Article

A Hierarchical Generative Embedding Model for Influence Maximization in Attributed Social Networks

Luodi Xie, Huimin Huang *, Hong Shen * and Qing Du

School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou 510006, China.; xield@mail2.sysu.edu.cn (L.X.); shen3@mail.sysu.edu.cn (H.S.); School of Mathematics and Information Engineering, Wenzhou University of Technology, Wenzhou 325000, China. huanghm45@gmail.com (H.H.); School of Software, South China University of Technology, Guangzhou 510640, China. duqing@scut.edu.cn(Q.D.)

* Correspondence: huanghm45@gmail.com(H.H.); shen3@mail.sysu.edu.cn(H.S.)

Abstract: Nowadays, we use social networks such as Twitter, Facebook, WeChat and Weibo as means to communicate with each other and get to know others. Gradually social networks has become indispensable in our everyday life, and we cannot absolutely imagine what the daily life would be like without social networks. Through social networks, we can access friends' opinions and behaviors easily and are influenced by them in turn. Thus, an effective algorithm to find the top-K influential nodes (the problem of influence maximization) in the social network is critical for various downstream tasks such as viral marketing, anticipating natural hazards, reducing gang violence, public opinion supervision etc. Solving the problem of influence maximization in real-world propagation scenarios often involves estimating influence strength (influence probability between two nodes), which cannot directly observed. To estimate influence strength, conventional approaches propose various humanly-devised rules to extract features of user interactions, the effectiveness of which heavily depends on domain expert knowledge. Besides, they are often applicable for special scenarios or specific diffusion models. Consequently, they are difficult to be generalized into different scenarios, diffusion models and even domains. Inspired by the powerful ability of neural networks in the field of representation learning, we design a deep hierarchical network embedding model HGE to map nodes (with attributes) into latent space automatically. In general, HGE takes an attributed social network as the input for learning latent network representation of each node, incorporating hierarchical community structure, node attributes and general network structure into a unified deep generative framework. Then, with the leaned latent representation of each node, we propose a HGE-GA algorithm to predict influence strength and compute the top-K influential nodes through a greedy-based maximization algorithm. Extensive experiments on real-world attributed networks demonstrate the outstanding superiority of the proposed HGE model and HGE-GA algorithm compared with the state-of-the-art methods, verifying the effectiveness of the proposed model and algorithm.

Keywords: Influence maximization; Influence strength; Hierarchical network embedding; Community Structures; Attributed social networks

1. Introduction

Fueled by the spectacular growth of internet and internet of things, plenty of social networks such as Facebook, Twitter and WeChat have sprung up, changed the interaction modes between peoples, and accelerated development of viral marketing. Originated from the idea of word-to-mouth advertising, viral marketing takes advantage of trust among close social circles of friends, colleagues or families to promote a new product, i.e., friend relationship affects user making decision on item selection. Motivated by applications to the early viral marketing, a new study area of influence diffusion has thrived. Thereinto, the problem of influence maximization is to select a fixed size set of seed nodes in a network to maximize the influence spread, according to a specially designed influence diffusion model. Figure 1 gives A toy example of social influence. The nodes v_1,v_2,v_3 in black color are seed nodes which is initially active, and the nodes in gray color are newly activated by the seed nodes. In terms of viral marketing, for example, if user v_1,v_2,v_3 bought a product, their friends in social network will likely buy this product because of the friend-to-friend influence.

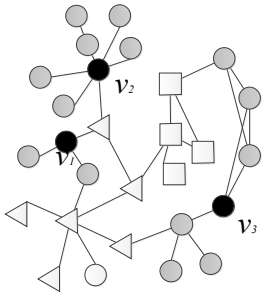

Figure 1. A toy example of social influence. The seed nodes are v_1,v_2,v_3 in black color, and the nodes influenced by the seed nodes are in gray color.

The applications of influence maximization is prevalent in real world such as viral marketing, anticipating natural hazards, reducing gang violence, public opinion supervision etc., whereas the time complexity of solving the problem of influence maximization is NP hard [1]. This inspires a lot of studies on influence diffusion and influence maximization algorithms. In early time, most studies have focused on the influence maximization algorithms themselves under general Independent Cascade (IC) diffusion model or general Linear Threshold (LT) diffusion model, including greedy-based algorithms [1–7] and heuristic-based algorithms [8–10]. In terms of running time, heuristic-based algorithms are generally more efficient than various greedy-based algorithms, but they do not have any theoretical guarantee.

As in the real world, influence propagation often involves latent variables that are not unobservable directly, including the influence probability between tow nodes etc., estimating influence probability between two nodes (or influence strength) is a fundamental problem for influence diffusion. Some models to estimate influence strength have been proposed, including topic model [11–13], probabilistic methods [14,15] and models based on deep learning [16,17]. Although existing models and methods have achieved a lot, they demonstrate a number of major drawbacks: (1) Conventional approaches propose virous humanly-devised rules to extract features of user interactions, the effectiveness of which heavily depends on domain expert knowledge. (2) They only consider general network structure and node attributes as factors underlying social network influence, but in the real world, the structure of large-scale network is more complex and often involves communities, community tensility and node depth in clustering tree. (3) They are often applicable for special scenarios or specific diffusion models, not for generalized conditions.

Like most scenarios in real-world propagation [14], influence probability of node pair in this paper is assumed to be unobservable directly. To predict the influence probability of node pair, we need to compute the similarity of node pair, with the motivation originated from the previous studies that the more equivalent the network structure and node attributes of two nodes are, the more likely they make similar judgments, even they having no direct connection reciprocally [18]. Inspired by the powerful ability of neural networks in the field of representation learning, we design a deep hierarchical network embedding model HGE to map nodes into latent space automatically, preserving network structures and node attributes as much as possible. The correspondence between the general attributed network and the hierarchical latent space is illustrated in Figure 2. Then, with the leaned latent representation of each node, we measure the similarity of node pair and regard it as influence probability between two nodes, since the underlying factors of influence probability are network structures and node attributes [18]. Next, the top-K influential nodes can be computed through a greedy-based maximization algorithm. The overview of the proposed method is illustrated in Figure 3.

Social networks usually have millions of users, with a large amount of user-related information, community structures, and complex hierarchical network structures. When embedding nodes into a low-dimensional vector space, in order to preserve network structures as much as possible, hierarchical structures should be taken into account. Furthermore, community is formed of nodes which have similar attributes while repel each other due to attribute differences measured by tensility of community. In other words, a community is a node set with tensility, and in order to capture this tensility, we embed a community as a Gaussian distribution. Specifically, we propose a hierarchical

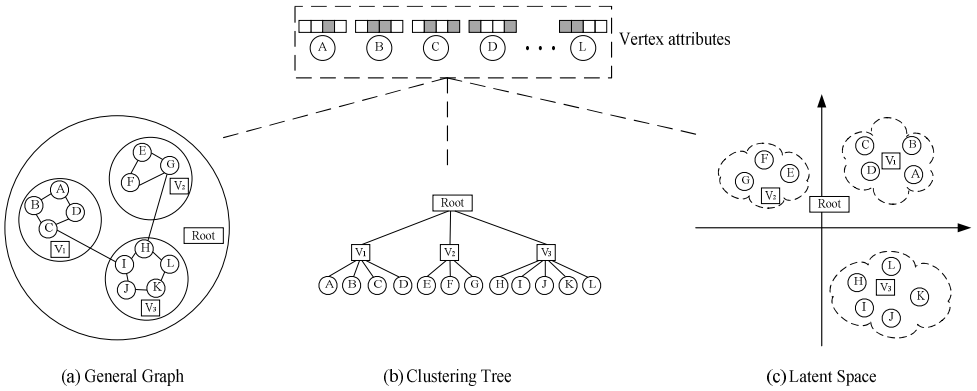

Figure 2. The correspondence between the general attributed network and the hierarchical latent space.

