^{k} |Pr_s(c_i|v) - Pr_s(c_i|u)| \cdot M_{max}$
6:     end for
7:     select a vertex $u$ from neighbors of $v$ based on Roulette Wheel
8:   else
9:     backtrack in the sequence and select a vertex at random
10:  end if
11: end while
12: return $\mathcal{S}_v$
```

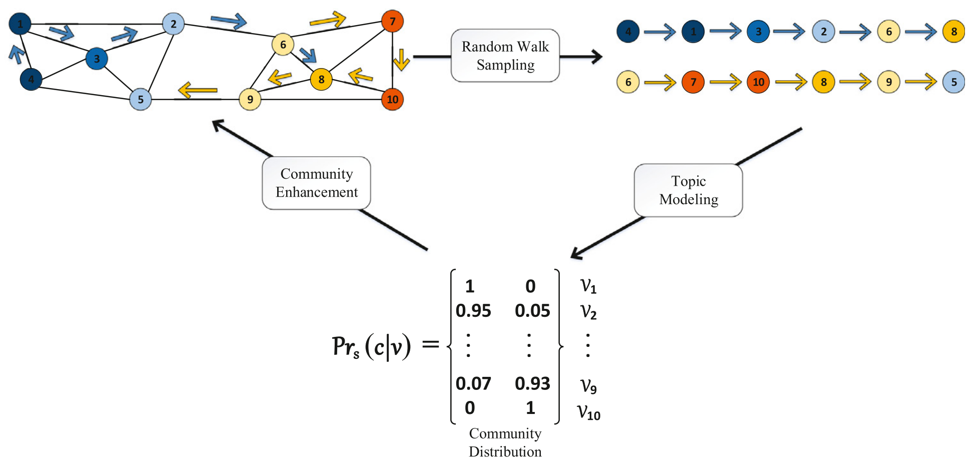


Fig. 1. A graphical illustration of the margin-based random walk. There are orange and blue vertices, representing two communities, and the color difference in the network indicates the margins among vertices, which enlarge during the iterative process.

The vertex sequence is initialized with a vertex v, and it iterates multiple times until the length of the sequence reaches the preset walk length l. Lines 4–7 in Algorithm 2 shows the core of our approach. In the case that the current vertex is v, the conditional probability of the next vertex u, $p(u|v)$, is related to the maximum margin and whether they belong to the same community. The probability will be 1 if they have the same community distribution or M_{max} is 0, while it will be 0 on the condition that they belong to entirely distinct communities and M_{max} equals to 1. The selection of subsequent vertices relies on the selection strategy based on the roulette-wheel selection of the genetic algorithm [48], so that vertices within the same community have a higher probability of being selected. It will backtrack in the sequence and select a vertex at random if the current vertex has no neighbors. Furthermore, the SkipGram and LDA model are online learning approaches, which means vertex sequences can be input into models in small batches rather than all at once. Our method is able to accommodate small changes in the network, and new random walks can focus on the changed region without the need for the recomputation of previous vertices.

Algorithm 3 Context Aggregation

```
Input: graph: $G(V, E, T)$; number of communities: $k$; probability
    of vertices belonging to structure-based communities: $Pr_s$;
    max margin among vertices: $M_{max}$.
Output: the contextual text information: $\mathcal{D}_v$
 1: Initialize $\mathcal{D}_v$ with $T_v$
 2: while $length(\mathcal{D}_v) < \gamma \cdot log|V|$ do
 3:   if current vertex $v$ has neighbors then
 4:     for each neighbor vertex $u$ of $v$ do
 5:       $p(u|v) = 1 - \frac{1}{2} \cdot \sum_{i=1}^{k} |Pr_s(c_i|v) - Pr_s(c_i|u)| \cdot M_{max}$
 6:     end for
 7:     select a vertex $u$ from neighbors of $v$ based on Roulette
        Wheel
 8:     $\mathcal{D}_v = \mathcal{D}_v \oplus T_u$
 9:   else
10:     $\mathcal{D}_v = \mathcal{D}_v \oplus T_v$
11:   end if
12: end while
13: return $\mathcal{D}_v$
```

Algorithm 3 shows the process of context aggregation. In order to lower the deviation between the posterior community distribution and the real community distribution, it must be satisfied that the length of a document is not less than the log of the total number of documents [49].

$$\gamma \cdot \log |V| \leqslant length(\mathcal{D}_v) \tag{6}$$

However, vertices may contain little attribute information, even just a few tags. Based on the fact that vertices in the same community are tightly connected and their attributes are highly similar, the text features of a vertex is aggregated to those of its neighbors into a single document, which satisfies the limitation of documents length while preserving the first-order proximity. Similar to the selection mechanism of Algorithm 2, Algorithm 3 selects neighbors with more consistent community distributions as well.

3.4. Complexity analysis

In the proposed algorithm, the training process consists of two part, respectively corresponding to the SkipGram model and LDA model. Specifically, the complexity of SkipGram in DeepWalk [9] is $\mathcal{O}(n\gamma l \log n)$, where n is the total number of vertices, γ is the number of walks and l is the walk length. For the structure-based LDA model, the complexity is $\mathcal{O}(n\gamma kl)$, in which k is the number of communities. For the text-based LDA model, its complexity is $\mathcal{O}(n\gamma kt)$. Here, t is the average length of the text information of each vertex, which is larger than $\gamma \cdot \log n$. In total, the computational complexity is $\mathcal{O}(n\gamma(l \log n + kl + kt))$ for COANE, and $\mathcal{O}(n\gamma l(\log n + k))$ for CONE.

4. Experiments

In this section, we first introduce the baselines, experimental networks and parameter settings. Then, we evaluate our algorithm on an unsupervised learning task: network visualization, and two supervised learning tasks: vertex classification and link prediction. The COANE and its attribute-free version (community-oriented network embedding, or CONE) are compared with six baseline methods of network embedding on real-world datasets. Besides, we also empirically analyze the effects of three key parameters, the margins appear moment M_m and the margins enlarge speed M_s, as well as the number of communities k.

4.1. Baselines

DeepWalk [9] is the pioneer algorithm that applies natural language processing to network embedding, which generates random walks over networks and employs SkipGram [50] to obtain the representation for each vertex. In addition, DeepWalk can be regarded as a special case of CONE with $M_m = 1$ or $M_s = \infty$.

HOPE [51] can well preserve asymmetric transitivity by minimizing $\|S - Y_s Y_t^T\|_F^2$, where S is the similarity matrix, and it is scalable to preserve high-order proximities of large-scale networks and capable of capturing the asymmetric transitivity.

M-NMF [18] is a modularized non-negative matrix factorization (NMF) model for network embedding, which optimizes the NMF-based embedding model and the modularity-based community detection model in a unified framework, so that both microscopic and global structure can be preserved.

