Deep Autoencoder-like Nonnegative Matrix Factorization for Community Detection

Fanghua Ye, Chuan Chen, Zibin Zheng
School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou, China
yefh5@mail2.sysu.edu.cn,{chenchuan,zhzibin}@mail.sysu.edu.cn

ABSTRACT

Community structure is ubiquitous in real-world complex networks. The task of community detection over these networks is of paramount importance in a variety of applications. Recently, nonnegative matrix factorization (NMF) has been widely adopted for community detection due to its great interpretability and its natural fitness for capturing the community membership of nodes. However, the existing NMF-based community detection approaches are shallow methods. They learn the community assignment by mapping the original network to the community membership space directly. Considering the complicated and diversified topology structures of real-world networks, it is highly possible that the mapping between the original network and the community membership space contains rather complex hierarchical information, which cannot be interpreted by classic shallow NMF-based approaches. Inspired by the unique feature representation learning capability of deep autoencoder, we propose a novel model, named Deep Autoencoder-like NMF (DANMF), for community detection. Similar to deep autoencoder, DANMF consists of an encoder component and a decoder component. This architecture empowers DANMF to learn the hierarchical mappings between the original network and the final community assignment with implicit low-to-high level hidden attributes of the original network learnt in the intermediate layers. Thus, DANMF should be better suited to the community detection task. Extensive experiments on benchmark datasets demonstrate that DANMF can achieve better performance than the state-of-the-art NMF-based community detection approaches.

KEYWORDS

Deep nonnegative matrix factorization; Community detection; Graph clustering; Deep learning; Network analytics

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271697

ACM Reference Format:
Fanghua Ye, Chuan Chen, Zibin Zheng. 2018. Deep Autoencoder-like Nonnegative Matrix Factorization for Community Detection. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271697

1 INTRODUCTION

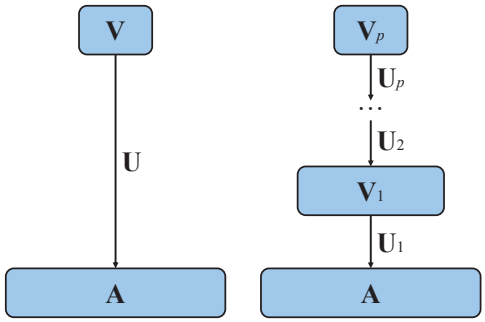
Many real-world complex interaction systems in nature and society can be characterized by complex networks, such as social networks, collaboration networks, citation networks, biological neural networks, and protein interaction networks [10, 22, 51]. These networks often consist of functional units, which manifest in the form of communities, i.e., groups of nodes with dense internal connections and sparse external connections [11]. It is well understood that analyzing the underlying community structure is of significant importance to reveal the organizational patterns and structural functions of network systems. Besides, community detection has boosted diversified practical applications, such as advertising, viral marketing, friend recommendation, and infectious disease control [9], to name but a few.

Over the past two decades, a great deal of effort has been devoted to analyzing the community structure of networks. Thus a plethora of community detection approaches have been proposed and successfully applied to specific networks [1, 25, 36, 38, 44]. Traditional community detection approaches seek to find the optimal community structure via optimizing certain criteria, e.g., modularity [24], normalized cut [31], permanence [7], and conductance [18]. These approaches usually assign each node to only one community, which contradicts the fact that a node can naturally participate in multiple communities. For example, a person can join in several discussion groups in an online forum, a researcher may be active in several areas. In recent years, nonnegative matrix factorization (NMF) has been broadly adopted for community detection [39, 41, 43, 47, 48], mostly because of the better interpretability derived from the nonnegative constraints and its natural fitness for disjoint and overlapping community detection. NMF-based community detection approaches approximately factorize the adjacency matrix $\mathbf{A}$ of a given network into two nonnegative factor matrices $\mathbf{U}$ and $\mathbf{V}$, i.e., $\mathbf{A} \approx \mathbf{UV}$ $(\mathbf{U} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0})$. As such, each column of the factor matrix $\mathbf{V}$ can be interpreted as the propensity of a node belonging to different communities (i.e., the community membership), and the factor matrix $\mathbf{U}$ can be treated as

the mapping between the original network and the community membership space. Some NMF-variants have also been utilized to deal with the community detection task, including bayesian NMF [28], nonnegative matrix tri-factorization [26, 49], and binary NMF [50].

Although various NMF-based approaches have been developed for community detection and promising performance has been delivered on some specific applications, it is still a big challenge to identify the intrinsic community structure of networks [10]. In addition, the existing NMF-based community detection approaches are all shallow methods. As aforementioned, there is only one layer mapping between the original network and the community membership space. Considering the complicated and diversified organizational patterns of real-world networks, it is highly possible that the mapping between the original network and the community membership space contains rather complex hierarchical and structural information with implicit lower-level hidden attributes, which cannot be interpreted by classic shallow NMF-based community detection approaches. Intuitively, similar nodes are more likely to be contained in the same community. In this regard, the classic shallow NMF-based methods actually learn the community-level similarity between nodes directly. Recently, deep autoencoder has been widely applied in unsupervised learning problems due to its unique feature representation learning capability [14]. Besides, deep autoencoder is an excellent scheme to narrow the gap between the lower-level abstraction and the higher-level abstraction of the original data [2]. Inspired by deep autoencoder, we can argue that by further factoring the mapping $\mathbf{U}$, in a way that each factor adds an extra layer of abstraction of the similarity between nodes from lower level to higher level, we can then obtain a better community-level similarity between nodes (i.e., a more accurate community membership matrix $\mathbf{V}$), as demonstrated in Figure 1. For example, we can learn the similarity between nodes from the first-order proximity [34], to the degree assortativity [8], the structural identity [19, 30], and finally the community-level similarity.

Based on the discussions above, in this paper, we propose a novel model, named Deep Autoencoder-like NMF (DANMF), to deal with the community detection task. Instead of merely applying the concept of NMF to a multi-layer structure as shown in Figure 1, DANMF consists of an encoder component and a decoder component, both with deep structures. Similar to deep autoencoder, the encoder component attempts to transform the original network into the community membership space with implicit low-dimensional hidden attributes captured in the intermediate layers. Each intermediate layer interprets the similarity between nodes at different levels of granularity. The decoder component is symmetric with the encoder component. It seeks to reconstruct the original network from the community membership space with the aid of the hierarchical mappings learnt in the encoder component. Different from traditional NMF-based community detection methods that consider only the loss function of the decoder component, DANMF integrates both the encoder component and the decoder component into a unified loss function. In this way, DANMF inherits the representation learning capability of deep autoencoder [2], while it improves the model's interpretability due to the nonnegative constraints, and it is suited for both disjoint community detection and overlapping community detection. Besides, DANMF incorporates a graph regularizer to respect the intrinsic geometric structure of node pairs. The overall framework of DANMF is illustrated in Figure 2.



(a) NMF (b) Deep NMF

Figure 1: (a) The architecture of NMF. (b) The architecture of deep NMF. Deep NMF learns a hierarchy of hidden attributes that aid in uncovering the final community membership of nodes.

Our main contributions can be summarized as follows:

- We propose a deep autoencoder-like NMF model, namely DANMF, to deal with the community detection task. To the best of our knowledge, it is the first approach to introduce deep NMF for community detection.
- We develop an efficient learning algorithm to optimize the proposed DANMF model, inspired by recent advances in deep learning [14].
- We conduct extensive experiments to evaluate the effectiveness and efficiency of DANMF. The results demonstrate that DANMF is superior over the state-of-the-art shallow NMF-based community detection methods.

The rest of this paper is organized as follows. Section 2 provides an overview of the related work. Section 3 describes the proposed DANMF model in detail. The learning algorithm is presented in Section 4. Then, we report the experimental results in Section 5. Finally, we conclude this paper in Section 6.