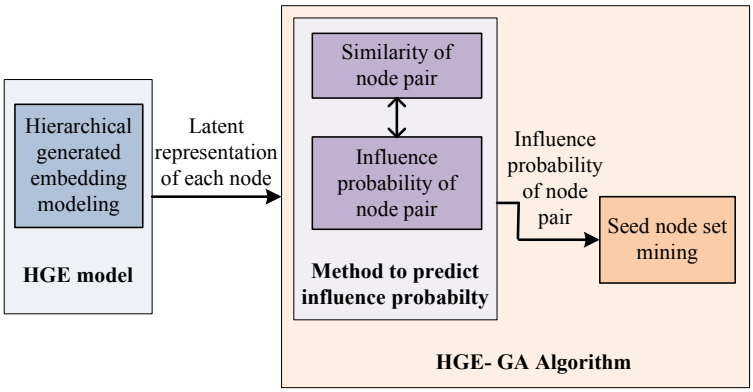

Figure 3. Overview of the system architecture.

generative embedding model, which integrates hierarchical community structure, node attributes and general network structure into a unified generative framework.

To summarize, we make the following contributions:

(1) We study the problem of incorporating hierarchical generative network embedding into influence strength prediction for the first time.

(2) In order to predict influence strength, we propose a novel model, HGE model, which automatically maps nodes into latent space, avoiding humanly devising rules to extract features of user interactions. Besides, the proposed HGE model integrates hierarchical community structure, node attributes and general network structure into a unified generative framework, that assures to preserve node characteristics and capture granularity of hierarchical characteristics as well as community tensility in attributed network effectively. Furthermore, we propose a new algorithm, HGE-GA, to predict influence strength and compute the top-K influential nodes effectively. The proposed HGE-GA Algorithm can be generalized to different domains.

(3) We evaluate our method on various datasets and tasks, and experimental results show that the proposed model and algorithm significantly outperform the state-of-the-art approaches.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 presents notations and problem formulation. Section 4 details the proposed method. Section 5 lists the experiment setup. Section 6 presents the experimental results & analysis. Section 7 concludes the paper.

2. Related Works

Influence Maximization. Kempe et al. [1] first formulate the problem of influence maximization into a discrete optimization problem, and prove the problem of influence maximization is NP-hard. By proving objective function of influence maximization problem is monotonous and submodular under general IC model and LT model, they propose a greedy algorithm with (1-1/e)-

approximation, iteratively selecting the nodes with the largest marginal gain. In recent years, the problem of influence maximization itself and the various extended problems of it have been studied extensively. In terms of the problem of influence maximization itself, with the scenarios of social networks becoming complex more and more partly due to the increasingly growing scales of social networks, it is of great significance to improve the performance of influence maximization algorithms. Researchers have proposed some algorithms to improve the effectiveness of influence maximization solving. Such influence maximization algorithms include various greedy-based algorithms, e.g., the efficient algorithm with the first provable approximation guarantees [1], CELF (Cost-Effective Lazy Forward selection) Algorithm [2], the improved greedy algorithm-NewGreedyIC [3], the static greedy algorithm-StaticGreedy [4], CELF++ (the improved CELF Algorithm) [5], Tim Algorithm with near-optimal time complexity and $(1-1/e-\varepsilon)$-approximate [6], IMM Alorithm in near-linear time and $(1-1/e-\varepsilon)$-approximate [7]. Another line of algorithms is heuristic-based methods [8–10] which emphasize analyzing and exploiting topology structure of network as well as utilizing specific scenario of social network. In terms of running time, they are generally more efficient than various greedy algorithms, but they can not provide any theoretical guarantee.

Influence Diffusion. The studies of influence diffusion focuses on the key factors for influence diffusion, such as diffusion model of single entity or multiple entities, influence probability between two nodes, which is more relevant to our work. Estimating influence probability between two nodes (or influence strength) is a fundamental problem for influence diffusion, since in real-world propagation scenarios, the variables such as influence strength, infection time etc. may be not directly observed. Some models to estimate influence strength have been proposed. One line of research is based on topic extraction and lays emphasis on incorporating topic model into influence diffusion. [11] proposes a probabilistic graph model to simulate the relationship between social conformity and social network influence. [12] proposes a topic model, TAP, to model topic-based social influence. [13] proposes a novel model, COLD, to model community level influence diffusion. Another line of studies utilize various probabilistic methods such as Bayesian inference [14], EM (Expectation Maximization) [15] etc. to stimulate influence diffusion and learn the parameters in the generative model. Recently, researchers endeavor to build models of influence prediction using deep learning, which can predict social influence automatically and no longer be limited to the expert knowledge when extracting user features or network features. [16] proposes DeepCas model to predict cascade, which utilizes the method of NLP(Natural Language Processing) representing a cascade graph as document, with node as word and cascade path as sentence, and solves the problem with RNN (Recurrent Neural Network). [17] proposes DeepInf model to predict social influence, which detects the dynamics of social influence and integrates network embedding and graph attention mechanism into the model.

To the best of our knowledge, the existing models to predict the influence probability between two nodes usually extract features of user interactions with virous humanly-devised rules, which heavily depends on domain expert knowledge. Besides, the underlying factors that affect social network influence, such as communities, community tensility and node depth in clustering tree etc., have not been taken into account in the existing models when computing the influence probability of node pair. Moreover, the existing models are often applicable for special scenarios or specific diffusion models, not for generalized conditions.

Network Embedding. Network embedding techniques aim at inferring representations, also called embeddings of entities in the networks. The basic idea of network embedding is to embed the nodes into a low-dimensional vector space in which the similarity between nodes can be measured and network structures can be preserved as much as possible. Many downstream learning tasks can benefit from this form of representations, such as link prediction[19,20], node classification [21,22], node clustering [23,24] and network visualization[25–27].

Existing models of network embedding are concerning about the technology of mapping or dimension reduction. Matrix factorization is an efficient way for dimension reduction, and numerous methods based on matrix factorization are proposed [28–30]. Yang et al. [30] prove that the essence of DeepWalk [21] is matrix factorization and they consider the text feature information in the process of Matrix factorization. GraRep [28] takes into account the global and local structure information, and this method can capture the relationships between remote nodes. NEU [29] has summarized

some embedding methods which can be regarded as matrix factorization. As a conclusion, if the matrix factorization can include higher-order information more accurately, it will bring better effect but the computational complexity will be higher too.

In order to obtain the effective low-dimensional embeddings, numerous algorithms have been proposed[20–22,28,31,32]. Inspired by Word2Vec [33], deep learning has been migrated to network embedding, DeepWalk[21], Node2Vec[20] and Line[25] extract a number of sequential nodes in the network by random walk. Nodes in a sequence are metaphorically equivalent to words in language models, and then Skip-Gram[33] is employed to obtain the embeddings of nodes. After obtaining the embeddings of nodes, the similarity of nodes can be calculated by the inner-product of their embeddings. Besides, LINE [25] uses first-order and second-order proximities and trains the model via negative sampling. SDNE [34] adopted a deep auto-encoder to preserve both the first-order and second-order proximities.

As one of the important features of social network, community structure is usually integrated into network embedding as auxiliary information[35–38]. M-NMF [35] incorporates the community structure into network embedding and, exploits the consensus relationship between the representations of nodes and community structure, and then jointly optimize NMF (Network Matrix Factorization) based representation learning model and modularity based community detection model in a unified framework. GraphAGM[36] combines AGM[39] and GAN[38] in a unified framework which can generate the most likely motifs with graph structure awareness in a computationally efficient way. NECS[37] preserves the high-order vertex proximity and incorporates the community structure of networks in vertex representation learning.

Hierarchical Network Embedding. Essentially, these methods mainly focus on preserving neighborhood information, or community structure at a particular scale. However, the community structures in complex networks are often hierarchical[40–44]. Inspired by the natural hierarchical structure of a galaxy with its stars, planets, and satellites, [41] propose the galaxy network embedding(GEM) model. GEM captures hierarchical structures by forming an optimization problem, including pairwise proximity, horizontal relationship and vertical relationship. In order to solve various defects caused by exponential decay of radius in GEM model, Long et al. [44] first apply subspace to hierarchical network embedding and propose SpaceNE model, which preserves proximity between pairwise nodes and between communities. [43] proposes NetHiex, a network embedding algorithm which can capture the different levels of granularity and alleviate data scarcity. Ma et al.[45] propose MINES framework which can embed multi-dimensional network with hierarchical structure to low-dimensional vector spaces, the learned representations for each dimension containing the hierarchical information. Nickel et al.[46] use hyperbolic space instead of Euclidean space to capture hierarchical structure in the network and the learned embeddings are difficult to be converted into the Euclidean space which is inappropriate for some downstream tasks of machine learning.