SeedNE [52] captures the informativeness of each vertex, and a self-paced informativeness-aware sampling strategy is introduced based on the informativeness. With the sampling strategy, SeedNE is able to select informative vertices to train model parameters.

TADW [30] proves that DeepWalk is equivalent to matrix factorization, then employs this matrix factorization model to incorporate text information of vertices into network embedding.

LANE [13] jointly projects an attributed network and labels into a unified embedding space by extracting their correlations. For a fair comparison, we employ *LANE_w/o_Label*, which only leverages the attributed network without labels.

GraphSAGE [11] is an inductive framework that leverages text information of vertices to generate the network representations by sampling and aggregating features from a vertex's local neighbors. In this paper, we employ the mean operator as the aggregator function, where the elementwise mean of the vectors in sampling neighbors is taken.

4.2. Datasets

Cora [53] contains 2708 machine learning papers from 7 classes and 5429 links among papers. Each vertex denotes a paper, and the citation relationships among documents form a typical complex network.

Citeseer [54] is a citation network as well, which consists of 3312 scientific publications classified into 6 classes with 4732 links among them. Similar to Cora, the links represent the citation between publications.

Wiki [55] has 2405 vertices, 17981 links, and 19 classes. Vertices in the network denote web pages of Wikipedia, and links among vertices are hyperlinks in the web pages. It is extremely dense compared with the other two datasets.

DBLP provides a comprehensive list of research papers in computer science. In order to facilitate subsequent network visualization, we construct an undirected paper citation network with papers from three different publication divisions in the DBLP dataset: Information Sciences, ACM Transactions on Graphics and World Wide Web. Note that vertices with degree less than 3 are filtered out, and the final network contains 1926 vertices and 10 590 edges. The title of a paper is used as the attribute information of a vertex, and there are 3624 different words after removing the stop words. DBLP dataset is used for network visualization because of its fewer classes.

For Cora and Citeseer, we employ the title and abstract contents of each paper as the attribute information, and the content in web pages is treated as the attribute features for Wiki. For text information of the above datasets, we remove stop words and words with document frequency less than 10. The statistics of datasets are summarized in Table 1. In the case of COANE, the text information of a vertex can be directly used for the training of the topic model, while it is transformed into a binary vector through the bag-of-words model for TADW, LANE and GraphSAGE.

Table 1
Statistics of datasets used in our experiments.

	# Vertices	# Edges	# Attributes	# Labels
Cora	2708	5429	1433	7
Citeseer	3312	4732	3703	6
Wiki	2405	17 981	4973	19
DBLP	1926	10 590	3624	3

4.3. Parameter settings

We uniformly set the embedding dimension $d = 128$ for all the above datasets. For DeepWalk, the number of walks γ is set 20, the walk length l is set 40, and the window size w is set 10. For M-NMF, we set $\alpha = 1$, $\beta = 0.5$ and k the true number of communities. We employ a 4-step transition probability matrix in HOPE. For SeedNE, the number of negative context vertices is set as 1, the batch size is set as 100 and the training epochs as 1000. The hyperparameter λ in TADW which indicates the weight of regularization term is set 0.2. For GraphSAGE, we set $K = 2$ with neighborhood sample sizes $S_1 = 25$ and $S_2 = 10$. For LANE, the weight α_1, which controls the contribution of attributes, is set 1. In order to provide a fair comparison, the parameter settings used for COANE are in line with values used for DeepWalk. The hyperparameters of the topic model are set to the empirical values that $\alpha = 50/k + 1$ and $\beta = 200/w$, in which k is the number of communities, and w is the size of vocabulary. For all datasets, we set k to the true number of communities. Besides, $M_m = 0.25$ and $M_s = 0.3$ are the results we found by trial and error.

4.4. Network visualization

Visualizations are indispensable for high-dimensional data analysis, which can intuitively reveal the intrinsic structure of data. The DBLP network is represented as low-dimensional vectors with different embedding models, and then these vectors of vertices are further mapped into a two-dimensional space using t-SNE [56].

Fig. 2 shows the visualization of network representations obtained from different algorithms. Since the titles of papers, which usually consist of less than 10 words, are taken as the attribute information, it is a great challenge for attributed network embedding algorithms to extract features from these short texts. For both DeepWalk and TADW, papers are roughly grouped by their publication divisions, though they are not linearly separable and the red vertices are divided into several parts. The visualization results of HOPE and LANE are not very meaningful, in which papers from different divisions are clustered together. Due to the utilization of community information, the performance of M-NMF is better. However, papers in the same divisions disperse into small clusters, and the compactness within a division is extremely low. SeedNE forms a more compact structure. Most similar vertices are pushed together, but some vertices from different classes are mixed with each other, especially for vertices in the center of the figure. GraphSAGE cannot well distinguish papers from Information Sciences and World Wide Web because the citations among them are frequent, and GraphSAGE just aggregate features from a vertex's local neighbors without considering their communities. Our proposed CONE and COANE achieve more compact and separated clusters compared with baseline methods. Initially, the obtained network representations are similar to those learned by TADW, then random walks within communities are employed to make vertices within the same community closer to each other. We can observe that the visualization of CONE has three clusters with clear boundaries among them, and the margin is larger between blue and green clusters in COANE. Intuitively, this experiment demonstrates that the margin-based random walk can obtain more meaningful and robust representations.