2 RELATED WORK

In this section, we briefly review the related work regarding community detection and deep matrix factorization.

2.1 Community Detection

Real-world complex networks often exhibit distinct characteristics, one of them is the presence of densely connected subnetworks, also referred to as communities. The task of community detection is to find the community structure of a given network. The problem has been a very popular research

topic in recent years and lots of effort has been devoted to developing delicate community detection methods [10, 18, 42]. However, there is no consensus on the formalization of the community detection task and a variety of criteria are proposed to characterize the property of a community, such as modularity [24], normalized cut [31], permanence [7], and conductance [18]. For a detailed review of such criteria, please refer to [6]. Among these criteria, modularity has received the most extensive attention, which requires that the number of edges within a community should be significantly larger than the expected number of edges when all edges are randomly distributed. The typical modularity-based methods include greedy algorithm [23], Louvain [3], and spectral optimization [24]. However, most of these methods aim to find disjoint communities, which contradicts the fact that a node can naturally participate in multiple communities.

As another research topic, NMF has emerged as an imperative tool for clustering analysis due to its powerful interpretability. The key of NMF is to reconstruct the original data from low-dimensional representations. With the nonnegative constraints, NMF naturally fits into disjoint community detection and overlapping community detection. As a result, numerous NMF-based community detection approaches have been proposed [19, 28, 30, 39, 41, 43, 47, 48, 50]. For example, Psorakis et al. [28] utilize a bayesian generative model to extract communities, which puts a half-normal prior over each community and then maximizes the log-likelihood of generating the original network. Zhang et al. [47] propose a preference-based NMF model to incorporate the implicit link preference information into overlapping community detection based on a basic assumption that a node prefers to build links with nodes inside its community than those outside. Yang and Leskovec [43] develop a scalable NMF-based model, which can be applied to detect densely overlapping, hierarchically nested as well as non-overlapping communities in massive networks. Recently, several network embedding techniques have also been employed to detect communities [5, 12, 34, 40], which are able to learn higher-order similarity between nodes. These methods have been proven to be closely related to NMF or standard matrix factorization as well [29].

2.2 Deep Matrix Factorization

It is common that complex data objects consist of hierarchical attributes, each of which represents a different level of abstract understanding of the objects. This phenomenon has motivated the rapid development of deep learning, which is a powerful technique to do representation learning [2]. As the success of deep learning, there have been some explorations on deep matrix factorization [13, 32, 37, 45]. The general idea of them is to stack one-layer matrix factorization into multiple layers, in the hope that hierarchical mappings can be obtained. In [37], a multi-layer semi-NMF model with a complete deep architecture is proposed to automatically learn a hierarchy of attributes to facilitate clustering tasks. In [32], Song et al. propose a structure of multi-layer NMF for classification tasks, where non-smooth NMF is adopted to solve typical NMF in each layer. A sparse deep NMF model is then proposed and successfully applied to explore the sparse structure of data objects by using the Nesterov's accelerated gradient descent algorithm [13]. More recently, Yu et al. [45] propose a deep non-smooth NMF architecture to learn part-based and hierarchical attributes simultaneously. However, all these models only consist of a decoder component.

Our proposed deep autoencoder-like NMF model DANMF integrates the encoder component and the decoder component. Thus, DANMF is fundamentally different from the existing deep matrix factorization models. What's more, DANMF is able to better inherit the representation learning capability of deep autoencoder. It is worth mentioning that some deep learning approaches like GraphEncoder [35] have already been employed for community detection. However, these approaches do not lend themselves to overlapping community detection, and there are usually a lot of parameters to be tuned. A nonnegative symmetric encoder-decoder approach [33] has also been developed for community detection, but it is a shallow model.

3 DEEP AUTOENCODER-LIKE NMF FOR COMMUNITY DETECTION

In this section, we describe our proposed deep autoencoder-like NMF mode (i.e., DANMF) for community detection. We start by introducing the notations and some preliminaries. Then, we present the details of DANMF. The architecture of DANMF is shown in Figure 2.

3.1 Notations and Preliminaries

Throughout this paper, we denote matrices by bold uppercase letters. For a given matrix $\mathbf{X}$, its (i,j)-th entry is denoted by $[\mathbf{X}]_{ij}$. The trace and Frobenius norm of $\mathbf{X}$ are denoted by $tr(\mathbf{X})$ and $\|\mathbf{X}\|_F$, respectively. The zero matrix is denoted by $\mathbf{0}$, and the identity matrix is denoted by $\mathbf{I}$.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a given network with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges, where $\mathcal{V}$ and $\mathcal{E}$ denote the node set and the edge set respectively. Typically, network $\mathcal{G}$ is described by an adjacency matrix $\mathbf{A}$, whose each entry $[\mathbf{A}]_{ij}$ characterizes the relationship between nodes i and j. For unweighted networks, we have $[\mathbf{A}]_{ij} = 1$ if there is an edge between nodes i and j, and $[\mathbf{A}]_{ij} = 0$ otherwise. If network $\mathcal{G}$ is weighted, then $\mathbf{A}$ is real-valued. When $\mathbf{A}$ violates the nonnegative constraints, we can normalize each entry of $\mathbf{A}$ to the range of $[0, 1]$.

Assume that network $\mathcal{G}$ consists of k communities. Let $\mathcal{C}$ denote the set of communities, i.e., $\mathcal{C} = \{C_i | C_i \neq \emptyset, C_i \neq C_j, 1 \leq i, j \leq k\}$, where C_i represents the i-th community. For disjoint community detection, it is required that $C_i \cap C_j = \emptyset$ if $i \neq j$. For overlapping community detection, this constraint is neglected. Suppose that we have two nonnegative matrices $\mathbf{U} \in \mathbb{R}_+^{n \times k}$ and $\mathbf{V} \in \mathbb{R}_+^{k \times n}$, where each column of $\mathbf{U}$ denotes the description of a community, and each column of $\mathbf{V}$ represents the association relationship of a node to different communities. Then, $[\mathbf{U}]_{il}[\mathbf{V}]_{lj}$ can be interpreted as the contribution of the l-th community to the edge $[\mathbf{A}]_{ij}$. That is, the expected interaction $[\hat{\mathbf{A}}]_{ij} = \sum_{l=1}^{k} [\mathbf{U}]_{il}[\mathbf{V}]_{lj}$ between

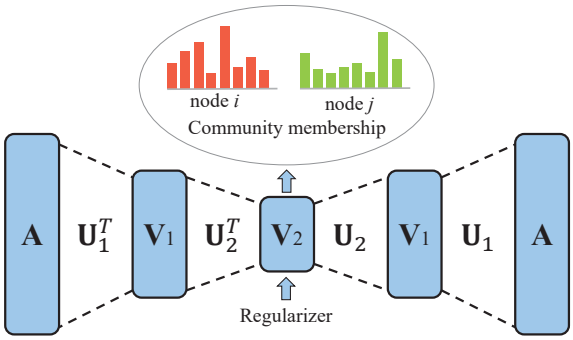


Figure 2: The architecture of DANMF. For illustration purpose, the depth is fixed at 2. The encoder component (the left part) transforms the network into the community membership space. The decoder component (the right part) reconstructs the network from the community membership space.

nodes i and j is the result of their mutual participation in the same communities [21, 28]. Obviously, $[\hat{\mathbf{A}}]_{ij}$ should be as closely consistent as possible with $[\mathbf{A}]_{ij}$, which results in the following objective function:

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{A} - \mathbf{U}\mathbf{V}\|_F^2, \text{ s.t. } \mathbf{U} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0}. \tag{1}$$

Based on the learnt $\mathbf{V}$, we can extract the community membership of nodes. For disjoint community detection, each node is assigned to the community where it gets the largest belonging propensity. For overlapping community detection, we need to set a threshold in order to determine whether a node belongs to a community or not. Such a threshold can be obtained by taking the same strategy as suggested in [47].

