Vicinal vertex allocation for matrix factorization in networks

He, Tiantian; Bai, Lu; Ong, Yew-Soon

2021

He, T., Bai, L. & Ong, Y. (2021). Vicinal vertex allocation for matrix factorization in networks. IEEE Transactions On Cybernetics.

<https://dx.doi.org/10.1109/TCYB.2021.3051606>

<https://hdl.handle.net/10356/147802>

<https://doi.org/10.1109/TCYB.2021.3051606>

© 2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at:
<https://doi.org/10.1109/TCYB.2021.3051606>.

Downloaded on 26 Apr 2023 13:47:59 SGT

Vicinal Vertex Allocation for Matrix Factorization in Networks

Tiantian He, Lu Bai, and Yew-Soon Ong, *Fellow, IEEE*

Abstract—In this paper, we present a novel matrix-factorization-based model, labeled here as Vicinal Vertex Allocated Matrix Factorization (VVAMo), for uncovering clusters in network data. Different from past related efforts of network clustering which consider edge structure, vertex features, or both in their design, the proposed model includes the additional detail on vertex inclinations w.r.t. topology and features into the learning. In particular, by taking the latent preferences between vicinal vertices into consideration, VVAMo is then able to uncover network clusters composed of proximal vertices that share analogous inclinations, and correspondingly high structural and feature correlations. To ensure such clusters to be effectively uncovered, we propose a unified likelihood function for VVAMo, and derive an alternating algorithm for optimizing the proposed function. Subsequently, we provide the theoretical analysis of VVAMo, including convergence proof and computational complexity analysis. To investigate the effectiveness of the proposed model, a comprehensive empirical study of VVAMo is conducted using extensive commonly used realistic network datasets. Results obtained show that VVAMo attained superior performances over existing classical and state-of-the-art approaches.

Index Terms—Network clustering, community detection, complex network, graph clustering, probabilistic matrix factorization, vicinal vertex allocated matrix factorization (VVAMo)

I. INTRODUCTION

RELATIONAL data are ubiquitous in the real world, and can be represented as a network, which contains a set of vertices and edges, respectively representing data samples and sample-sample relationships. Different from randomly generated networks, ones constructed based on real data possess latent structures, which may reflect the knowledge of interest behind the chaotic relations. Aiming at uncovering the latent structure indicating groups of cohesive vertices, network clustering has been one of the most challenging learning problems and has drawn much interest in the recent.

Network clustering is related to many real applications, such as social community detection [14], [21], document classification [7], [20], computer security [50], and biological module identification [1], [13], [29], [40]. To tackle it,

several approaches have been proposed. These methods can generally be categorized as heuristic-based and model-based ones. Heuristic methods can discover clusters in the network via searching groups of vertices satisfying some predefined measurement. For example, modularity [8], which evaluates the difference in terms of density between vertices in the same group and that if vertices are randomly distributed, is one of the most widely accepted measures for network clustering. Many approaches, e.g., Clauset-Newman-Moore algorithm [8], and Fast unfolding [5], have been proposed to uncover network clusters via optimizing modularity. Besides, particular sub-structures can also be used as heuristic measures for network clustering. For example, clique, which is a fully connected sub-network is considered by some approaches as the foundation of network clusters. Based on it, methods like Clique percolation [29], are proposed to perform cluster analysis in different network data.

More recent approaches for network clustering are model-based, and consider discovering network clusters as a learning process. Compared with heuristic ones, model-based methods for network clustering may perform the task by optimizing the formulated objective functions, which quantify the holistic quality of vertex segmentation in terms of edge density, and other additional information, e.g., vertex features. Many machine learning techniques, e.g., spectral analysis [35], [45], matrix factorization [15], [43], [44], Bayesian modeling [7], [30], and deep learning [6], [18], [25], [48], have been successfully adapted to perform the task of uncovering network clusters, taking into consideration the edge structure, vertex features, or both.

Though effective in discovering clusters in various types of networks, most previous approaches, which are either heuristic, or model-based, only utilize those observed data, e.g., network topology, vertex features, and similarities to perform the clustering tasks. Latent vertex-vertex relevance underlying the observed data is overlooked by most previous approaches. Consequently, the impact of the latent relevance on unfolding network clusters is not fully explored. In this paper, we attempt to investigate such latent vertex-vertex relevance, which is named as latent preference, and utilize it to uncover meaningful clusters in network data.