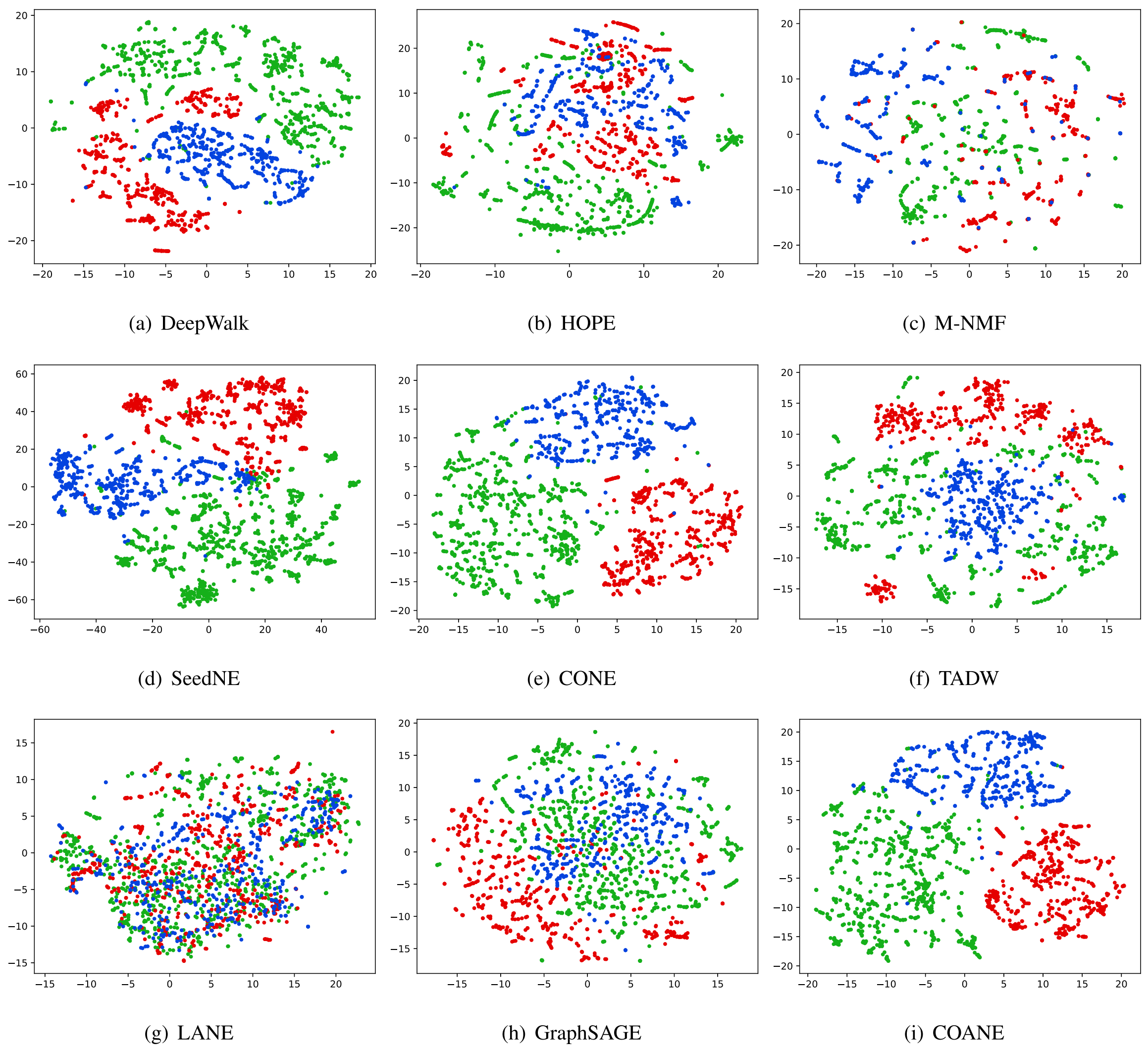


Fig. 2. Visualization of the DBLP network. Each point represents one paper and is mapped to the 2-D space by different network embedding algorithms. Colors of vertices indicate different publication divisions. Green: "Information Science", red: "ACM Transactions on Graphics", blue: "World Wide Web". (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

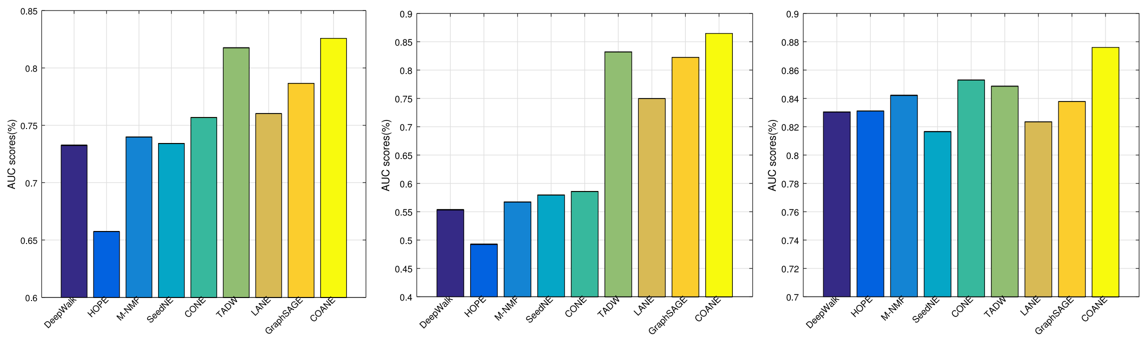
4.5. *Link prediction*

Link prediction is a supervised learning task that attempts to predict whether there is a link between pairwise vertices. On the premise that the residual network is connected, 50% of edges in the original network are removed randomly to evaluate the performance of models. We adopt a standard evaluation metric Area Under Curve (AUC) score [57], which indicates the probability that the potentially connected vertices are more similar than irrelevant ones.

It is observed in Fig. 3 that our proposed COANE consistently outperforms all the other baselines on these three datasets, and the attribute-free version CONE achieves superior performance as well. HOPE has a poor performance, and so does LANE when compared with other text-associated algorithms. TADW is second only to our proposed algorithm, because random walk based methods can well explore the network structure. However, DeepWalk is an algorithm that samples on the graph, which has a problem of insufficient sampling on dense networks. Due to the simplicity of the original random walk strategy and the limitation of the bag-of-words model, DeepWalk and TADW are not so effective on link prediction when they deal with dense networks with rich text information, such as Wiki. Similarly, the self-paced sampling strategy of SeedNE is more suitable for sparse networks, since it is able to adaptively capture the informativeness of each vertex, which is hard to distinguish on dense networks. The experimental results demonstrate the superior performance of our model, which fully utilizes the network topological structure and attribute features, especially when it is adopted to dense networks.

4.6. *Vertex classification*

Vertex classification is used to evaluate the quality of the obtained representations, where L2-regularized logistic regression [58] is employed as a supervised classifier, and the training ratio varies from 10% to 90%. In the experiments, we randomly select labeled vertices as training data and the remaining vertices as test data. We consider precision, recall, Micro-F1 score and Macro-F1 score as evaluation metrics in our experiments. Suppose the number of samples classified correctly into a class is True Positive (TP), the number of samples classified incorrectly into this class is False Positive (FP), and the number of samples in this class classified into other classes is True Negative (TN). The precision and recall are defined as TP/(TP+FP) and TP/(TP+TN), respectively.-F1 score is a weighted combination of precision and



(a) Cora (b) Citeseer (c) Wiki

Fig. 3. AUC scores of different methods on link prediction tasks.

recall, which is defined as:

$$F1\text{-}score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (7)$$

Thereby, Macro-F1 score is calculated by averaging F1 score of all categories, while Micro-F1 score is computed by averaging the precision and recall of all instances. We repeat the trial for 10 times and the average classification accuracy with different training ratio on these three networks is shown in Tables 2–13.