3.2 Deep NMF

As shown in Eq. (1), NMF learns a one-layer mapping $\mathbf{U}$ and a community-level similarity between nodes (i.e., the community membership matrix $\mathbf{V}$) directly. However, real-world networks often consist of complicated and diversified organizational patterns. Therefore, it is highly possible that the mapping between the original network and the community membership space contains rather complex hierarchical and structural information with implicit lower-level hidden attributes. It is well known that deep learning is able to narrow the gap between the lower-level abstraction and the higher-level abstraction of the original data [2]. In this sense, we propose to further factorize the mapping $\mathbf{U}$, in the hope that each factor adds an extra layer of abstraction of the similarity between nodes from low level to high level. Specifically, the adjacency matrix $\mathbf{A}$ is factorized into $p+1$ nonnegative factor matrices, as follows:

$$\mathbf{A} \approx \mathbf{U}_1\mathbf{U}_2\cdots\mathbf{U}_p\mathbf{V}_p, \tag{2}$$

where $\mathbf{V}_p \in \mathbb{R}_+^{k\times n}$, $\mathbf{U}_i \in \mathbb{R}_+^{r_{i-1}\times r_i}$ $(1 \leq i \leq p)$, and we set $n = r_0 \geq r_1 \geq \cdots \geq r_{p-1} \geq r_p = k$.

The formulation in Eq. (2) allows for a hierarchy of p layers of abstract understanding of the original network, which can be given by the following factorizations:

$$\begin{aligned} \mathbf{V}_{p-1} &\approx \mathbf{U}_p\mathbf{V}_p, \\ &\vdots \\ \mathbf{V}_2 &\approx \mathbf{U}_3\cdots\mathbf{U}_p\mathbf{V}_p, \\ \mathbf{V}_1 &\approx \mathbf{U}_2\cdots\mathbf{U}_p\mathbf{V}_p. \end{aligned} \tag{3}$$

We retain the nonnegative constraints on $\mathbf{V}_i$ $(1 \leq i < p)$ as well. By doing so, each layer of abstraction $\mathbf{V}_i$ captures the similarity between nodes at different levels of granularity, ranging from the first-order proximity, to the structural identity, and finally the community-level similarity. This deep structure will lead to more accurate community detection results, i.e., a better $\mathbf{V}_p$. In order to learn the factor matrices, we derive the following objective function:

$$\begin{aligned} &\min_{\mathbf{U}_i,\mathbf{V}_p} \mathcal{L}_{\mathrm{D}} = \|\mathbf{A} - \mathbf{U}_1\mathbf{U}_2\cdots\mathbf{U}_p\mathbf{V}_p\|_F^2, \\ &\text{s.t. } \mathbf{V}_p \geq \mathbf{0}, \mathbf{U}_i \geq \mathbf{0}, \forall i = 1, 2, \cdots, p. \end{aligned} \tag{4}$$

After optimizing Eq. (4), we can obtain the hidden attributes $\mathbf{V}_i$ $(i < p)$ by solving $\|\mathbf{A} - \mathbf{U}_1\mathbf{U}_2\cdots\mathbf{U}_i\mathbf{V}_i\|_F^2$, similar to [37].

3.3 Deep Autoencoder-like NMF

As can be seen, both Eq. (1) and Eq. (4) are based on reconstructing the original network, which corresponds to the decoder component of an autoencoder. To better inherit the representation learning capability of autoencoders, it is essential to incorporate the encoder component into the NMF-based community detection models, resulting in autoencoder-like NMF models. The rationality of an autoencoder-like NMF model is quite straightforward. For an ideal community membership matrix $\mathbf{V}$, on the one hand, it should be able to reconstruct the original network via the mapping $\mathbf{U}$ with smaller reconstruction error, and on the other hand, it should be obtained by directly projecting the original network $\mathbf{A}$ into the community membership space with the aid of the mapping $\mathbf{U}$, i.e., $\mathbf{V} = \mathbf{U}^T\mathbf{A}$. By integrating the encoder component and the decoder component into a unified loss function, the two components are capable of guiding each other during the learning process, and thus we tend to obtain the ideal community membership of nodes. To achieve this goal in the deep model, we derive the following objective function for the encoder component:

$$\begin{aligned} &\min_{\mathbf{U}_i,\mathbf{V}_p} \mathcal{L}_{\mathrm{E}} = \|\mathbf{V}_p - \mathbf{U}_p^T\cdots\mathbf{U}_2^T\mathbf{U}_1^T\mathbf{A}\|_F^2, \\ &\text{s.t. } \mathbf{V}_p \geq \mathbf{0}, \mathbf{U}_i \geq \mathbf{0}, \forall i = 1, 2, \cdots, p. \end{aligned} \tag{5}$$

By combining Eq. (4) and Eq. (5), the unified objective function of our deep autoencoder-like NMF model (i.e., DANMF) is then given as follows:

$$\begin{aligned} \min_{\mathbf{U}_i,\mathbf{V}_p} \mathcal{L} &= \mathcal{L}_{\mathrm{D}} + \mathcal{L}_{\mathrm{E}} + \lambda\mathcal{L}_{\mathrm{reg}} \\ &= \|\mathbf{A} - \mathbf{U}_1\mathbf{U}_2\cdots\mathbf{U}_p\mathbf{V}_p\|_F^2 + \|\mathbf{V}_p - \mathbf{U}_p^T\cdots\mathbf{U}_2^T\mathbf{U}_1^T\mathbf{A}\|_F^2 \\ &\quad + \lambda tr(\mathbf{V}_p\mathbf{L}\mathbf{V}_p^T), \\ \text{s.t. } &\mathbf{V}_p \geq \mathbf{0}, \mathbf{U}_i \geq \mathbf{0}, \forall i = 1, 2, \cdots, p. \end{aligned} \tag{6}$$

In Eq. (6), a graph regularizer $\mathcal{L}_{\text{reg}} = tr(\mathbf{V}_p\mathbf{L}\mathbf{V}_p^T)$ is further introduced to respect the intrinsic geometric structure of node pairs. λ denotes the regularization parameter, and $\mathbf{L}$ represents the graph Laplacian matrix. There are many ways to define $\mathbf{L}$ [26]. In this paper, we focus on undirected networks, and we set $\mathbf{L} = \mathbf{D} - \mathbf{A}$ ($\mathbf{D}$ is a diagonal matrix whose elements are row sums of $\mathbf{A}$), based on a basic assumption that linked nodes are more likely to be contained in the same communities [48].

4 OPTIMIZATION

To expedite the approximation of the factor matrices in the proposed model, we pre-train each of the layers to have an initial approximation of the factor matrices $\mathbf{U}_i$ and $\mathbf{V}_i$. This pre-training process can greatly reduce the training time of our model. The effectiveness of pre-training has been proven before on deep autoencoder networks [15]. To perform the pre-training, we first decompose the adjacency matrix $\mathbf{A} \approx \mathbf{U}_1\mathbf{V}_1$ by minimizing $\|\mathbf{A} - \mathbf{U}_1\mathbf{V}_1\|_F^2 + \|\mathbf{V}_1 - \mathbf{U}_1^T\mathbf{A}\|_F^2$, where $\mathbf{U}_1 \in \mathbb{R}_+^{n\times r_1}$ and $\mathbf{V}_1 \in \mathbb{R}_+^{r_1\times n}$. Then, we decompose the matrix $\mathbf{V}_1$ as $\mathbf{V}_1 \approx \mathbf{U}_2\mathbf{V}_2$ by minimizing $\|\mathbf{V}_1 - \mathbf{U}_2\mathbf{V}_2\|_F^2 + \|\mathbf{V}_2 - \mathbf{U}_2^T\mathbf{V}_1\|_F^2$, where $\mathbf{U}_2 \in \mathbb{R}_+^{r_1\times r_2}$ and $\mathbf{V}_2 \in \mathbb{R}_+^{r_2\times n}$. Continue to do so until all of the layers have been pre-trained. Afterwards, each layer is fine-tuned by alternating minimization of the proposed objective function in Eq. (6). In the following, we present the updating rules.