However, these hierarchical embedding algorithms suffer from some defects: (1) They embed each node or community as a vector in the latent space, which ignore the tensility of community. Essentially, a community is a set of nodes or smaller communities with similar attributes and tensility, and a simple vector can not represent this relationship of affiliation. The tensility of a community has been defined in definition 2. (2) They do not consider node depth in clustering tree when dealing with vertical relationship in hierarchical network. Thus, it is inappropriate to use the traditional method of measuring vertical relationship in hierarchical network to compute vector distance in latent space. The granularity of hierarchical structures should be taken into account and personalized method to measure vector distance in latent space should be designed. The granularity of hierarchical structures has been defined in definition 3.

3. Notations and Problem Formulation

In this section, we first introduce the definitions and notations, and then formulate the problem studied in this paper.

DEFINITION 1 (attributed network). An attributed network is an undirected graph $\mathcal{G} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{A}})$, where $\boldsymbol{\mathcal{V}}$ is the set of network nodes, $\boldsymbol{\mathcal{E}}$ is the set of edges and $\boldsymbol{\mathcal{A}} \in \{0,1\}^{|\boldsymbol{\mathcal{V}}| \times F}$ is binary-valued attribute matrix recording node-attribute associations with F being the number of

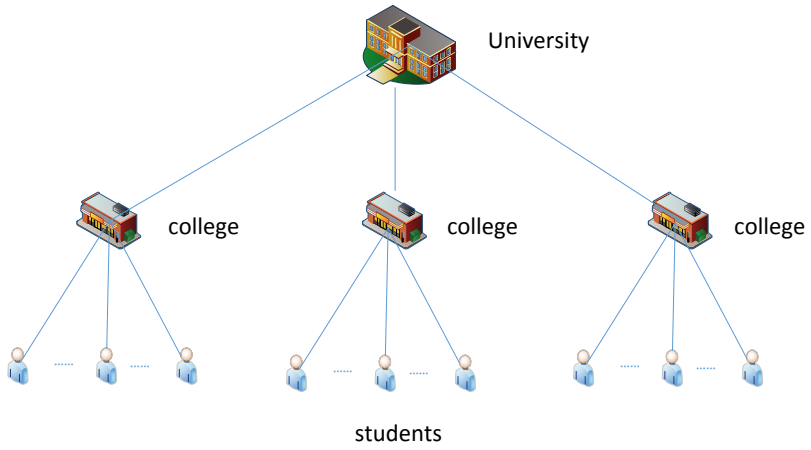

Figure 4. Illustration of vertexes at different levels having different granularity features

attributes. The element $A_{i,a} \in \boldsymbol{\mathcal{A}}$ is a binary-valued attribute weight indicating that attribute a associates with node i iff $A_{i,a} = 1$.

DEFINITION 2 (the tensility of community). A community is a set composed of nodes with similar attributes. These nodes form a community due to similar attributes and repel each other in the community due to differences in attributes. The tensility of the community is used to measure the difference of node attributes in the community.

DEFINITION 3 (the granularity of hierarchical characteristics). In hierarchical networks, vertexes at different levels have different granularity of characteristics. As shown in Figure 4, the university in the top level has the characteristics of name, address etc., and the colleges in the second level has the characteristics of majors, number of students etc. The characteristics of nodes at different levels have different granularity, while granularity of characteristics is same for the nodes at the same level.

The hierarchical clustering tree of G is denoted as T with a depth of L. $\boldsymbol{\mathcal{V}}^l$ is the vertex set at the l-th level of T and $v_i^l \in \boldsymbol{\mathcal{V}}^l$ is the i-th vertex at the l-th level. $Ch(v)$ denotes the child vertex set of the vertex v, f a(v) denotes the father vertex of v and $\Phi(v)$ denotes the embedding of vertex v. In our paper, we use 'vertex' to represent a user or community in the hierarchical network, while using 'node' to represent a user in the general network. Especially, when $l = L$, v_i^l represents a leaf vertex (user) at the bottom of the clustering tree. For ease of reference, we summarize the basic notations in Table 1.

Table 1: **Main notation used across the whole paper.**

Notation	Description
$\mathcal{G}$	a undirected graph
$\mathcal{V}$	set of nodes
$\mathcal{E}$	set of edges
T	a hierarchical clustering tree
L	the depth of the clustering tree
$Ch(v)$	the child vertex set of vertex v
$fa(v)$	the father vertex of v
v_i^l	the i-th vertex at the l-th level
$\Phi^{\|v}$	the embedding of vertex v

With the definitions and notations described above, we formulate the problem studied.

Problem 1. Node Embedding. Given an undirected graph $\mathcal{G} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{A}})$, the goal is to learn the latent representation of nodes with the mapping function Ξ:

$$\mathcal{G} \xrightarrow{\Xi} \Phi_{\mathcal{V}} \tag{1}$$

such that the information of hierarchical network structure, general network and node attributes can be preserved as much as possible by $\Phi_{\mathcal{V}} \in \mathcal{R}^{N\times D}$, where $\Phi_{\mathcal{V}}$ is latent embedding matrix for all nodes and D is the dimension size of the embeddings.

Problem 2. Influence Maximization in Attributed Social Networks. Given IC diffusion model, an undirected graph $\mathcal{G} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{A}})$ and the latent embedding matrix for all nodes $\Phi_{\mathcal{V}}$, the goal is to find the node set $\boldsymbol{\mathcal{S}}$ with an algorithm $\mathcal{F}$:

$$\mathcal{G}, \Phi_{\mathcal{V}} \xrightarrow{\mathcal{F}} \mathcal{S} \tag{2}$$

such that the expected spread of node set $\boldsymbol{\mathcal{S}}$, i.e., $\sigma(\boldsymbol{\mathcal{S}})$ is maximized and $|\boldsymbol{\mathcal{S}}| = K$. Expected spread of node set $\boldsymbol{\mathcal{S}}$ is defined as the total number of active nodes after the influence spread ends, including the newly activated nodes and the initially active nodes.

4. The Proposed Method

Like most scenarios in real-world propagation [14], influence probability of node pair in this paper is assumed to be unobservable directly. To predict the influence probability of node pair, we first embed the nodes into a low-dimensional vector space by jointly modeling horizontal constraint, vertical constraint, affiliation constraint and node attributes. After obtaining the embedding of each node, we use two-norm form to measure the similarity of node pair. Then we predict the influence probability of node pair with the measured similarity of node pair. We now describe the proposed method in the following subsections, including hierarchical generative embedding model § 4.1, learning procedure §4.2, HGE-GA Algorithm §4.3.

4.1. Hierarchical Generative Embedding Model

The proposed HGE model focuses on preserving the following characteristics of the network: (1) Horizontal structure characteristics. Nodes belonging to the same community are more similar than nodes belonging to different communities (in terms of blood relations, blood brothers are closer than Cousins). (2) Vertical structure characteristics. The degree of similarity between a node and its father is proportional to the number of its children. (3) Affiliation relationship characteristics. Users in the same community often have both similarities and differences between them, and differences between nodes are defined as tensility of node and should be preserved. (4) General network structure characteristics. In a general network, the more similar the nodes are, the closer they are in the embedding space. (5) Node attribute characteristics. Node attributes provide rich information of node characteristics, which should be taken advantage for preserving node characteristics in latent space.

We will detail HGE model from three aspects. The first one is how to preserve hierarchical network structure, the second one is how to capture general network structure properties, and the third one is how to acquire node attribute features.

4.1.1. Hierarchical Structure Properties

Similar to most embedding methods[20,21,25,26], we preserve the network structure through the distance of the vertex in the hidden space. A community is a node set with tensility, and in order to capture this tensility, we embed a community as a Gaussian distribution. The greater the variance, the greater tensility of the community, that is, the difference between nodes contained in the community will be greater. So we use KL distance to measure the similarity between communities. More specifically, suppose $\Phi_{v_1} = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$,$\Phi_{v_1} = \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$, the distance between v_1 and v_2 can be calculated in two ways as follows:

$$\begin{aligned}\delta(v_1, v_2) = D_{KL}(\mathcal{N}_1 \| \mathcal{N}_2) = \frac{1}{2}[tr(\Sigma_2^{-1}\Sigma_2) \\ + (\mu_1 - \mu_2)^T \Sigma_2^{-1}(\mu_1 - \mu_2) - L - log\frac{det(\Sigma_2)}{det(\Sigma_1)}].\end{aligned} \tag{3}$$

where v_1 and v_2 are communities, L is a constant.

$$\delta(v_1, v_2) = \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|_2 \tag{4}$$

where v_1 and v_2 are nodes.