Tables 2–5 show the evaluation results of different algorithms on Cora, where CONE is considerably better than DeepWalk, HOPE, M-NMF and SeedNE. Note that, the essential difference between CONE and DeepWalk is the utilization of community information and margin-based random walk during the procedure of sequence generation. CONE outperforms DeepWalk by relatively 3% when using 90% of labeled vertices for training, which demonstrates the necessity of introducing global community structure to network embedding. COANE also shows competitive performance compared with the other text-associated algorithms by learning representations jointly from both network topological structure and attribute information.

Tables 6–9 indicate the evaluation results of different algorithms on Citeseer. The Citeseer network is extremely sparse and most vertices have only one edge connected to them. When 10% or 20% training data is used, TADW has the highest average accuracy. As the training ratio increases, COANE performs slightly better than TAWD when the training ratio is larger than 30%. Our method follows the similar random walk strategy with DeepWalk in the early stage. What the topic model needs are a large number of different sequences, so that it can learn the community-vertex distribution. Due to the sparsity of the network, many partly identical sequences could be formed in random walk procedure. However, vertices within a community are densely connected but those between different communities are linked sparsely, and the random walk sequences are prone to access vertices within the same community, so the generated sequences are sufficiently diverse. In this way, vertices of the same community are more likely to appear in a sequence, and the LDA model is able to capture the co-occurrence probability without losing much accuracy.

Tables 10–13 present the evaluation results on Wiki. Wiki is very dense compared with Cora and Citeseer, hence DeepWalk and TADW cannot walk over the whole network and sample enough sequences to learn proximities among vertices. Nevertheless, it is a benefit for our algorithm. As we discussed above, there will be sufficient different sequences generated for the topic model, and these sequences serve to form more precise community boundaries. The scope of random walks is limited and the model can focus on preserving the structure within communities, thus the problem of insufficient sampling in the dense network is solved. It can be reflected in Tables 10–13 that COANE significantly improves the precision, recall and F1-score by 5%, which verifies our ability to fully explore dense networks with rich text.

4.7. Parameter sensitivity

We measure the effect of two key parameters M_m and M_s of COANE, which control the process of the random walk. We fix the training ratio to 50% and display the average accuracy of vertex classification and AUC scores of link prediction with different M_m and M_s, as shown in Fig. 4. In addition, we measure the effect of the number of communities k, as shown in Fig. 5. Considering that the results of different networks show similar tends, we take Wiki as an example.

As mentioned before, the two parameters M_m and M_s are uniformly varied from 0 to 1. The results of both vertex classification and link prediction task achieve the best when M_m and M_s are about 0.3, because the topic model requires sufficient training data to detect communities. If the value of M_m is too small, the margin-based random walk procedure will start at an early stage. In this case, there are not enough sequences generated for LDA, and the communities detected may be inaccurate or even wrong. However, the effect of communities will decrease if margins appear too late, and our proposed COANE will be equivalent to DeepWalk when M_m reaches 1. The same is true with M_s, which also needs to be set modest to avoid M_{max} increasing too fast or too slow.

We also measure the effect of the number of communities k, which is shown in Fig. 5. Parameter k varies from 14 to 24 when M_m and M_s are randomly set to 0.2 and 0.3. We can see the curves are relatively stable for CONE and COANE, which indicates the number of communities k has only a slight influence on the performance of vertex classification and link prediction. Though the number of communities (topics) k affects the performance of the topic model, we have the network representations obtained before margins appear, which have preliminarily determined the network structure, and the gradually enlarged margin also make our results more robust even if k is a little large or small.

4.8. Efficiency evaluation

The proposed framework consists of a margin-based random walk procedure and the community detection with the statistical topic model. In this section, we evaluate the effect of different topic models, e.g. PLSA [37], LDA [22], and Gaussian LDA [59]. PLSA randomly chooses a topic from the distribution over topics and a word from the corresponding distribution over the vocabulary, and any common prior probability distribution is not

Table 2
Precision (%) of vertex classification on Cora.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	76.16	77.69	78.07	78.87	79.56	79.79	80.37	80.86	80.52
HOPE	76.56	77.60	78.05	79.60	79.62	80.98	81.01	80.18	80.86
M-NMF	77.21	79.29	79.82	80.90	81.84	82.06	81.76	83.17	81.64
SeedNE	76.85	78.14	78.89	80.49	80.59	80.96	81.25	82.30	80.89
CONE	**77.56**	**79.80**	**80.64**	**82.22**	**82.47**	**83.40**	**83.41**	**84.03**	**83.15**
TADW	78.30	82.26	82.31	83.00	83.75	84.33	85.18	84.82	84.84
LANE	77.10	80.32	80.93	81.63	83.34	83.04	84.48	83.84	83.36
GraphSAGE	78.27	81.08	82.00	83.18	83.76	83.69	84.54	85.04	84.59
COANE	**78.48**	**82.51**	**82.48**	**83.84**	**84.68**	**85.04**	**86.23**	**86.43**	**86.40**

Table 3
Recall (%) of vertex classification on Cora.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	72.90	74.77	75.79	76.66	76.86	77.19	78.54	78.18	78.92
HOPE	73.31	74.91	75.80	76.70	77.00	78.51	79.15	77.32	79.63
M-NMF	73.78	76.27	77.49	78.84	79.16	79.35	80.36	80.45	80.39
SeedNE	74.18	75.56	76.48	77.99	78.19	78.92	78.87	79.92	79.54
CONE	**74.51**	**76.63**	**78.41**	**79.52**	**80.21**	**81.32**	**81.76**	**81.42**	**81.63**
TADW	76.35	79.39	80.24	80.99	82.31	82.59	82.94	83.36	83.61
LANE	75.19	77.23	78.81	79.95	81.58	81.08	82.72	82.92	81.98
GraphSAGE	76.13	77.65	79.82	81.42	81.55	81.51	82.24	83.28	83.53
COANE	**76.79**	**79.52**	**80.28**	**82.12**	**82.94**	**82.92**	**83.99**	**84.97**	**85.20**

Table 4
Micro-F1 score (%) of vertex classification on Cora.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	75.60	77.31	78.11	78.72	79.14	79.12	80.27	80.47	80.87