4.1 Updating Rules

4.1.1 Updating rule for the mapping matrix $\mathbf{U}_i$ $(1 \le i \le p)$. By fixing all the variables except for $\mathbf{U}_i$, the objective function in Eq. (6) is reduced to:

$$\begin{aligned}\min_{\mathbf{U}_i}\ & \mathcal{L}(\mathbf{U}_i) = \|\mathbf{A} - \mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{V}_p\|_F^2 \\ & + \|\mathbf{V}_p - \mathbf{\Phi}_{i+1}^T\mathbf{U}_i^T\mathbf{\Psi}_{i-1}^T\mathbf{A}\|_F^2, \\ \text{s.t. }& \mathbf{U}_i \ge \mathbf{0},\end{aligned} \tag{7}$$

where $\mathbf{\Psi}_{i-1} = \mathbf{U}_1\mathbf{U}_2\cdots\mathbf{U}_{i-1}$ and $\mathbf{\Phi}_{i+1} = \mathbf{U}_{i+1}\cdots\mathbf{U}_{p-1}\mathbf{U}_p$. When $i = 1$, we set $\mathbf{\Psi}_0 = \mathbf{I}$. Similarly, when $i = p$, we set $\mathbf{\Phi}_{p+1} = \mathbf{I}$.

To solve Eq. (7), we introduce a Lagrangian multiplier matrix $\mathbf{\Theta}_i$ to enforce the nonnegative constraints on $\mathbf{U}_i$, resulting in the following equivalent objective function:

$$\begin{aligned}\min_{\mathbf{U}_i,\mathbf{\Theta}_i}\ & \mathcal{L}(\mathbf{U}_i,\mathbf{\Theta}_i) = \|\mathbf{A} - \mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{V}_p\|_F^2 \\ & + \|\mathbf{V}_p - \mathbf{\Phi}_{i+1}^T\mathbf{U}_i^T\mathbf{\Psi}_{i-1}^T\mathbf{A}\|_F^2 - tr(\mathbf{\Theta}_i\mathbf{U}_i^T),\end{aligned} \tag{8}$$

which can be further rewritten as follows:

$$\begin{aligned}\min_{\mathbf{U}_i,\mathbf{\Theta}_i} \mathcal{L}(\mathbf{U}_i,\mathbf{\Theta}_i) = tr(&\mathbf{A}^T\mathbf{A} + \mathbf{V}_p^T\mathbf{V}_p - 4\mathbf{A}^T\mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{V}_p \\ & + \mathbf{V}_p^T\mathbf{\Phi}_{i+1}^T\mathbf{U}_i^T\mathbf{\Psi}_{i-1}^T\mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{V}_p \\ & + \mathbf{A}^T\mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{\Phi}_{i+1}^T\mathbf{U}_i^T\mathbf{\Psi}_{i-1}^T\mathbf{A} - \mathbf{\Theta}_i\mathbf{U}_i^T).\end{aligned} \tag{9}$$

By setting the partial derivative of $\mathcal{L}(\mathbf{U}_i,\mathbf{\Theta}_i)$ with respect to $\mathbf{U}_i$ to $\mathbf{0}$, we have:

$$\mathbf{\Theta}_i = -4\mathbf{\Psi}_{i-1}^T\mathbf{A}\mathbf{V}_p^T\mathbf{\Phi}_{i+1}^T + 2\mathbf{\Pi}_i, \tag{10}$$

where

$$\begin{aligned}\mathbf{\Pi}_i = &\mathbf{\Psi}_{i-1}^T\mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{V}_p\mathbf{V}_p^T\mathbf{\Phi}_{i+1}^T \\ & + \mathbf{\Psi}_{i-1}^T\mathbf{A}\mathbf{A}^T\mathbf{\Psi}_{i-1}\mathbf{U}_i\mathbf{\Phi}_{i+1}\mathbf{\Phi}_{i+1}^T.\end{aligned} \tag{11}$$

From the complementary slackness condition of the Karush-Kuhn-Tucker (KKT) conditions [4], we obtain:

$$\mathbf{\Theta}_i \odot \mathbf{U}_i = (-4\mathbf{\Psi}_{i-1}^T\mathbf{A}\mathbf{V}_p^T\mathbf{\Phi}_{i+1}^T + 2\mathbf{\Pi}_i) \odot \mathbf{U}_i = \mathbf{0}, \tag{12}$$

where $\odot$ denotes the element-wise product. Equation (12) is the fixed point equation that the solution must satisfy at convergence. By solving this equation, we derive the following updating rule for $\mathbf{U}_i$:

$$\mathbf{U}_i \leftarrow \mathbf{U}_i \odot \frac{2\mathbf{\Psi}_{i-1}^T\mathbf{A}\mathbf{V}_p^T\mathbf{\Phi}_{i+1}^T}{\mathbf{\Pi}_i}. \tag{13}$$

4.1.2 Updating rule for the community membership matrix $\mathbf{V}_p$. By fixing all the variables except for $\mathbf{V}_p$, the objective function in Eq. (6) is reduced to:

$$\begin{aligned}\min_{\mathbf{V}_p}\ & \mathcal{L}(\mathbf{V}_p) = \|\mathbf{A} - \mathbf{\Psi}_p\mathbf{V}_p\|_F^2 + \|\mathbf{V}_p - \mathbf{\Psi}_p^T\mathbf{A}\|_F^2 \\ & + \lambda tr(\mathbf{V}_p\mathbf{L}\mathbf{V}_p^T), \\ \text{s.t. }& \mathbf{V}_p \ge \mathbf{0}.\end{aligned} \tag{14}$$

Following similar derivation process of the updating rule for $\mathbf{U}_i$, the updating rule for $\mathbf{V}_p$ is formulated as follows:

$$\mathbf{V}_p \leftarrow \mathbf{V}_p \odot \frac{2\mathbf{\Psi}_p^T\mathbf{A} + \lambda\mathbf{V}_p\mathbf{A}}{\mathbf{\Psi}_p^T\mathbf{\Psi}_p\mathbf{V}_p + \mathbf{V}_p + \lambda\mathbf{V}_p\mathbf{D}}. \tag{15}$$

4.1.3 Updating rule for the feature matrix $\mathbf{V}_i$ $(1 \le i < p)$. The updating of $\mathbf{V}_i$ is optional, since it does not affect the value of the objective function in Eq. (6). However, we would like to extract the hidden attributes in each intermediate layer. To optimize $\mathbf{V}_i$, we in fact seek to optimize the following objective function:

$$\begin{aligned}\min_{\mathbf{V}_i}\ & \mathcal{L}(\mathbf{V}_i) = \|\mathbf{A} - \mathbf{\Psi}_i\mathbf{V}_i\|_F^2 + \|\mathbf{V}_i - \mathbf{\Psi}_i^T\mathbf{A}\|_F^2, \\ \text{s.t. }& \mathbf{V}_i \ge \mathbf{0}.\end{aligned} \tag{16}$$

Similar to $\mathbf{V}_p$, $\mathbf{V}_i$ can be updated by

$$\mathbf{V}_i \leftarrow \mathbf{V}_i \odot \frac{2\mathbf{\Psi}_i^T\mathbf{A}}{\mathbf{\Psi}_i^T\mathbf{\Psi}_i\mathbf{V}_i + \mathbf{V}_i}. \tag{17}$$

Until now, we have all the updating rules done. The overall optimization process of DANMF is outlined in Algorithm 1, where the "ShallowNMF" procedure performs the pre-training as described earlier.

4.2 Convergence Analysis

The convergence of the updating rules is guaranteed by the following two theorems.