Unlike normal networks, hierarchical networks contain a large number of hierarchical relationships. To capture this complex hierarchical relationship, we force the embedding of vertex to submit to two constraints, the horizontal and the vertical:

Horizontal constraint: Generally, community-based embedding methods[35,36] consider that the nodes belonging to the same community are more similar than those belonging to different communities. Thus, we intend to extend this constraint to hierarchical networks and name such constraint as horizontal constraint. The horizontal constraint can be defined as follows: For each layer l in clustering tree T, for all vertex-pair v_i^l, v_j^l at layer l with $fa(v_i^l) = fa(v_j^l)$, for all v_k^l at layer l with $fa(v_i^l) \neq fa(v_k^l)$ and $fa(v_j^l) \neq fa(v_k^l)$, we have

$$\delta(v_i^l, v_j^l) < \delta(v_i^l, v_k^l) \tag{5}$$

$$\delta(v_i^l, v_j^l) < \delta(v_j^l, v_k^l) \tag{6}$$

Vertical constraint: From the biological point of view, a clustering tree can be likened to a gene tree, in which the lower vertexes propagate from the upper vertexes. The more children a vertex has, the more genes it inherits from its father, that is, the more similar it is to its father. We use a personalized distance rank method to describe the vertical constraint: for all vertex m,

$$\delta(v_1, v_m) < \delta(v_2, v_m) < \cdots < \delta(v_n, v_m) \tag{7}$$

iff $n(v_1) > n(v_2) \cdots > n(v_n)$, where $v_1, v_2, \cdots, v_n$ are children of v_m, i.e., $v_1, v_2, \cdots, v_n \in Ch(v_m)$, and $n(v_1), n(v_2), \cdots, n(v_n)$ respectively denote the number of the children of vertex $v_1, v_2 \cdots v_n$.

Affiliation constraint: In hierarchical networks, the bottom node clustering forms small communities, while the small community clustering forms large communities. Therefore, the parent node can be regarded as the feature set of all child nodes. Our HGE model describes the formation process of hierarchical networks from the perspective of generation: as the attribute set of the entire hierarchical network, the root node generates the child node of the next layer according to the attribute classification and repeat this top-down generation process to obtain the entire hierarchical network. In order to capture this hierarchical affiliation (generation) relationship, in the HGE model, nodes are embedded as Gaussian distributions, the mean value represents the position in the embedding space and is generated by the distribution of the parent node, and the variance represents the tensility of the node, which can also represent the differences among all child nodes. Especially, the leaf node has no tensility, that is, the variance of the embedding is 0.

A community is a small group of users who have the same hobbies, occupations, social relations and so on. Users in the same community often have similar attributes. Traditional community-based embedding methods always focus on how to capture the similarity between nodes in the same community[35] which ignore the differences between nodes. The affiliation constraint can be defined as follows: for vertex v_i^l and his father vertex v_j^{l-1},$\Phi_i^l = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$,$\Phi_j^{l-1} = \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where the μ_i is sampled from the hyper-distribution$\Phi_j^{'-1}$, i.e., $\boldsymbol{\mu}_i \sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j$. Especially, when $i = T$ (vertex i is at the bottom layer),$\Phi_i^l = \mathcal{N}(\boldsymbol{\mu}_i, \mathbf{0})$.

4.1.2. General Network Structure Properties

In order to preserve the structure of general network, we keep the embedding of nodes satisfy the first-order proximity and the second-order proximity. The first-order proximity is that, for each pair of nodes v_1 and v_2 linked by an edge(v_1, v_2), the similarity between v_1 and v_2 should be greater than that of two nodes without edge connection, and the second-order proximity is to keep each node pair(v_1, v_2) that share the same neighborhood to be similar.

4.1.3. Node Attribute Properties

Given a node i with an attribute a, the logistic model is used with mean value of multidimensional gaussian distribution of node i, i.e., $\boldsymbol{\mu}_i$ as input features to predict the probability of node i associating attribute a:

$$R_{i,a} = \sigma(W_a^T \boldsymbol{\mu}_i + b), \tag{8}$$

where W_a is the logistic weight factor, b is the bias and $\sigma(.)$ is a logistic function defined as $\sigma(x) = (1 + e^{-x})^{-1}$.

4.2. Learning Procedure

In this subsection, we will introduce the learning procedure of HGE model. Like most unsupervised embedding models, we formulate network structure preservation as an optimization problem. By optimizing the lower bound of corresponding loss function, the HGE model converges. Our loss function can be divided into three independent parts, corresponding to capture the structure of hierarchical network, general network and node attributes.

4.2.1. Hierarchical Network Optimization

First, the similarity of the same community node is captured by optimizing the lower bound of the loss function. Nodes in the same community can be closer in the hidden space, while nodes in different communities will be pushed farther. It can be defined as follows:

$$\mathcal{L}_{hor} = \sum_{l=3}^{T} \sum_{\substack{i,j,k \in \mathcal{S}_l, \\ fa(v_i^l)=fa(v_j^l)\neq fa(v_k^l)}} \delta(v_i^l, v_j^l)^2 + exp^{-\delta(v_i^l, v_k^l)}, \tag{9}$$

where, $\mathcal{S}_l$ is the vertex set of layer l, T is the depth of the clustering tree.

In order to capture the hierarchical vertical relationship, we use the following loss function to achieve the personalized distance ranking in Eq. (7), as shown below:

$$\mathcal{L}_{ver} = \sum_{l=1}^{T-1} \sum_{\substack{i,j \in \mathcal{S}_l, \\ fa(v_i^l)=fa(v_j^l)}} \delta(v_i^l, fa(v_i^l))^2 - exp^{\delta(v_j^l, fa(v_i^l))}, \tag{10}$$

when $n(v_i^l) < n(v_j^l)$

To capture hierarchical affiliation, the HGE model assumes that the mean of node embedding is subordinated to the Gaussian distribution corresponding to its father vertex. We use maximum likelihood to approximate the embedding of the father node. For arbitrary vertex i, suppose that, vertex v_1, v_2, ..., v_m are children of vertex i with $\Phi_{v_1} = \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \Phi_{v_i} = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, ..., $\Phi_{v_m} = \mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$, the maximum likelihood function can be defined as follows:

$$L(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{m} f(\boldsymbol{\mu}_i;\ \boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{11}$$

$$= \prod_{i=1}^{m} (2\pi)^{-\frac{n}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu})) \tag{12}$$

$$= (2\pi)^{-\frac{mn}{2}} |\boldsymbol{\Sigma}|^{-\frac{m}{2}} \exp(-\frac{1}{2} \sum_{i=1}^{m} (\boldsymbol{\mu}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_i - \boldsymbol{\mu})) \tag{13}$$

where n is the dimension of $\boldsymbol{\mu}$.

The logarithmic likelihood function is that:

$$\ln L(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \ln(2\pi)^{-\frac{mn}{2}} + \ln|\boldsymbol{\Sigma}|^{-\frac{m}{2}} \tag{14}$$

$$+ \ln \exp(-\frac{1}{2}\sum_{i=1}^{m}(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu})) \tag{15}$$

$$= C - \frac{m}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{i=1}^{m}(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu})), \tag{16}$$

where C is a constant.

We use the negative logarithmic maximum likelihood as loss function to capture hierarchical affiliation:

$$\mathcal{L}_{aff} = -\ln L(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{17}$$

To sum up, the objective function to capture hierarchical network structure can be defined as follows:

$$\mathcal{L}_{hie} = \mathcal{L}_{hor} + \mathcal{L}_{ver} + \lambda_1 \mathcal{L}_{aff}, \tag{18}$$

where λ_1 is a trade off parameter to balance the three parts of the loss function $\mathcal{L}_{hierarchical}$. We can optimize the parameters such that the loss $\mathcal{L}_{hierarchical}$ is minimized, thus the three constrains we proposed can be satisfied.