HOPE	76.01	77.33	77.92	79.01	79.19	80.46	80.58	79.32	81.58
M-NMF	76.54	78.64	79.46	80.93	81.17	81.18	81.96	82.58	82.34
SeedNE	76.76	77.99	78.70	80.02	80.32	80.55	80.65	81.96	81.73
CONE	**77.27**	**79.19**	**80.63**	**81.70**	**82.16**	**83.03**	**83.20**	**83.75**	**83.80**
TADW	78.63	81.69	82.53	83.40	84.08	84.73	84.91	85.06	85.09
LANE	77.44	79.80	81.12	82.31	83.31	83.30	84.52	84.48	83.72
GraphSAGE	78.30	80.31	82.08	83.62	83.49	83.90	84.27	85.20	84.95
COANE	**79.12**	**81.90**	**82.70**	**84.23**	**84.61**	**85.08**	**85.91**	**86.59**	**86.82**

Table 5
Macro-F1 score (%) of vertex classification on Cora.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	74.13	76.05	77.01	77.85	77.88	78.09	79.33	79.21	79.29
HOPE	74.64	76.16	76.71	77.95	78.25	79.45	79.71	78.32	80.21
M-NMF	74.95	77.50	78.37	79.95	80.12	80.33	81.25	81.52	80.83
SeedNE	75.01	76.63	77.40	79.02	79.23	79.81	79.69	80.92	80.34
CONE	**75.78**	**77.79**	**79.38**	**80.48**	**81.13**	**82.03**	**82.49**	**82.58**	**82.48**
TADW	76.85	80.57	81.16	82.10	83.05	83.36	83.82	83.77	83.94
LANE	75.72	78.46	79.79	80.88	82.41	81.81	83.59	83.39	82.29
GraphSAGE	76.85	79.07	80.93	82.27	82.61	82.65	83.15	84.05	83.79
COANE	**77.50**	**80.67**	**81.34**	**82.78**	**83.41**	**83.62**	**84.71**	**85.40**	**85.51**

Table 6
Precision (%) of vertex classification on Citeseer.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	52.21	54.46	56.01	56.77	57.32	57.97	57.96	58.60	58.51
HOPE	48.83	50.62	51.14	51.72	51.67	51.89	51.86	52.94	53.14
M-NMF	52.88	55.49	57.08	57.03	58.34	58.95	58.91	59.35	59.50
SeedNE	53.48	55.75	56.98	57.51	57.49	58.04	58.90	59.28	59.44
CONE	**55.14**	**56.43**	**57.88**	**59.01**	**59.36**	**59.57**	**59.61**	**60.52**	**61.23**
TADW	**61.18**	**62.34**	63.39	64.26	64.48	65.07	64.88	65.00	65.46
LANE	50.02	55.69	58.91	59.95	60.79	62.68	62.19	61.48	62.86
GraphSAGE	57.42	60.57	62.18	62.70	63.19	64.64	65.80	65.36	64.71
COANE	59.88	62.24	**63.77**	**64.33**	**64.78**	**65.21**	**67.02**	**67.02**	**68.31**

specified. However, the two probability contributions in LDA are assumed to be multinomial and share the common Dirichlet prior. Gaussian LDA replaces LDA's parameterization of topics as categorical distributions over latent words with multivariate Gaussian distributions on the embedding space, which facilitate the model to group words that are priori known (word embeddings) to be semantically related into topics. The random walk based methods calculate vertex embeddings with SkipGram, thus it does not introduce additional time consumption. In order to avoid the influence of textual information, we compare the Micro-F1 score

Table 7
Recall (%) of vertex classification on Citeseer.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	50.39	52.25	54.21	54.66	56.13	56.09	55.66	56.29	55.91
HOPE	46.30	48.27	49.20	49.75	50.34	50.13	49.83	50.48	50.35
M-NMF	50.86	53.80	55.20	55.06	56.67	56.84	57.22	56.60	56.82
SeedNE	51.19	53.74	55.02	56.00	55.96	55.98	56.82	56.50	56.91
CONE	**52.41**	**54.23**	**55.87**	**57.43**	**57.57**	**57.58**	**58.05**	**58.28**	**58.55**
TADW	**60.71**	**61.90**	62.84	63.64	63.59	64.45	64.12	64.03	65.24
LANE	49.92	55.36	58.26	59.32	60.12	61.50	60.60	61.17	61.98
GraphSAGE	57.41	59.98	61.60	61.91	62.94	63.62	64.60	65.13	64.26
COANE	59.36	61.10	**63.40**	**63.88**	**64.08**	**64.59**	**65.58**	**66.05**	**67.92**

Table 8
Micro-F1 score (%) of vertex classification on Citeseer.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	54.16	56.61	58.43	59.50	60.28	60.64	60.34	60.84	60.42
HOPE	50.42	52.65	53.51	54.15	54.53	54.49	54.18	55.14	55.32
M-NMF	54.79	57.97	59.18	59.86	60.94	61.40	61.40	61.59	61.73
SeedNE	55.21	57.90	59.24	60.29	60.30	60.53	61.31	61.10	61.28
CONE	**56.53**	**58.80**	**60.29**	**61.65**	**62.06**	**62.09**	**62.21**	**62.64**	**63.08**
TADW	**65.66**	**66.74**	67.56	68.31	69.06	69.38	68.91	69.38	70.18
LANE	54.67	59.94	62.96	64.44	65.36	66.56	65.92	66.38	67.01
GraphSAGE	62.09	64.53	66.68	67.02	68.09	68.70	69.94	70.29	69.17
COANE	64.33	66.10	**67.91**	**68.75**	**69.21**	**69.61**	**70.76**	**71.53**	**72.68**

Table 9
Macro-F1 score (%) of vertex classification on Citeseer.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	50.42	52.66	54.62	54.77	55.87	56.37	55.78	55.91	55.81
HOPE	46.55	48.53	49.49	50.00	50.57	50.19	50.06	50.23	50.27
M-NMF	50.61	54.08	55.13	55.16	56.63	57.19	57.22	57.16	56.78
SeedNE	51.34	53.96	55.16	55.78	55.95	55.76	57.05	56.42	56.29
CONE	**52.39**	**54.54**	**56.30**	**57.22**	**57.87**	**57.53**	**57.45**	**58.27**	**58.58**
TADW	**60.39**	**61.82**	62.31	63.20	63.16	63.86	63.86	63.49	64.27
LANE	49.45	55.23	57.58	59.37	59.48	61.19	60.60	60.72	61.09
GraphSAGE	56.70	59.33	61.30	61.88	62.17	63.24	64.54	64.30	63.81
COANE	58.63	61.38	**62.97**	**63.70**	**63.52**	**64.11**	**65.35**	**65.92**	**67.33**

Table 10
Precision (%) of vertex classification on Wiki.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	53.72	55.68	56.38	59.98	61.46	61.39	61.26	60.07	60.27
HOPE	55.34	56.61	57.76	62.18	63.25	62.51	62.16	59.54	58.90
M-NMF	55.59	56.43	58.08	61.53	61.84	62.41	62.50	59.98	61.04
SeedNE	53.34	53.69	55.76	59.57	61.50	60.14	61.45	60.05	59.84
CONE	**57.48**	**57.78**	**59.02**	**63.40**	**64.47**	**64.38**	**65.15**	**63.21**	**63.81**
TADW	57.42	61.15	64.48	64.19	63.51	63.80	65.18	65.98	67.32
LANE	57.34	62.98	63.29	66.03	63.80	62.59	64.12	65.10	65.87
GraphSAGE	55.84	61.93	65.88	65.93	65.26	65.82	67.94	66.30	67.79
COANE	**58.68**	**64.92**	**69.41**	**71.17**	**69.08**	**70.74**	**72.01**	**73.54**	**73.35**

Table 11
Recall (%) of vertex classification on Wiki.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	48.89	51.29	52.60	53.18	54.20	55.32	55.46	55.20	56.78
HOPE	50.11	52.04	53.75	55.68	56.54	56.32	56.23	54.69	54.97
M-NMF	50.41	51.21	53.87	54.44	54.98	56.50	56.81	55.68	57.21
SeedNE	48.18	48.90	51.86	52.96	55.03	54.71	55.66	55.64	56.37
CONE	**52.05**	**53.13**	**54.93**	**56.83**	**57.62**	**58.27**	**59.53**	**58.82**	**60.23**
TADW	53.59	54.48	57.58	57.54	57.68	58.00	59.54	60.77	65.27
LANE	53.38	56.17	55.42	59.12	58.09	57.18	58.09	60.25	63.77
GraphSAGE	52.59	55.49	58.60	59.20	59.82	60.24	62.00	61.49	65.87
COANE	**54.80**	**58.21**	**62.00**	**64.12**	**63.41**	**64.91**	**66.24**	**68.54**	**71.55**

and time consumption of the attribute-free framework leveraging these models, named CONE-PLSA, CONE-LDA and CONE-GLDA, with those of the random walk sampling algorithm, DeepWalk. The experiment is performed on a NVIDIA Tesla V100 GPU. We fix the training ratio to 50% and display the average Micro-F1 score and time consumption in Table 14.