THEOREM 4.1. *The limited solutions of the updating rules in Eq.* (13) *and Eq.* (15) *satisfy the KKT optimality condition.*

PROOF. At convergence, we have $\mathbf{U}_i^{(\infty)} = \mathbf{U}_i^{(t+1)} = \mathbf{U}_i^{(t)} = \mathbf{U}_i$, where t denotes the t-th iteration. That is,

$$\mathbf{U}_i = \mathbf{U}_i \odot \frac{2\mathbf{\Psi}_{i-1}^T\mathbf{A}\mathbf{V}_p^T\mathbf{\Phi}_{i+1}^T}{\mathbf{\Pi}_i}, \tag{18}$$

Algorithm 1 Optimization algorithm of DANMF

Input: The adjacency matrix of network $\mathcal{G}$, $\mathbf{A}$;
The layer size of each layer, r_i;
The regularization parameter, λ;
Output: The mapping matrix $\mathbf{U}_i$ $(1 \leq i \leq p)$, the feature matrix $\mathbf{V}_i$ $(1 \leq i < p)$, and the community membership matrix $\mathbf{V}_p$;

▷ **Pre-training process:**
$\mathbf{U}_1, \mathbf{V}_1 \leftarrow \text{ShallowNMF}(\mathbf{A}, r_1)$;
for $i = 2$ **to** p **do**
  $\mathbf{U}_i, \mathbf{V}_i \leftarrow \text{ShallowNMF}(\mathbf{V}_{i-1}, r_i)$;
end for
▷ **Fine-tuning process:**
while not converged **do**
  for $i = 1$ **to** p **do**
    $\mathbf{\Psi}_{i-1} \leftarrow \prod_{\tau=1}^{i-1} \mathbf{U}_\tau$ $(\mathbf{\Psi}_0 \leftarrow \mathbf{I})$;
    $\mathbf{\Phi}_{i+1} \leftarrow \prod_{\tau=i+1}^{p} \mathbf{U}_\tau$ $(\mathbf{\Phi}_{p+1} \leftarrow \mathbf{I})$;
    Update $\mathbf{U}_i$ according to Eq. (13);
    $\mathbf{\Psi}_i \leftarrow \mathbf{\Psi}_{i-1}\mathbf{U}_i$;
    Update $\mathbf{V}_i$ according to Eq. (17) $(i < p, \text{optional})$ or according to Eq. (15) $(i = p)$;
  end for
end while
return $\mathbf{U}_i$, $\mathbf{V}_i$, $\forall i = 1, 2, \cdots, p$;

which is equivalent to

$$(-4\mathbf{\Psi}_{i-1}^T \mathbf{A}\mathbf{V}_p^T \mathbf{\Phi}_{i+1}^T + 2\mathbf{\Pi}_i) \odot \mathbf{U}_i = \mathbf{0}. \tag{19}$$

Clearly, Eq. (19) is identical to Eq. (12). In the same way, the correctness of the updating rule in Eq. (15) for $\mathbf{V}_p$ can be proved. □

Theorem 4.2. *The objective function $\mathcal{L}$ in Eq.* (6) *is non-increasing under the updating rules in Eq.* (13) *and Eq.* (15).

The theorem above can be proved by leveraging an auxiliary function, following a similar process as described in [17]. To save space, we omit the proof here.

4.3 Time Complexity

Algorithm 1 is composed of two stages, i.e., the pre-training stage and the fine-tuning stage. The computational complexity for the pre-training stage is of order $\mathcal{O}(pt_p(n^2r + nr^2))$, where p is the number of layers, t_p is the number of iterations to achieve convergence, and r is the maximal layer size out of all layers. The computational complexity for the fine-tuning stage is of order $\mathcal{O}(pt_f(n^2r + nr^2 + r^3))$, where t_f is the number of iterations in the fine-tuning process. In general, $r < n$, thus the complexity is $\mathcal{O}(pt_f(n^2r + nr^2))$. To sum up, the overall time complexity is $\mathcal{O}(p(t_p + t_f)(n^2r + nr^2))$.

4.4 Discussion

Our DANMF model is closely related to orthogonal NMF (ONMF) [27] and projective NMF (PNMF) [46]. As DANMF aims to optimize the encoder component and the decoder component simultaneously, we have $\mathbf{V}_p \approx \mathbf{\Psi}_p^T \mathbf{A}$ and $\mathbf{A} \approx \mathbf{\Psi}_p \mathbf{V}_p$. Then, we have $\mathbf{V}_p \approx \mathbf{\Psi}_p^T \mathbf{\Psi}_p \mathbf{V}_p$, which requires that $\mathbf{\Psi}_p^T \mathbf{\Psi}_p \approx \mathbf{I}$. In this sense, DANMF is related to ONMF. On the other hand, we have $\mathbf{A} \approx \mathbf{\Psi}_p \mathbf{\Psi}_p^T \mathbf{A}$, which leads to the PNMF model. However, both ONMF and PNMF are shallow models.

5 EXPERIMENTS

Now we move forward to evaluate the performance of the proposed DANMF model for disjoint community detection and overlapping community detection. All experiments are conducted on a server with two 2.4GHz Intel Xeon CPUs and 128GB main memory running Ubuntu 14.04.5 (64-bit).

5.1 Baseline Methods

Our basic hypothesis in this paper is that DANMF is able to learn a better community-level similarity between nodes (i.e., a more accurate community membership matrix) than shallow NMF-based community detection approaches, with the aid of a hierarchy of hidden attributes extracted in the intermediate layers of the autoencoder-like deep structure. To verify this hypothesis, we choose seven representative shallow NMF-based methods as baselines. We also compare DANMF with three state-of-the-art network embedding methods based on the considerations that these methods can learn higher-order similarity between nodes and that they are also closely related to matrix factorization [29].

The NMF-based shallow models include:

- **NMF:** NMF is the fundamental component of the proposed DANMF model. It has been adopted for community detection in [21].
- **ONMF:** ONMF is a variant of NMF by enforcing orthogonal constraints on the mapping matrix $\mathbf{U}$, i.e., $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ [27].
- **PNMF:** PNMF directly projects the original network to a subspace by minimizing $\|\mathbf{A} - \mathbf{U}\mathbf{U}^T\mathbf{A}\|_F^2$ [46].
- **BNMF:** BNMF is a bayesian NMF model. It has been adopted for community detection in [28].
- **BigClam:** BigClam is a cluster affiliation model, which relaxes the graph fitting problem into a continuous optimization problem [43].
- **HNMF:** HNMF is a probabilistic approach. It models the homogeneous relationships between edges and communities for community detection [48].
- **NSED:** NSED is a nonnegative symmetric encoder-decoder approach proposed for community detection. Though it takes into account the encoder component, it extracts the community membership from the mapping matrix $\mathbf{U}$ rather than the feature matrix $\mathbf{V}$ [33].

The network embedding methods include:

- **LINE:** LINE preserves the first-order and second-order proximities between nodes for learning low-dimensional representations of nodes [34].
- **Node2Vec:** Node2Vec aims to learn higher-order similarity between nodes via truncated random walks [12]. The in-out hyperparameter is fixed at 2 to better capture the community structure of networks.

Table 1: Layers configuration of DANMF

Dataset	n	Layers Configuration
Email	1005	1005-256-128-42
Wiki	2405	2405-256-128-19
Cora	2708	2708-256-64-7
Citeseer	3312	3312-256-64-6
Pubmed	19717	19717-512-64-3

Table 2: Error comparison on Wiki

Method	Encoder	Decoder	Encoder+Decoder
NMF	0.2326	0.0543	0.2869
ONMF	0.0043	0.0547	0.0590
PNMF	0.0000	445.58	445.58
NSED	0.0025	0.0547	0.0572
DNMF	0.2131	0.0546	0.2677
DANMF	0.0020	0.0541	0.0561

- **MNMF:** MNMF is a modularized NMF model, which incorporates the community structure into network embedding [40].