4.2.2. General Network Optimization

Following the LINE model [25], we construct our unsupervised training loss based on the first-order and second-order proximities, which is an efficient unsupervised learning objective for graph data and can capture both the direct and indirect similarities between nodes in general network.

$$\mathcal{L}_1 = -\sum_{(v_i, v_j)\in\mathcal{E}} w_{i,j} \log p_1(v_i, v_j) \tag{19}$$

$$\mathcal{L}_2 = -\sum_{(v_i, v_j)\in\mathcal{E}} w_{i,j} \log p_2(v_j|v_i) \tag{20}$$

$$\mathcal{L}_{ns} = \mathcal{L}_1 + \mathcal{L}_2 \tag{21}$$

where the $\mathcal{L}_1$ and $\mathcal{L}_2$ is the first-order and second-order objectives, Ons is the node similarity objectives of our model. The probability p_1 and p_2 are computed as:

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-(\boldsymbol{\mu}_i)^T \cdot \boldsymbol{\mu}_j)}, \tag{22}$$

$$p_2(v_j|v_i) = \frac{\exp((\boldsymbol{\mu}_j)^T \cdot \boldsymbol{\mu}_i)}{\sum_{k=1}^{V} \exp((\boldsymbol{\mu}_k)^T \cdot \boldsymbol{\mu}_i)}, \tag{23}$$

The $\mathcal{L}_2$ can further be optimized by the negative sampling[47].

4.2.3. Node Attribute Optimization

Associating with Eq. (8), the loss function of this part is defined as follows:

$$\mathcal{L}_A = D_{KL}(\hat{R}||R) = -\sum_{i\in\mathcal{V}}\sum_{A_{i,a}\in\boldsymbol{\mathcal{A}}} A_{i,a} \log R_{i,a}, \tag{24}$$

Algorithm 1: HGE-GA algorithm

Input : social network $\mathcal{G} = (\boldsymbol{\mathcal{V}}, \boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{A}})$, the embeddings of all nodes Φ, seed set size K,
Output : Seed node set $\mathcal{S}$

predicts influence probability of arbitrary pairwise nodes, $p_{u,v}$;
foreach $j = 1 \dots K$ **do**
  $u^* \leftarrow \arg\max_{u^* \in \mathcal{U}} \sigma(\mathcal{S} \cup u^*) - \sigma(\mathcal{S})$;
  $\mathcal{S} = \mathcal{S} \cup u^*$;
end for
return $\mathcal{S}$;

where $A_{i,a} \in \boldsymbol{\mathcal{A}}$ is a binary-valued attribute weight indicating that attribute a associates with node i iff $A_{i,a} = 1$, $\hat{R}$ is the empirical distribution of attribute weight with empirical distribution of node i associating attribute a, i.e., $\hat{R}_{i,a}$, being simply set as $A_{i,a}$. Note that all the attributes in this paper are binary-valued. Although $\hat{R}_{i,a}$ is applicable for not only binary-valued attributes but also real-valued attributes, we focus primarily on binary-valued attributes in this paper.

To sum up, the final objective $\mathcal{L}_{final}$ of our model can be written by the sum of the $\mathcal{L}_{hie}$, $\mathcal{L}_{ns}$ and $\mathcal{L}_{A}$:

$$\mathcal{L}_{final} = \mathcal{L}_{hie} + \mathcal{L}_{ns} + \mathcal{L}_{A}, \tag{25}$$

in which, the $\mathcal{L}_{final}$ can be optimized by the Adam method [48].

To ensure that embedding can preserve both vertical and horizontal features after training, the whole embedding process is executed from the bottom up. For each node v_i^l at layer l, the learned representation can be obtained after optimizing the horizontal loss of layer l and the vertical loss at layer $i+1$ with Adam.

4.3. HGE-GA Algorithm

It has been proposed by previous work on social networks influence [49] that the more equivalent the network structure of two nodes is, the more likely they make similar judgments, even without edge between them, because these two nodes connect to other nodes more identically. It has also revealed by previous studies on social network that opinions and behaviors of nodes are affected by node attribute [18], i.e., similar attributes of nodes cause similar behaviors. Therefore, the similarity of network structure and node attributes are the two factors underlying effect the influence probability between two nodes. We measure the similarity of node pair and predict influence probability of node pair with the embedding of nodes inferred from the proposed HGE model. With the predicted influence probability of node pair, we utilize general greedy strategy to compute the top-K influential nodes in the network. The proposed HGE-GA algorithm is outlined in Algorithm 2, and will be detailed in this subsection.

Traditional network representation methods embed nodes into a low-dimensional vector space, and they always use inner-product to measure the similarities of node pairs and then the relationships between node pairs can be captured. However, this way can not capture the relationships among neighborhoods of nodes in some cases[50]. As shown in Fig.5, for a node v_1 and its two neighborhoods u_1 and u_2, two relationships are represented as follows:

$$\begin{aligned} v_1 \cdot u_1 &= v_1 \cdot u_2, \\ u_1 \cdot u_2 &= 0, \end{aligned} \tag{26}$$

As neighbors of node v_1, nodes u_1 and u_2 are similar in attributes, but the method based on inner-product can not capture the relationship between u_1 and u_2 in the latent space. Similarly, the relationship between node v_1 and v_2 can not be captured by using inner-product.

To avoid the case shown in Eq.(26), we use the two-norm form which satisfies the critical triangle inequality. The norm of a vector can simply be understood as its length, or the corresponding distance between two points. Two-normal form is a common way to measure distance between

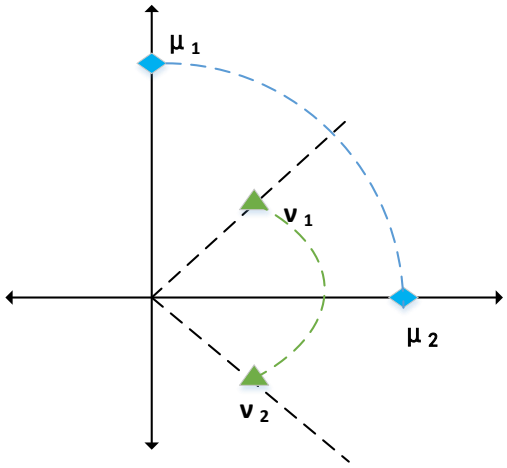

Figure 5. A case that methods based inner-product can not capture the relationship between nodes. As we can see, the relationship between u_1 and u_2 and the relationship between node v_1 and v_2 can not be captured.

vectors[51–53]. Using Two-normal form, we define the relationship between node i and node j as follows:

$$g(\Phi_i,\Phi_j) = \left\|\Phi_i - \Phi_j\right\|_2 \tag{27}$$

The above relationship with two-normal form well captures the similarity of two nodes. And we regard the similarity of two nodes as influence probability between two nodes, since the similarity of network structure and node attributes are the two factors underlying effect the influence probability between two nodes. Thus, influence probability between node i and j, i.e., $p_{i,j}$, can be predicted as follows.

$$p_{i,j} = g(\Phi_i,\Phi_j) = \left\|\Phi_i - \Phi_j\right\|_2 \tag{28}$$

After obtaining influence probability of arbitrary pairwise nodes, under general Independent Cascade (IC) diffusion model, we utilize general greedy strategy to compute the top-K influential nodes, as Algorithm 2 shows.

5. Experiments Setup

We quantitatively evaluate the performance of the proposed HGE model in downstream learning tasks (vertex classification, link prediction, network visualization) and compare the proposed algorithm HGE-GA with the state-of-art influence maximization algorithms using the metric of expected spread, on several large-scale real-world datasets. In this section, we will detail the experimental setup, including research questions §5.1, datasets §5.2, baselins §5.3, evaluation metrics and experimental settings §5.4.

5.1. Research Questions

We aim at answering the following research questions to evaluate the performance of the proposed HGE model and HGE-GA algorithm.

(RQ1) How does the proposed HGE model perform in terms of traditional graph mining tasks, e.g., vertex classification, link prediction and network visualization, comparing with baselines?

(RQ2) can the proposed influence maximization algorithm, HGE-GA, outperform the state-of-art influence maximization algorithms w.r.t. expected spread?

(RQ3) Does the Two-normal form to measure distance contribute to improve the performance of the proposed algorithm?

5.2. Datasets

We use four datasets, the detailed properties of which are shown in Table 2.