The proposed framework with different topic models achieves higher Micro-F1 score compared with DeepWalk, whereas these

Table 12
Micro-F1 score (%) of vertex classification on Wiki.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	58.42	61.99	63.38	63.54	64.47	65.27	65.92	66.07	66.49
HOPE	59.86	62.84	64.41	65.61	66.31	66.59	66.87	65.75	64.93
M-NMF	60.33	62.30	64.62	64.83	64.89	66.69	66.95	66.27	67.04
SeedNE	58.04	59.75	62.41	62.89	64.88	64.47	66.28	66.07	66.16
CONE	**61.64**	**63.68**	**65.47**	**66.85**	**67.50**	**68.14**	**69.55**	**69.27**	**69.83**
TADW	64.67	66.41	68.52	69.02	69.60	70.44	71.04	71.88	73.83
LANE	64.43	68.16	66.95	70.91	70.41	69.37	69.42	71.26	72.31
GraphSAGE	63.29	67.61	69.73	70.78	71.62	72.34	73.45	72.82	74.36
COANE	**65.60**	**70.16**	**73.08**	**75.47**	**75.30**	**77.06**	**77.32**	**79.41**	**79.61**

Table 13
Macro-F1 score (%) of vertex classification on Wiki.

% Labeled vertices	10%	20%	30%	40%	50%	60%	70%	80%	90%
DeepWalk	49.09	51.35	52.80	54.10	55.44	56.11	56.04	56.16	56.34
HOPE	50.04	52.67	53.59	56.40	57.23	57.43	56.90	55.71	54.80
M-NMF	51.14	51.89	54.08	55.65	55.91	57.51	57.51	56.26	57.24
SeedNE	48.23	49.33	52.02	53.89	55.51	55.50	56.67	55.77	56.06
CONE	**52.33**	**53.63**	**55.15**	**57.82**	**58.68**	**59.30**	**60.10**	**59.47**	**60.14**
TADW	53.91	55.16	58.00	58.46	58.06	58.85	60.40	61.25	64.85
LANE	53.90	56.67	56.92	60.27	59.07	57.80	58.89	60.63	63.18
GraphSAGE	52.73	56.21	59.67	60.33	60.24	60.39	63.06	62.08	65.49
COANE	**55.18**	**59.34**	**63.00**	**65.10**	**64.11**	**65.42**	**67.19**	**68.97**	**70.71**

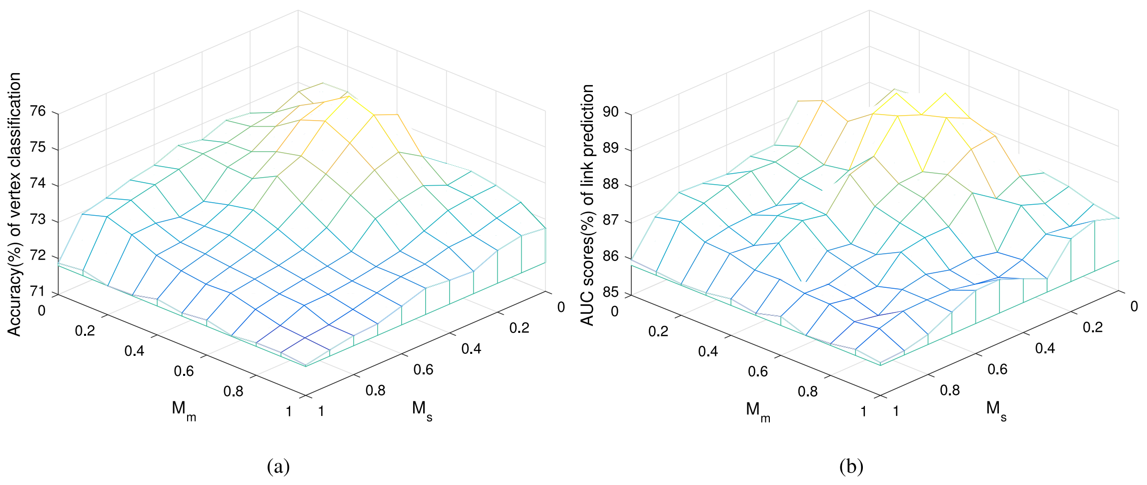


Fig. 4. The effect of parameters M_m and M_s on Wiki. (a) Accuracy of vertex classification tasks. (b) AUC scores of link prediction tasks.