For the network embedding methods, we set the size of latent representations to be 64, and then apply the standard k-means algorithm to identify communities. We also implement a pruned version of DANMF, named **DNMF**, which ignores the encoder component. For a fair comparison, we run each algorithm 20 times and the average results are reported.

5.2 Disjoint Community Detection

5.2.1 Datasets. We adopt five real-world networks[1] for disjoint community detection.

- **Email:** A communication network involving 1005 researchers from 42 departments and 25571 relationships.
- **Wiki:** A document network consisting of 2405 web pages from 19 categories and 17981 edges.
- **Cora:** A citation network with 2708 nodes and 5429 edges. Each node is classified into one of 7 classes.
- **Citeseer:** A citation network with 3312 nodes and 4732 edges. Each node is classified into one of 6 classes.
- **Pubmed:** A citation network with 19717 nodes and 44338 edges. Each node is divided into one of 3 classes.

5.2.2 Community detection results. To measure the community detection results, we employ three evaluation metrics including Adjusted Rand Index (**ARI**), Normalized Mutual Information (**NMI**), and Accuracy (**ACC**). For these metrics, larger value indicates better performance. A detailed description of them can be found in the survey paper [6]. The regularization parameter of DANMF is tuned in the range of $\{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}\}$. And the layer size configuration of DANMF is provided in Table 1. We implement DANMF with three hidden layers. Although we have experimented with more hidden layers, the performance promotion is not significant while much more time is taken to train the model.

[1]See https://snap.stanford.edu/ and https://linqs.soe.ucsc.edu.

Table 3: Performance evaluation based on ARI

Method	Email	Wiki	Cora	Citeseer	Pubmed
NMF	0.4989	0.1195	0.2145	0.0590	0.0978
ONMF	0.4832	0.1233	0.1964	0.0825	0.1589
PNMF	0.4641	0.1151	0.1863	0.0801	0.0967
BNMF	0.3545	0.1705	0.1812	0.0838	0.0872
BigClam	0.2478	0.0217	0.0306	0.0283	0.0258
HNMF	0.2079	0.1448	0.1113	0.0262	0.0360
NSED	0.5215	0.1253	0.1782	0.0866	0.1258
LINE	0.3325	0.1344	0.1271	0.0278	0.1017
Node2Vec	0.4195	0.1621	0.1063	0.0182	0.0170
MNMF	0.0041	0.0016	0.0002	0.0007	0.0001
DNMF	0.5256	0.1341	0.2452	0.0990	0.1185
DANMF	0.5521	0.1628	0.3194	0.1343	0.2563

Table 4: Performance evaluation based on NMI

Method	Email	Wiki	Cora	Citeseer	Pubmed
NMF	0.6751	0.2673	0.2851	0.1319	0.1606
ONMF	0.6734	0.2607	0.2416	0.1423	0.1582
PNMF	0.6770	0.2684	0.2893	0.1355	0.1511
BNMF	0.5960	0.2903	0.2521	0.0835	0.0714
BigClam	0.5796	0.2722	0.1864	0.0735	0.0291
HNMF	0.5146	0.2959	0.1425	0.0312	0.0311
NSED	0.6845	0.2659	0.2928	0.1492	0.1729
LINE	0.6393	0.2772	0.2376	0.0573	0.1357
Node2Vec	0.6784	0.3331	0.1978	0.0486	0.0635
MNMF	0.2138	0.0274	0.0035	0.0031	0.0002
DNMF	0.6850	0.2798	0.3572	0.1582	0.1709
DANMF	0.6943	0.3406	0.4114	0.1831	0.2221

Since we assume that DANMF is able to better inherit the learning capability of deep autoencoder, our first experiment is to evaluate whether it achieves lower coding and reconstruction error. The coding error corresponding to the encoder component is calculated as $\frac{1}{n}\|\mathbf{V} - \mathbf{U}^T\mathbf{A}\|_F$, where n denotes the number of nodes. The reconstruction error corresponding to the decoder component is calculated as $\frac{1}{n}\|\mathbf{A} - \mathbf{UV}\|_F$. Note that in PNMF, we set $\mathbf{V} = \mathbf{U}^T\mathbf{A}$ directly. In DNMF and DANMF, we have $\mathbf{V} = \mathbf{V}_p$ and $\mathbf{U} = \mathbf{U}_1\mathbf{U}_2\cdots\mathbf{U}_p$. The average results of 20 runs on Wiki is reported in Table 2. We only report the results of the methods that involve the mapping $\mathbf{U}$ explicitly. The results show that DANMF achieves much lower coding error than NMF and DNMF, which verifies the necessity of the encoder component. Although DANMF is much harder to train due to the multiple factor matrices, it achieves comparable reconstruction error with NMF, which demonstrates the effectiveness of our optimization algorithm. Although the coding error of PNMF is 0, its reconstruction error is extremely large.

Next we introduce the community detection results. Tabels 3-5 show the comparison in ARI, NMI, and ACC respectively. The best results are presented in blue color. As can be seen, our DANMF model outperforms all the baselines across different evaluation metrics except for ARI on Wiki. For example, on the largest network Pubmed, DANMF achieves

Table 5: Performance evaluation based on ACC

Method	Email	Wiki	Cora	Citeseer	Pubmed
NMF	0.5851	0.3027	0.4103	0.3074	0.5133
ONMF	0.5761	0.3069	0.3811	0.3330	0.5575
PNMF	0.5791	0.3052	0.4029	0.3451	0.5073
BNMF	0.4299	0.3751	0.4191	0.3324	0.5110
BigClam	0.4768	0.2545	0.3781	0.3046	0.3978
HNMF	0.3463	0.3518	0.3903	0.2569	0.4128
NSED	0.6179	0.2981	0.4234	0.3448	0.5201
LINE	0.4657	0.3289	0.4044	0.3019	0.4990
Node2Vec	0.5244	0.3568	0.3674	0.2521	0.4067
MNMF	0.1075	0.0886	0.1647	0.1890	0.3397
DNMF	0.6199	0.3543	0.4849	0.3635	0.5389
DANMF	0.6358	0.4112	0.5499	0.4242	0.6393

a relative performance promotion of 9.74%, 4.92% and 8.18% with respect to ARI, NMI and ACC respectively. It is noted that DNMF also outperforms NMF consistently, which shows that with the deep structure, we are indeed able to learn a hierarchy of abstract understanding of the original networks that can aid in uncovering the community membership of nodes. The superiority of DANMF over DNMF further verifies that by integrating the encoder component and the decoder component, DANMF is able to better inherit the learning capability of deep autoencoder. One may note that the network embedding methods do not show satisfactory performance, even though they seek to preserve higher-order similarity between nodes. The reason for LINE and Node2Vec is that they are primarily focused on modeling the microscopic structure instead of the mesoscopic community structure of networks. The reason for MNMF may be that it adopts modularity to reveal the community structure. However, modularity may suffer from the resolution limit problem [6].

Although DANMF has a deep structure, it can be trained efficiently. The runtime of DANMF on all the benchmark networks is depicted in Figure 3. Note that Figure 3 is plotted in log scale. It is observed that DANMF is quite efficient on small networks. On the largest network Pubmed, DANMF can also finish its training process in about 4000 seconds.