• **Cora**[54]:The Cora datasets is citation networks, which is composed of a large number of academic articles. Nodes are publications and edges are citation links.

• **DBLP**[55]: DBLP dataset [1], which consists of bibliography data in computer science. Each paper may cite or be cited by other papers, which naturally forms a citation network.

• **BlogCatalog**[26]: This dataset is a social relationship network, which is crawled from the BlogCatalog website [2]. BlogCatalog is composed of the bloggers and their social relationships. The labels of nodes indicate the interests of the bloggers.

• **Flickr** [56]: Flickr is a social network where users can share pictures and videos [3]. In this dataset, each node is a user and each side is a friend relationship between users. In addition, each node has a label that identifies the user interest group.

• **Pubmed**[54]: Pubmed is a public search database that provides biomedical paper and abstract search service[4]. In this dataset, nodes are publications and edges are citation links.

Table 2: **Statistics of the data sets used in the experiments.**

	#Nodes	#Edges	#Attributes	#Labels
Cora	2,708	5,429	1,433	7
DBLP	17,716	105,734	1,639	4
BlogCatalog	5,196	171,743	8,189	6
Flickr	7,575	239,738	12,047	9
Pubmed	19,717	44,338	500	3

5.3. Baselines

We evaluate the performance of the proposed HGE model comparing with the following baselines:

- **GraphSAGE** [57]: an attributed network embedding model which leverages node attributes and generates embeddings by sampling and aggregating features from a node's local neighborhood.
- **AANE** [58]: a model which learns attributed network embedding efficiently by decomposing the complex modeling and optimization into sub-problems.
- **M-NMF** [35]: a single-layer community structure preserving baseline, which integrates the community information through a matrix factorization.
- **GNE** [41]: a multi-layer community structure preserving baseline, which embeds communities onto surface of the spheres.
- **SpaceNE** [44]: a method which applies subspace to hierarchical network embedding and model and preserves proximity between pairwise nodes and between communities.

Moreover, we evaluate the performance of the proposed HGE-GA algorithm comparing with the following baselines:

- **PMIA** [9]: This is a heuristic algorithm that defines the Maximum Influence In-Arborescence (MIIA) and leverage sequence submodularity to compute influence spread.
- **IMM** [7]: It utilizes a classical statistical tool, martingale as well as RR Sets (Reverse Reachable Sets), and can provides higher efficiency in practice.
- **TSH-GA** : Zhou et al. [18] proposes a method to compute influence probability of node pair, which considers social tie, general network structure, node attributes as factors underlying

[1] Available from: *http```dblp`uni`trier`de`xml`* .
[2] Available from: *http```www`blogcatalog`com`*
[3] Available from: *http```www`Flickr`com`*
[4] Available from: *http```pubmed`com`cutestat`com`*

Table 3: Node classification performance.

method	DBLP	BlogCatalog	Flickr
	F1	F1	F1
AANE	.702	.515	.517
GraphSAGE	.731	.625	.649
M-NMF	.524	.604	.587
GNE	.741	.623	.631
SpaceNE	.767	.654	**.651**
HGE	**.807**	**.669**	.635

influence but executes feature extraction of these three factors by typically hand-crafted rules. TSH-GA is an influence maximization algorithm jointing the method to compute influence probability of node pair proposed in [18] and the general greedy algorithm.

- **HGE-1N-GA**: It is the variant of our HGE-GA, using one-normal form to compute the similarity of two nodes.

5.4. Evaluation Metrics and Experimental Settings

Evaluation metrics for the proposed HGE model. We evaluate the performance of link prediction of HGE in terms of Area Under Curve (AUC) and average precision (AP) [26], and evaluate the performance of vertex classification in terms of F1-Measure (F1), which is defined as $F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$.

Evaluation metric for the proposed HGE-GA algorithm. We evaluate the overall performance of our HGE-GA algorithm by adopting expected spread as evaluation metrics, which is an extensively used metric on influence maximization problem. And expected spread is defined as the expected number of active nodes in the network, including the newly activated nodes and the set of seed nodes that were initially active (seed node set), after the influence spread is over (there is no any more node being activated), given the seed set size K, which is an integer.

Experimental settings. All embedding sizes are 64, and the number of training epochs is 1,000. The initial learning rate of Adam is set as 0.001. All results are obtained by averaging 10 experiments. Besides, we implement all the baselines by the codes released by the authors. The parameters of the baselines are tuned to be optimal or set according to the corresponding literatures. All algorithms run under the IC diffusion model. Furthermore, for the two algorithms, PMIA and IMM, which do not have the step of predicting influence probability of node pair, we set the propagation probability from node u to node v, i.e., $p(u,v)$, as $1/i$, where i denotes the number of incoming edge of node v. This method of setting influence probability is extensively adopted in the previous literatures [9,59–61].

6. Results and Analysis

6.1. Performance of Vertex Classification of HGE Model

Vertex classification is one of the important tasks to detect the performance of embedding models. In this section, we perform vertex classification task to evaluate the performance of the learned embeddings and compare with the baseline methods. We choose three datasets (DBLP, BlogCatalog and Flickr) which have ground-truth classes. To be specific, We first sample a small number of nodes as training data and the rest is test data. Same to [21], we use one-vs-rest logistic regression for node classification, and the training data size is 10%, the results are reported in Tab 3. As it can be seen in Tab.3, the proposed HGE model performs the best in DBLP and BlogCatalog datasets, and shows competitive performance Flickr dataset compared with other network embedding baselines. It proves that the proposed model can better capture the network structures.

To obtain the effect of the size of the training data on the classification result, we make a series of choices on the percentage of the training data(2%, 4%, 6%, 8%,10%). This process is repeated for 10 times, and we use the average score of F1-score as evaluation metrics. Fig.6 shows the effect

of the proportion of data used for training logistic regression on experimental performance. The proposed HGE model can obtain the optimal performance under various conditions which proves the proposed model is robust across different percentage of test datasets.

6.2. Performance of Link Prediction of HGE Model

Link prediction aims to predict the future interactions of nodes in the network. In this subsection, we compare the proposed HGE model with the baselines on link prediction task. Same to [34], we create a validation/test set that contains 5%/10% randomly selected edges respectively and equal number of randomly selected non-edges. To measure the performance of link prediction task, we report the area under the ROC curve (AUC) and the average precision (AP) scores for each method.

Tab. 4 shows the link prediction performance of the proposed HGE model and the baselines on the five datasets mentioned in §5.2. We can see that, our HGE model significantly outperforms the baselines across all datasets, which demonstrates that modelling hierarchical network structure, general network structure as well as node attributes in latent space is effective to learn better node embeddings.

6.3. Performance of Network Visualization of HGE Model

Network visualization is also one of the important means to detect the quality of the learned embedding, and it maps a network into the two-dimensional space. In this section, for better visualization, we visualize a sub-network which are selected from the BlogCatalog dataset with 175 nodes and 150 edges.

From Figure 7, we can find that our HGE model preserves the default hierarchical structure, and distributes the nodes of each layer more uniformly in a fan-shaped area.

6.4. Overall Performance of HGE-GA algorithm

We adopt the metric of **expected spread** to evaluate the overall performance of all influence maximization algorithms. To compare the proposed HGE-GA algorithm with the baselines, we change the size of seed node set K as 1, 20, 30, 40, 50. The experimental results are reported in Fig. 7 (a)-(e). It can be observed from Fig. 7 (a)-(e) that:

(1) The expected spread increases for all algorithms when enlarging the size of seed node set. And the proposed HGE-GA algorithm outperforms all the baselines significantly for all seed size, on the five datasets. For instance, on Pubmed data set, when the seed set size $K = 50$, the value of expected spread of the proposed HGE-GA algorithm is 1,445, comparing with 721 for PMIA, 819 for IMM, 1,052 for TSH-GA, 1,384 for HGE-1N-GA.

(2) On all the five datasets, all of the attribute-related algorithms, TSH-GA, HGE-1N-GA and HGE-GA, outperform non-attribute-related algorithms, PMIA and IMM, in terms of the performance of expected spread. This is because Algorithm PMIA and Algorithm IMM do not have the step of predicting influence probability of node pair, with influence probability being set uniformly. Moreover, this demonstrates that homophily is a cause of similar behaviors and is useful for predicting future behaviors of users. Consequently, the expected spread can be increased obviously when node attributes are taken into account in modeling.