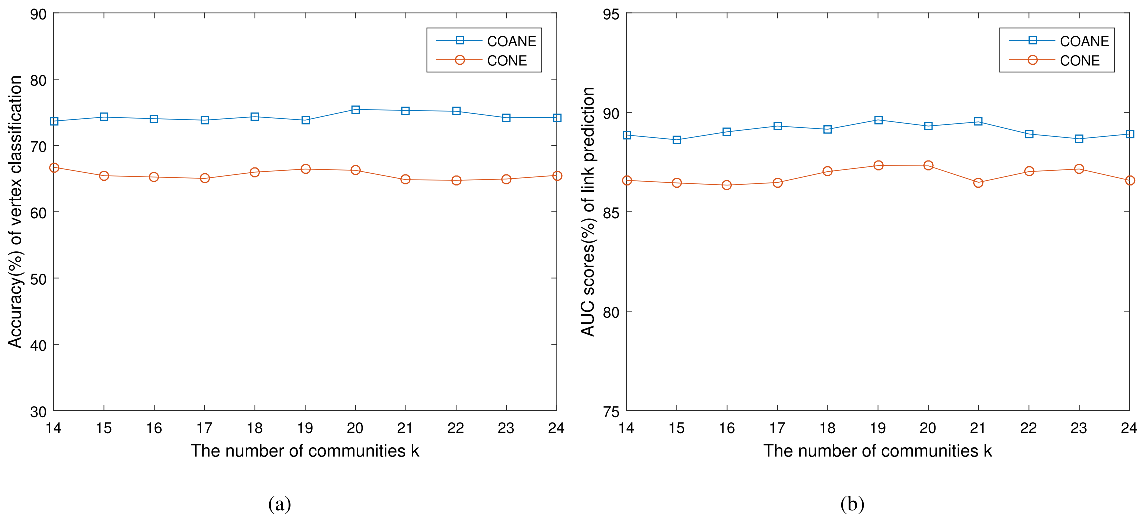


Fig. 5. The effect of parameters k on Wiki. (a) Accuracy of vertex classification tasks. (b) AUC scores of link prediction tasks.

Table 14
Efficiency (F1 score / time consumption) of different topic models.

Number of walks γ	5	10	20	30
DeepWalk	78.08 / 25.3 s	78.67 / 46.0 s	79.20 / 93.8 s	80.63 / 138.1 s
CONE-PLSA	80.74 / 44.5 s	81.19 / 91.2 s	81.81 / 174.7 s	82.58 / 254.6 s
CONE-LDA	81.52 / 54.8s	81.71 / 112.9 s	82.93 / 218.5 s	83.78 / 288.4 s
CONE-GLDA	82.15 / 176.0 s	82.64 / 326.5 s	83.46 / 517.3 s	83.95 / 748.7 s

community-oriented algorithms require more additional computation on the training of topic models. Specifically, the classification accuracy of topic models is positively correlated with the computation time and the number of walks γ. Leveraging vertex embeddings calculated by SkipGram, CONE with Gaussian LDA outperforms DeepWalk by nearly 4%, which demonstrates the superior performance of the margin-based random walk strategy.

5. Conclusion

In this work, we develop a novel community-oriented method for attributed network embedding. We propose a margin-based random walk strategy to jointly preserve the low-order proximity and community information, and it solves the problem that DeepWalk cannot sample sufficient walks on dense networks. The proposed COANE overcomes the shortcomings of previous works which learn representations from attribute information and network structure independently, and achieves high performance on text attribute embedding. Experimental results on real-world networks show the effectiveness and robustness of COANE while compared with six state-of-the-art baselines.

We will explore the following directions in the future:

(1) We have investigated the effectiveness of COANE on several real-world networks, and we will make it scalable for large-scale networks by parallelizing our method subsequently. Meanwhile, we plan to improve the selection algorithm to speed up the procedure of margin-based random walk.

(2) COANE explores the latent community distribution from the network topological structure and text features. However, real-world networks usually consist of different types of vertices, relations and explicit attribute information [60]. Thus, we would also like to extend the proposed method to heterogeneous networks in the future.

CRediT authorship contribution statement

Yuan Gao: Formal analysis, Investigation, Methodology, Software, Writing - original draft. **Maoguo Gong:** Conceptualization, Funding acquisition, Project administration, Resources, Methodology, Supervision. **Yu Xie:** Data curation, Validation, Visualization. **Hua Zhong:** Writing - review & editing.

Acknowledgements

This work was supported by the National key research and development program of China (Grant no. 2017YFB0802200), the National Natural Science Foundation of China (Grant no. 61772393), and the Key research and development program of Shaanxi Province (Grant no. 2018ZDXM-GY-045).

References

[1] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: A survey, Knowl.-Based Syst. 151 (2018) 78–94.
[2] H. Cai, V.W. Zheng, K. Chang, A comprehensive survey of graph embedding: problems, techniques and applications, IEEE Trans. Knowl. Data Eng. 30 (9) (2018) 1616–1637.
[3] I. Brugere, B. Gallagher, T.Y. BergerWolf, Network structure inference, a survey: Motivations, methods, and applications, ACM Comput. Surv. 51 (2) (2018) 24.
[4] F. Huang, X. Zhang, J. Xu, C. Li, Z. Li, Network embedding by fusing multimodal contents and links, Knowl.-Based Syst. 171 (2019) 44–55.
[5] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, J. Am. Soc. Inf. Sci. Technol. 58 (7) (2007) 1019–1031.
[6] S. Bhagat, G. Cormode, S. Muthukrishnan, Node classification in social networks, in: Social Network Data Analytics, 2011, pp. 115–148.
[7] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 1067–1077.
[8] S. Cao, W. Lu, Q. Xu, Grarep: Learning graph representations with global structural information, in: Proceedings of the 24th ACM International Conference on Information and Knowledge Management, 2015, pp. 891–900.
[9] B. Perozzi, R. AlRfou, 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.
[10] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.
[11] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems, 2017, pp. 1024–1034.
[12] H. Gao, H. Huang, Deep Attributed Network Embedding, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3364–3370.
[13] X. Huang, J. Li, X. Hu, Label informed attributed network embedding, in: Proceedings of the 10th ACM International Conference on Web Search and Data Mining, 2017, pp. 731–739.
[14] J. Liang, P. Jacobs, J. Sun, S. Parthasarathy, Semi-supervised embedding in attributed networks with outliers, in: Proceedings of the SIAM International Conference on Data Mining, 2018, pp. 153–161.
[15] X. Huang, J. Li, X. Hu, Accelerated attributed network embedding, in: Proceedings of the SIAM International Conference on Data Mining, 2017, pp. 633–641.
[16] C. Li, Z. Li, S. Wang, Y. Yang, X. Zhang, J. Zhou, Semi-Supervised Network Embedding, in: Proceedings of the 22nd International Conference on Database Systems for Advanced Applications, 2017, pp. 131–147.
[17] L. Yang, X. Cao, D. He, C. Wang, X. Wang, W. Zhang, Modularity based community detection with deep learning, in: Proceedings of the 25th International Joint Conference on Artificial Intelligence, 2016, pp. 2252–2258.
[18] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, S. Yang, Community preserving network embedding, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, 2017, pp. 203–209.
[19] M.M. Keikha, M. Rahgozar, M. Asadpour, Community aware random walk for network embedding, Knowl.-Based Syst. 148 (2018) 47–54.