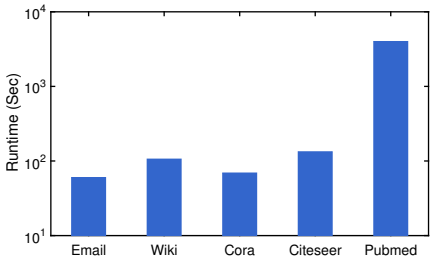
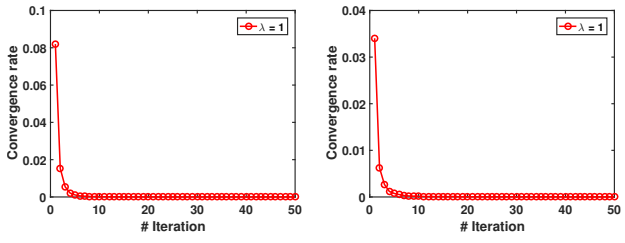


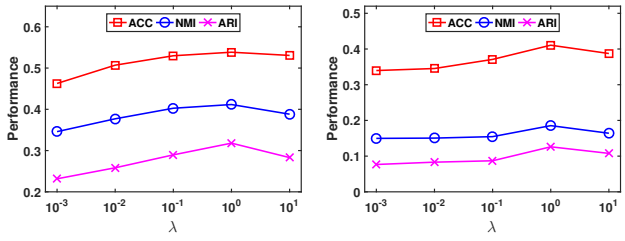
Figure 3: Time overheads of DANMF.

5.2.3 Convergence analysis. The updating rules of our optimization algorithm are essentially iterative. Different from the exact runtime, here we further investigate how fast these rules can converge. Recall that our optimization algorithm consists of the pre-training stage and the fine-tuning stage. In the pre-training stage, each layer in fact performs a shallow NMF model, whose convergence has already been analyzed [33]. Thus we focus on the fine-tuning stage and analyze its convergence rate, which denotes the change rate of the objective function value. To test the convergence speed, we fix the regularization parameter at 1. The results on Cora and Citeseer are shown in Figure 4. Similar results can be observed on other networks. From Figure 4, we can see that DANMF can achieve fast convergence within about 10 iterations.



(a) Cora (b) Citeseer

Figure 4: Convergence rate analysis.



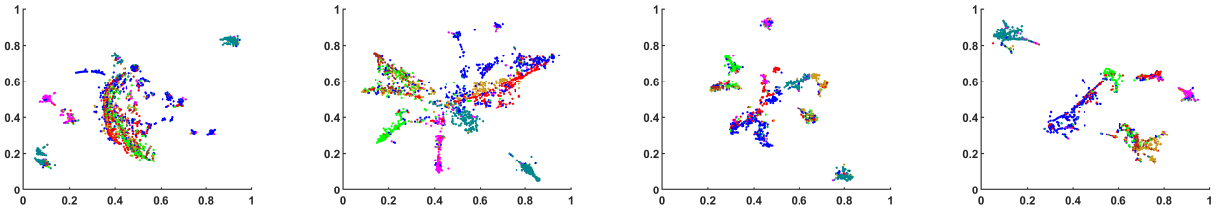
(a) Cora (b) Citeseer

Figure 5: The effect of parameter λ.

5.2.4 Parameter sensitivity. In DANMF, the parameter λ is used to adjust the contribution of the graph regularizer. It is tuned in the range of $\{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}\}$. The effect of λ on Cora and Citeseer is shown in Figure 5. To some extent, DANMF is robust to the parameter λ. On both Cora and Citeseer, DANMF tends to obtain the best performance when $\lambda = 1$. The results indicate that although the performance of DANMF is stable with respect to λ, a proper λ can make DANMF be more robust.

There is no doubt that the performance of DANMF will be affected by the layer size configuration of each layer, an in-depth exploration of which is left as our future work.

5.2.5 Visualization. As DANMF is expected to be able to learn the similarity between nodes from different levels of granularity, we feed the learnt hidden attributes (i.e., the feature matrix $\mathbf{V}_i$ and the community membership matrix $\mathbf{V}_p$) into the standard t-SNE tool [20] to visualize them. For comparison, we also visualize the original network. The result on Cora is shown in Figure 6, where nodes belonging to the same community share the same color. It is observed that the original network represented by the adjacency matrix does not embody clear community structure. While the hidden attributes learnt in the intermediate layers of DANMF capture the similarity between nodes more accurately. Besides, nodes belonging to the same community gather more and more closer to each other as the layers go deeper.



(a) Cora - before training (b) Cora - layer 1 (c) Cora - layer 2 (d) Cora - layer 3

Figure 6: 2D visualization of the representations learnt in different layers of DANMF on Cora.

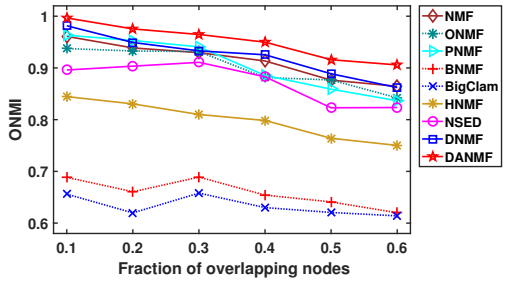


Figure 7: ONMI on LFR benchmarks with $\mu = 0.1$.

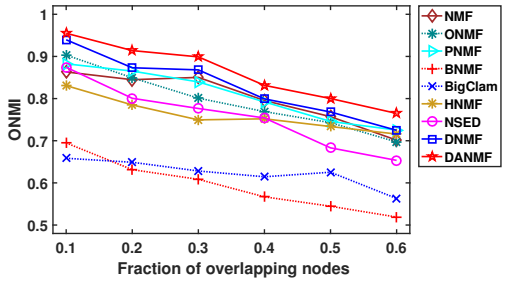


Figure 8: ONMI on LFR benchmarks with $\mu = 0.3$.

5.3 Overlapping Community Detection

5.3.1 Datasets. Since few networks with ground-truth overlapping communities are publically available, we employ the well-known LFR toolkit [16] to generate synthetic networks with overlapping community structure. The parameters of the LFR benchmarks are set as follows. The number of nodes is 5000, the average degree is 20, and the maximum degree is 50. The community size ranges from 100 to 250. The exponents of the power-law distributions of node degree and community size are kept at 2 and 1, respectively. The number of communities that an overlapping node belongs to is fixed at 2. The mixing parameter μ (each node shares a fraction μ of its edges with nodes in other communities) is set to either 0.1 or 0.3, and the fraction of overlapping nodes varies from 0.1 to 0.6 with an increment of 0.1. Thus, there are 12 synthetic networks in total. For each network, the layer size configuration of DANMF is set to 5000-512-128-k, where k denotes the number of ground-truth communities.

5.3.2 Community detection results. Following [16], we use the Overlapping NMI (**ONMI**) metric to evaluate the overlapping community detection results. The detailed description of ONMI can also be found in the survey paper [6]. The results with respect to $\mu = 0.1$ and $\mu = 0.3$ are shown in Figure 7 and Figure 8, respectively. Since the network embedding methods are not suitable for overlapping community detection, their results are neglected. As shown in Figure 7 and Figure 8, DANMF outperforms the other methods on all the LFR benchmark networks. When the mixing parameter $\mu = 0.1$, the performance of DANMF is relatively stable with the change of the fraction of overlapping nodes. For example, even on the network with 60% overlapping nodes, the ONMI of DANMF reaches 0.9. While the performance of all the methods on the LFR benchmark networks with $\mu = 0.3$ drops off precipitously as the fraction of overlapping nodes increases. This is because when $\mu = 0.3$, the community structure becomes less significant, which makes the community detection task more difficult and more challenging. However, DANMF still shows superior performance over the other methods consistently. The results demonstrate that DANMF is capable of detecting overlapping communities with better performance.

6 CONCLUSION

In this paper, we have introduced a novel deep autoencoder-like model DANMF to combat the problem of community detection. Different from traditional NMF-based community detection methods, DANMF integrates the encoder component and the decoder component into a unified loss function. Both components are with deep structures. This architecture empowers DANMF to better inherit the learning capability of deep autoencoder. Although DANMF is much harder to train due to the multiple factor matrices, the proposed optimization algorithm can solve it efficiently. We have also conducted extensive experiments for both disjoint community detection and overlapping community detection. The results demonstrate the superiority of DANMF over shallow NMF-based methods. For future work, we plan to use other cost functions to quantify the quality of the approximation, e.g., the Kullback-Leibler divergence.