(3) HGE-1N-GA and HGE-GA outperform TSH-GA on the five datasets, for the metric of expected spread. The reason is that HGE-1N-GA and HGE-GA consider general network structure, hierarchical network structure as well as node attributes to learn users' latent feature representation for predicting influence probability of node pair, but TSH-GA use hand-crafted rules to extract the features of users' interactions, network structure and node homophily, heavily depending the domain expert knowledge and without considering hierarchical network structure.

(4) The expected spread of HGE-GA is superior than that of HGE-1N-GA. The reason is obvious: HGE-1N-GA is with one-norm form to compute similarity of two nodes, while HGE-GA leverage two-norm form to compute similarity, which can capture the relationships between nodes better.

Table 4: Link prediction performance with embedding size L = 64.

method	Cora		DBLP		BlogCatalog		Flickr		Pubmed	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
AANE	.834	.812	.761	.789	.771	.776	.678	.639	.842	.837
GraphSAGE	.869	.892	.803	.811	.807	.819	.802	.817	.812	.825
M-NMF	.867	.861	.841	.837	.716	.710	.728	.736	.816	.825
GNE	.944	.941	.935	.944	.805	.802	.828	.837	.945	.942
SpaceNE	.927	.914	.927	.933	.815	.827	.908	.917	.936	.958
HGE	**.979**	**.974**	**.986**	**.989**	**.836**	**.845**	**.929**	**.913**	**.977**	**.972**

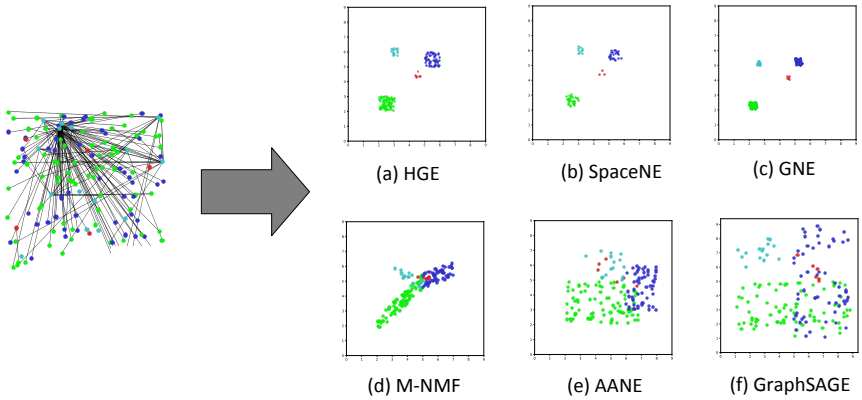

Figure 6. The experimental results of network visualization in 2-D space.

7. Conclusions

In this paper, we study the problem of influence maximization in attributed social networks, assuming the influence strength is unobservable directly. From the deep learning perspective, we formulate the problem and propose a deep hierarchical network embedding model HGE to map nodes (with attributes) into latent space automatically, incorporating hierarchical community structure, node attributes and general network structure into a unified deep generative framework. Then, with the leaned latent representation of each node, we propose a HGE-GA algorithm to predict influence strength and find the seed node set through a greedy-based maximization algorithm. The experimental results show that the proposed HGE model is able to learn representations of nodes in attributed networks far more effectively than state-of-the-art model in terms of several downstream applications such as vertex classification, link prediction and network visualization. It is also verified by the experimental results that the proposed HGE-GA algorithm significantly outperforms the state-of-the-art algorithms for influence maximization in attributed social networks.

As to future work, we intend to extend our HGE model to temporal attributed networks, which is more challenging because this kind of attributed networks evolve over time and vertexes need to be embedded dynamically.

Author Contributions: Conceptualization, Huimin Huang; Data curation, Luodi Xie and Qing Du; Formal analysis, Luodi Xie; Methodology, Luodi Xie and Huimin Huang; Project administration, Hong Shen; Software, Qing Du; Supervision, Huimin Huang and Hong Shen; Validation, Huimin Huang; Visualization, Luodi Xie and Qing Du; Writing C original draft, Luodi Xie; Writing C review editing, Huimin Huang and Hong Shen.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, of in the decision to publish the results.

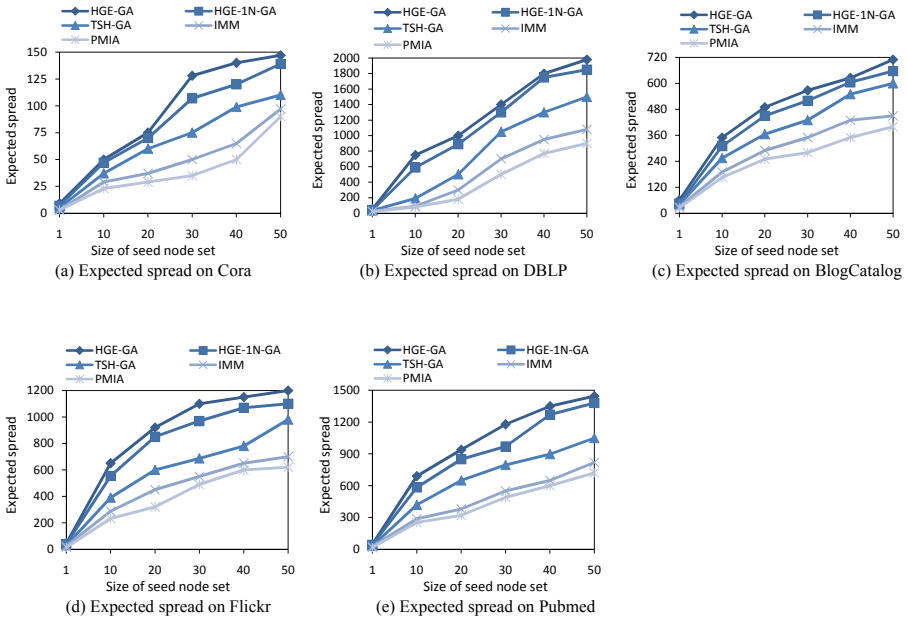

Figure 7. Expected spread with different size of seed node set K.

References

1. Kempe, D.; Kleinberg, J.; Tardos, É. Maximizing the spread of influence through a social network. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, pp. 137–146.
2. Leskovec, J.; Krause, A.; Guestrin, C.; Faloutsos, C.; VanBriesen, J.; Glance, N. Cost-effective outbreak detection in networks. Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, 2007, pp. 420–429.
3. Chen, W.; Wang, Y.; Yang, S. Efficient influence maximization in social networks. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009, pp. 199–208.
4. Cheng, S.; Shen, H.; Huang, J.; Zhang, G.; Cheng, X. Staticgreedy: solving the scalability-accuracy dilemma in influence maximization. Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013, pp. 509–518.
5. Goyal, A.; Lu, W.; Lakshmanan, L.V. Celf++ optimizing the greedy algorithm for influence maximization in social networks. Proceedings of the 20th international conference companion on World wide web, 2011, pp. 47–48.
6. Tang, Y.; Xiao, X.; Shi, Y. Influence maximization: Near-optimal time complexity meets practical efficiency. Proceedings of the 2014 ACM SIGMOD international conference on Management of data, 2014, pp. 75–86.
7. Tang, Y.; Shi, Y.; Xiao, X. Influence maximization in near-linear time: A martingale approach. Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, 2015, pp. 1539–1554.
8. Galhotra, S.; Arora, A.; Roy, S. Holistic influence maximization: Combining scalability and efficiency with opinion-aware models. Proceedings of the 2016 International Conference on Management of Data, 2016, pp. 743–758.
9. Chen, W.; Wang, C.; Wang, Y. Scalable influence maximization for prevalent viral marketing in large-scale social networks. Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, 2010, pp. 1029–1038.
10. Jiang, Q.; Song, G.; Gao, C.; Wang, Y.; Si, W.; Xie, K. Simulated annealing based influence maximization in social networks. Proceedings of the AAAI Conference on Artificial Intelligence, 2011, Vol. 25.
11. Tang, J.; Wu, S.; Sun, J. Confluence: Conformity influence in large social networks. Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013, pp. 347–355.
12. Tang, J.; Sun, J.; Wang, C.; Yang, Z. Social influence analysis in large-scale networks. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009, pp. 807–816.
13. Hu, Z.; Yao, J.; Cui, B.; Xing, E. Community level diffusion extraction. Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, 2015, pp. 1555–1569.
14. Shaghaghian, S.; Coates, M. Bayesian inference of diffusion networks with unknown infection times. 2016 IEEE Statistical Signal Processing Workshop (SSP). IEEE, 2016, pp. 1–5.
15. Barbieri, N.; Bonchi, F.; Manco, G. Topic-aware social influence propagation models. *Knowledge and information systems* **2013**, *37*, 555–584.
16. Li, C.; Ma, J.; Guo, X.; Mei, Q. Deepcas: An end-to-end predictor of information cascades. Proceedings of the 26th international conference on World Wide Web, 2017, pp. 577–586.
17. Qiu, J.; Tang, J.; Ma, H.; Dong, Y.; Wang, K.; Tang, J. Deepinf: Social influence prediction with deep learning. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 2110–2119.
18. Zhou, F.; Jiao, R.J.; Lei, B.Y. A linear threshold-hurdle model for product adoption prediction incorporating social network effects. *Inf. Sci.* **2015**, *307*, 95–109.