[20] S. Fortunato, M. Barthelemy, Resolution limit in community detection, Proc. Natl. Acad. Sci. USA 104 (1) (2007) 36–41.
[21] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Inf. Process. Manag. 24 (5) (1988) 513–523.
[22] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (1) (2003) 993–1022.
[23] J. Xu, X. Liu, Z. Huo, C. Deng, F. Nie, H. Huang, Multi-class support vector machine via maximizing multi-class margins, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3154–3160.
[24] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.
[25] J. Bütepage, M.J. Black, D. Kragic, H. Kjellström, Deep representation learning for human motion prediction and classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1591–1599.
[26] X. Du, J.J.-Y. Wang, Support image set machine: Jointly learning representation and classifier for image set classification, Knowl.-Based Syst. 78 (2015) 51–58.
[27] J. Li, J. Li, X. Fu, M.A. Masud, J.Z. Huang, Learning distributed word representation with multi-contextual mixed embedding, Knowl.-Based Syst. 106 (2016) 220–230.
[28] M. Janner, K. Narasimhan, R. Barzilay, Representation learning for grounded spatial reasoning, Trans. Assoc. Comput. Linguist. 6 (2018) 49–61.

[29] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
[30] C. Yang, Z. Liu, D. Zhao, M. Sun, E.Y. Chang, Network representation learning with rich text information, in: Proceedings of the 24th International Joint Conference on Artificial Intelligence, 2015, pp. 2111–2117.
[31] Z. Chen, T. Cai, C. Chen, Z. Zheng, G. Ling, SINE: Side Information Network Embedding, in: Proceedings of the 24th International Conference on Database Systems for Advanced Applications, 2019, pp. 692–708.
[32] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1225–1234.
[33] X. Wang, D. Jin, X. Cao, L. Yang, W. Zhang, Semantic Community Identification in Large Attribute Networks, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence, 2016, pp. 265–271.
[34] M. Li, J. Liu, P. Wu, X. Teng, Evolutionary network embedding preserving both local proximity and community structure, IEEE Trans. Evol. Comput. (2019) 1.
[35] J. Chen, Q. Zhang, X. Huang, Incorporate group information to enhance network embedding, in: Proceedings of the 25th ACM International Conference on Information and Knowledge Management, 2016, pp. 1901–1904.
[36] X. Xia, D. Lo, Y. Ding, J.M. AlKofahi, T.N. Nguyen, X. Wang, Improving automated bug triaging with specialized topic model, IEEE Trans. Softw. Eng. 43 (3) (2017) 272–297.
[37] T. Hofmann, Probabilistic latent semantic indexing, in: ACM SIGIR Forum, 1999, pp. 50–57.
[38] M. Steyvers, P. Smyth, M. RosenZvi, T. Griffiths, Probabilistic author-topic models for information discovery, in: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004, pp. 306–315.
[39] M. RosenZvi, T. Griffiths, M. Steyvers, P. Smyth, The author-topic model for authors and documents, in: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, 2004, pp. 487–494.
[40] Q. Mei, D. Cai, D. Zhang, C. Zhai, Topic modeling with network regularization, in: Proceedings of the 17th International Conference on World Wide Web, 2008, pp. 101–110.
[41] Y. Shi, M. Lei, H. Yang, L. Niu, Diffusion network embedding, Pattern Recognit. 88 (2019) 518–531.
[42] H. Chen, H. Yin, T. Chen, Q.V.H. Nguyen, W.-C. Peng, X. Li, Exploiting centrality information with graph convolutions for network representation learning, in: Proceedings of the 35th IEEE International Conference on Data Engineering, 2019, pp. 590–601.
[43] W. Zhao, H. Ma, Z. Li, X. Ao, N. Li, SBRNE: An Improved Unified Framework for Social and Behavior Recommendations with Network Embedding, in: Proceedings of the 24th International Conference on Database Systems for Advanced Applications, 2019, pp. 555–571.
[44] Q. Li, J. Zhong, Q. Li, Z. Cao, C. Wang, Enhancing network embedding with implicit clustering, in: Proceedings of the 24th International Conference on Database Systems for Advanced Applications, 2019, pp. 452–467.
[45] L. Wu, D. Wang, S. Feng, Y. Zhang, G. Yu, MDAL: Multi-task Dual Attention LSTM Model for Semi-supervised Network Embedding, in: Proceedings of the 24th International Conference on Database Systems for Advanced Applications, 2019, pp. 468–483.
[46] R. Krestel, P. Fankhauser, W. Nejdl, Latent dirichlet allocation for tag recommendation, in: Proceedings of the 3rd ACM Conference on Recommender Systems, 2009, pp. 61–68.
[47] L. Tang, H. Liu, Relational learning via latent social dimensions, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009, pp. 817–826.
[48] D. Whitley, A genetic algorithm tutorial, Stat. Comput. 4 (2) (1994) 65–85.
[49] J. Tang, Z. Meng, X. Nguyen, Q. Mei, M. Zhang, Understanding the limiting factors of topic modeling via posterior contraction analysis, in: Proceedings of the 31th International Conference on Machine Learning, 2014, pp. 190–198.
[50] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781.
[51] M. Ou, P. Cui, J. Pei, Z. Zhang, W. Zhu, Asymmetric transitivity preserving graph embedding, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1105–1114.
[52] H. Gao, H. Huang, Self-paced network embedding in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2018, pp. 1406–1415.
[53] A.K. McCallum, K. Nigam, J. Rennie, K. Seymore, Automating the construction of internet portals with machine learning, Inf. Retr. 3 (2) (2000) 127–163.
[54] C.L. Giles, K.D. Bollacker, S. Lawrence, CiteSeer: An automatic citation indexing system, in: Proceedings of the 3rd ACM conference on Digital Library, 1998, pp. 89–98.
[55] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. EliassiRad, Collective classification in network data, AI Mag. 29 (3) (2008) 93.
[56] L. Maaten, G. Hinton, Visualizing data using t-SNE, J. Mach. Learn. Res. 9 (11) (2008) 2579–2605.
[57] J.A. Hanley, B.J. McNeil, The meaning and use of the area under a receiver operating characteristic (ROC) curve, Radiology 143 (1) (1982) 29–36.
[58] R. Fan, K. Chang, C. Hsieh, X. Wang, C. Lin, LIBLINEAR: A library for large linear classification, J. Mach. Learn. Res. 9 (8) (2008) 1871–1874.
[59] R. Das, M. Zaheer, C. Dyer, Gaussian lda for topic models with word embeddings, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, 2015, pp. 795–804.
[60] L. Gui, Y. Zhou, R. Xu, Y. He, Q. Lu, Learning representations from heterogeneous network for sentiment classification of product reviews, Knowl.-Based Syst. 124 (2017) 34–45.