ACKNOWLEDGMENTS

This work was supported by the National Key Research and Development Plan (2018YFB1003800), the National Natural Science Foundation of China (11801595), the Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme 2016, the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2016ZT06D211)

and the Pearl River S&T Nova Program of Guangzhou (201710010046). Chuan Chen is the corresponding author.

REFERENCES

[1] Michael J Barber. 2007. Modularity and community detection in bipartite networks. *Physical Review E* 76, 6 (2007), 066102.

[2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE T-PAMI* 35, 8 (2013), 1798–1828.

[3] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *J. Stat. Mech.* 2008, 10 (2008), P10008.

[4] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[5] Sandro Cavallari, Vincent W Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. 2017. Learning community embedding with community detection and node embedding on graphs. In *CIKM*. ACM, 377–386.

[6] Tanmoy Chakraborty, Ayushi Dalmia, Animesh Mukherjee, and Niloy Ganguly. 2017. Metrics for community analysis: A survey. *ACM computing curveys (csur)* 50, 4 (2017), 54.

[7] Tanmoy Chakraborty, Sriram Srinivasan, Niloy Ganguly, Animesh Mukherjee, and Sanjukta Bhowmick. 2014. On the permanence of vertices in network communities. In *KDD*. ACM, 1396–1405.

[8] Marek Ciglan, Michal Laclavík, and Kjetil Nørvåg. 2013. On community detection in real-world networks and the importance of degree assortativity. In *KDD*. ACM, 1007–1015.

[9] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *SIGMOD*. ACM, 991–1002.

[10] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3-5 (2010), 75–174.

[11] Michelle Girvan and Mark EJ Newman. 2002. Community structure in social and biological networks. *PNAS* 99, 12 (2002), 7821–7826.

[12] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. ACM, 855–864.

[13] Zhenxing Guo and Shihua Zhang. 2017. Sparse deep nonnegative matrix factorization. *arXiv preprint arXiv:1707.09316* (2017).

[14] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18, 7 (2006), 1527–1554.

[15] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.

[16] Andrea Lancichinetti, Santo Fortunato, and János Kertész. 2009. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics* 11, 3 (2009), 033015.

[17] Daniel D Lee and H Sebastian Seung. 2001. Algorithms for non-negative matrix factorization. In *NIPS*. 556–562.

[18] Jure Leskovec, Kevin J Lang, and Michael Mahoney. 2010. Empirical comparison of algorithms for network community detection. In *WWW*. ACM, 631–640.

[19] Tianshu Lyu, Yuan Zhang, and Yan Zhang. 2017. Enhancing the network embedding quality with structural similarity. In *CIKM*. ACM, 147–156.

[20] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *JMLR* 9, Nov (2008), 2579–2605.

[21] Shawn Mankad and George Michailidis. 2013. Structural and functional discovery in dynamic networks with non-negative matrix factorization. *Physical Review E* 88, 4 (2013), 042812.

[22] Mark EJ Newman. 2003. The structure and function of complex networks. *SIAM review* 45, 2 (2003), 167–256.

[23] Mark EJ Newman. 2004. Fast algorithm for detecting community structure in networks. *Physical review E* 69, 6 (2004), 066133.

[24] Mark EJ Newman. 2006. Modularity and community structure in networks. *PNAS* 103, 23 (2006), 8577–8582.

[25] Nam P Nguyen, Thang N Dinh, Sindhura Tokala, and My T Thai. 2011. Overlapping communities in dynamic networks: Their detection and mobile applications. In *MobiCom*. ACM, 85–96.

[26] Yulong Pei, Nilanjan Chakraborty, and Katia Sycara. 2015. Nonnegative matrix tri-factorization with graph regularization for community detection in social networks. In *IJCAI*. 2083–2089.

[27] Filippo Pompili, Nicolas Gillis, P-A Absil, and François Glineur. 2014. Two algorithms for orthogonal nonnegative matrix factorization with application to clustering. *Neurocomputing* 141 (2014), 15–25.

[28] Ioannis Psorakis, Stephen Roberts, Mark Ebden, and Ben Sheldon. 2011. Overlapping community detection using bayesian non-negative matrix factorization. *Physical Review E* 83, 6 (2011), 066114.

[29] Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *WSDM*. ACM, 459–467.

[30] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *KDD*. ACM, 385–394.

[31] Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE T-PAMI* 22, 8 (2000), 888–905.

[32] Hyun Ah Song, Bo-Kyeong Kim, Thanh Luong Xuan, and Soo-Young Lee. 2015. Hierarchical feature extraction by multi-layer non-negative matrix factorization network for classification task. *Neurocomputing* 165 (2015), 63–74.

[33] Bing-Jie Sun, Huawei Shen, Jinhua Gao, Wentao Ouyang, and Xueqi Cheng. 2017. A non-negative symmetric encoder-decoder approach for community detection. In *CIKM*. ACM, 597–606.

[34] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *WWW*. ACM, 1067–1077.

[35] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu. 2014. Learning deep representations for graph clustering. In *AAAI*. 1293–1299.

[36] Vincent A Traag and Jeroen Bruggeman. 2009. Community detection in networks with positive and negative links. *Physical Review E* 80, 3 (2009), 036115.

[37] George Trigeorgis, Konstantinos Bousmalis, Stefanos Zafeiriou, and Bjoern Schuller. 2014. A deep semi-nmf model for learning hidden representations. In *ICML*. 1692–1700.

[38] Nate Veldt, David F Gleich, and Anthony Wirth. 2018. A correlation clustering framework for community detection. In *WWW*. ACM, 439–448.

[39] Fei Wang, Tao Li, Xin Wang, Shenghuo Zhu, and Chris Ding. 2011. Community discovery using nonnegative matrix factorization. *DMKD* 22, 3 (2011), 493–521.

[40] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding. In *AAAI*. 203–209.

[41] Wenhui Wu, Sam Kwong, Yu Zhou, Yuheng Jia, and Wei Gao. 2018. Nonnegative matrix factorization with mixed hypergraph regularization for community detection. *Information Sciences* 435 (2018), 263–281.

[42] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski. 2013. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM CSUR* 45, 4 (2013), 43.

[43] Jaewon Yang and Jure Leskovec. 2013. Overlapping community detection at scale: A nonnegative matrix factorization approach. In *WSDM*. ACM, 587–596.

[44] Jaewon Yang, Julian McAuley, and Jure Leskovec. 2013. Community detection in networks with node attributes. In *ICDM*. IEEE, 1151–1156.

[45] Jinshi Yu, Guoxu Zhou, Andrzej Cichocki, and Shengli Xie. 2018. Learning the hierarchical parts of objects by deep non-smooth nonnegative matrix factorization. *arXiv preprint arXiv:1803.07226* (2018).

[46] Zhijian Yuan and Erkki Oja. 2005. Projective nonnegative matrix factorization for image compression and feature extraction. In *Scandinavian Conference on Image Analysis*. Springer, 333–342.

[47] Hongyi Zhang, Irwin King, and Michael R Lyu. 2015. Incorporating implicit link preference into overlapping community detection.. In *AAAI*. 396–402.

[48] Hongyi Zhang, Tong Zhao, Irwin King, and Michael R Lyu. 2016. Modeling the homophily effect between links and communities for overlapping community detection.. In *IJCAI*. 3938–3944.

[49] Yu Zhang and Dit-Yan Yeung. 2012. Overlapping community detection via bounded nonnegative matrix tri-factorization. In *KDD*. ACM, 606–614.

[50] Zhong-Yuan Zhang, Yong Wang, and Yong-Yeol Ahn. 2013. Overlapping community detection in complex networks using symmetric binary matrix factorization. *Physical Review E* 87, 6 (2013), 062803.

[51] Zibin Zheng, Fanghua Ye, Rong-Hua Li, Guohui Ling, and Tan Jin. 2017. Finding weighted k-truss communities in large networks. *Information Sciences* 417 (2017), 344–360.