19. Li, J.; Cheng, K.; Wu, L.; Liu, H. Streaming Link Prediction on Dynamic Attributed Networks **2018**. pp. 369–377.
20. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016, pp. 855–864.
21. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: On learning of social representations. Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2014, pp. 701–710.
22. Roweis, S.T.; Saul, L.K. Nonlinear dimensionality reduction by locally linear embedding. *science* **2000**, *290*, 2323–2326.
23. Tian, F.; Gao, B.; Cui, Q.; Chen, E.; Liu, T.Y. Learning deep representations for graph clustering. Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014.
24. Yang, L.; Cao, X.; He, D.; Wang, C.; Wang, X.; Zhang, W. Modularity Based Community Detection with Deep Learning. IJCAI, 2016, Vol. 16, pp. 2252–2258.
25. Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; Mei, Q. Line: Large-scale information network embedding. Proceedings of the 24th international conference on world wide web. International World Wide Web Conferences Steering Committee, 2015, pp. 1067–1077.
26. Meng, Z.; Liang, S.; Bao, H.; Zhang, X. Co-embedding Attributed Networks. Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining. ACM, 2019, pp. 393–401.
27. Meng, Z.; Liang, S.; Fang, J.; Xiao, T. Semi-supervisedly Co-embedding Attributed Networks. Advances in Neural Information Processing Systems, 2019, pp. 6504–6513.
28. Cao, S.; Lu, W.; Xu, Q. Grarep: Learning graph representations with global structural information. Proceedings of the 24th ACM international on conference on information and knowledge management. ACM, 2015, pp. 891–900.
29. Yang, C.; Sun, M.; Liu, Z.; Tu, C. Fast Network Embedding Enhancement via High Order Proximity Approximation. IJCAI, 2017, pp. 3894–3900.
30. Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; Chang, E. Network representation learning with rich text information. Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.
31. Zhang, D.; Yin, J.; Zhu, X.; Zhang, C. User profile preserving social network embedding. IJCAI International Joint Conference on Artificial Intelligence, 2017.
32. Kingma, D.P.; Mohamed, S.; Rezende, D.J.; Welling, M. Semi-supervised learning with deep generative models. Advances in neural information processing systems, 2014, pp. 3581–3589.
33. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* **2013**.
34. Wang, D.; Cui, P.; Zhu, W. Structural deep network embedding. Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016, pp. 1225–1234.
35. Wang, X.; Cui, P.; Wang, J.; Pei, J.; Zhu, W.; Yang, S. Community preserving network embedding. Thirty-First AAAI Conference on Artificial Intelligence, 2017.
36. Jia, Y.; Zhang, Q.; Zhang, W.; Wang, X. Communitygan: Community detection with generative adversarial nets. The World Wide Web Conference. ACM, 2019, pp. 784–794.
37. Li, Y.; Wang, Y.; Zhang, T.; Zhang, J.; Chang, Y. Learning network embedding with community structural information. Proceedings of the 28th International Joint Conference on Artificial Intelligence. AAAI Press, 2019, pp. 2937–2943.
38. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. Advances in neural information processing systems, 2014, pp. 2672–2680.
39. Yang, J.; Leskovec, J. Community-affiliation graph model for overlapping network community detection. 2012 IEEE 12th international conference on data mining. IEEE, 2012, pp. 1170–1175.
40. Clauset, A.; Moore, C.; Newman, M.E. Structural inference of hierarchies in networks. ICML Workshop on Statistical Network Analysis. Springer, 2006, pp. 1–13.
41. Du, L.; Lu, Z.; Wang, Y.; Song, G.; Wang, Y.; Chen, W. Galaxy Network Embedding: A Hierarchical Community Structure Preserving Approach. IJCAI, 2018, pp. 2079–2085.
42. Sales-Pardo, M.; Guimera, R.; Moreira, A.A.; Amaral, L.A.N. Extracting the hierarchical organization of complex systems. *Proceedings of the National Academy of Sciences* **2007**, *104*, 15224–15229.
43. Ma, J.; Cui, P.; Wang, X.; Zhu, W. Hierarchical taxonomy aware network embedding. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1920–1929.
44. Long, Q.; Wang, Y.; Du, L.; Song, G.; Jin, Y.; Lin, W. Hierarchical Community Structure Preserving Network Embedding: A Subspace Approach **2019**. pp. 409–418.
45. Ma, Y.; Ren, Z.; Jiang, Z.; Tang, J.; Yin, D. Multi-Dimensional Network Embedding with Hierarchical Structure **2018**. pp. 387–395.
46. Nickel, M.; Kiela, D. Poincaré embeddings for learning hierarchical representations. Advances in neural information processing systems, 2017, pp. 6338–6347.
47. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 2013, pp. 3111–3119.
48. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**.
49. Wasserman, S.; Faust, K.; others. Social network analysis: Methods and applications **1994**.
50. Hsieh, C.K.; Yang, L.; Cui, Y.; Lin, T.Y.; Belongie, S.; Estrin, D. Collaborative metric learning. Proceedings of the 26th international conference on world wide web, 2017, pp. 193–201.

51. Dokmanic, I.; Parhizkar, R.; Ranieri, J.; Vetterli, M. Euclidean Distance Matrices: Essential theory, algorithms, and applications. *IEEE Signal Processing Magazine* **2015**, *32*, 12–30.
52. Oh Song, H.; Xiang, Y.; Jegelka, S.; Savarese, S. Deep metric learning via lifted structured feature embedding. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 4004–4012.
53. Zhang, Y.G.; Zhang, C.; Zhang, D. Distance metric learning by knowledge embedding. *Pattern Recognition* **2004**, *37*, 161–163.
54. Bojchevski, A.; Günnemann, S. Deep Gaussian Embedding of Graphs: Unsupervised Inductive Learning via Ranking. International Conference on Learning Representations, 2018.
55. Pan, S.; Wu, J.; Zhu, X.; Zhang, C.; Wang, Y. Tri-party deep network representation. *Network* **2016**, *11*, 12.
56. Huang, X.; Li, J.; Hu, X. Label informed attributed network embedding. Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, 2017, pp. 731–739.
57. Hamilton, W.L.; Ying, Z.; Leskovec, J. Inductive Representation Learning on Large Graphs. Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 1024–1034.
58. Huang, X.; Li, J.; Hu, X. Accelerated Attributed Network Embedding. Proceedings of the 2017 SIAM International Conference on Data Mining, Houston, Texas, USA, April 27-29, 2017. SIAM, 2017, pp. 633–641.
59. Goyal, A.; Bonchi, F.; Lakshmanan, L.V. A Data-Based Approach to Social Influence Maximization. *Proceedings of the VLDB Endowment* **2011**, *5*.
60. Jung, K.; Heo, W.; Chen, W. Irie: Scalable and robust influence maximization in social networks. 2012 IEEE 12th International Conference on Data Mining. IEEE, 2012, pp. 918–923.
61. Wang, C.; Chen, W.; Wang, Y. Scalable influence maximization for independent cascade model in large-scale social networks. *Data Mining and Knowledge Discovery* **2012**, *25*, 545–576.