Taking a pair of connected users on the social networking site as an example, Fig. 1 graphically shows how to model the aforementioned latent preference between pairwise vertices in the network. As depicted, this pair of connected users (vertices) possess their own social relationships (topology) and descriptive profiles, e.g., tags and comments (vertex features). Accordingly, user-user correlations w.r.t. social relations and

Manuscript received May 2, 2020; revised October 13, 2020; accepted January 3 2021. This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2018-004), Data Science & Artificial Intelligence Research Centre, Nanyang Technological University, and the Singtel Cognitive and Artificial Intelligence Lab for Enterprises (SCALE@NTU), which is a collaboration between Singapore Telecommunications Limited (Singtel) and Nanyang Technological University (NTU) that is funded by the Singapore Government through the Industry Alignment Fund - Industry Collaboration Projects Grant. (Corresponding author: Lu Bai.)

T. He, L. Bai, and Y.S. Ong are with School of Computer Science and Engineering, Nanyang Technological University, Singapore, and also with Data Science & Artificial Intelligence Research Centre, Nanyang Technological University, Singapore. (e-mail: {tiantian.he, bailu, asysong}@ntu.edu.sg).

Fig. 1. Graphical view of how to define the vertex-vertex latent preference. The proposed latent preference is inferred over heterogeneous correlations and corresponding vertex-wise inclinations, not a simple mean of heterogeneous correlations. Highly correlated vertices (vicinal vertices) can thereby be identified.

descriptive information can be easily computed via a proper similarity metric. It is known that the simple mean of the computed heterogeneous correlations cannot appropriately represent the preference between pairwise users. Instead, for different users, the heterogeneous correlations potentially leading one to interact with others are considered to have different degrees of significance, which can be named as inclinations. Such instances widely exist in different scenarios in the real world. For example, a social network user may attempt to make friends directly according to the similarity of other's profiles. Correlations of social relations here become the secondary factor determining the establishment of friendship. Inspired by this phenomenon, we further define vertex-vertex latent preference, over the heterogeneous correlations and corresponding vertex-wise inclinations to unfold those highly correlative vertices.

It is obvious that grouping those vertices with high latent preferences in the process of network clustering may lead a model to discover interpretable clusters whose vertices lean to possess analogous edge structure and features. In this paper, we therefore propose a novel probabilistic matrix factorization model, which takes into consideration the degree of latent preference among proximal vertices on the allocation of vertex-cluster membership in the learning process. The contributions of this paper are summarized as follows:

- We propose a novel probabilistic matrix factorization model, dubbed Vicinal Vertex Allocated Matrix Factorization (VVAMo). Different from previous approaches, VVAMo is capable of inferring the latent neighboring preference over the heterogeneous vertex-vertex correlations and corresponding vertex-wise inclinations. Vertices sharing relatively high latent preferences are therefore more likely to be allocated into the same cluster by VVAMo. As a result, interpretable clusters can be discovered by the proposed model.
- A novel generative process is designed for VVAMo to generate network data. Given it, we formulate a unified likelihood function for VVAMo to perform the task of network clustering. We derive an effective alternating algorithm to learn the optimal parameters of the model.

In addition, the theoretical analysis on VVAMo, including convergence proof and computational complexity analysis is provided.

- VVAMo has been tested on various network data and has been compared with a number of strong baselines. The remarkable results validate the effectiveness of the proposed model.

The rest of the paper is organized as follows. In Section II, we investigate the prevalent works related to VVAMo, and summarize the differences between the proposed and the previous. In Section III, we elaborate VVAMo, and derive the iterative updating algorithm for model fitting. In Section IV, we theoretically prove the convergence of VVAMo and analyze the computational complexity of VVAMo. In Section V, the proposed model is validated on real world datasets, and compared with a number of strong baselines. Finally, we summarize the contributions of this paper and conclude future works that may further improve the proposed model.

II. RELATED WORK

To effectively discover clusters in network data, a number of approaches have been proposed. Either these methods are model-based, or heuristic-based, they can be categorized according to what data are utilized by them to uncover network clusters. Some approaches are able to uncover clusters utilizing network topology. For example, Clauset-Newman-Moore algorithm [8], Fast-unfolding [5], Clique percolation [29], and Community enhanced method [37], are classical heuristic-based approaches, which utilize edge structure to discover network clusters. Besides, Stochastic block models (SBM) [1], [30], [33] are popular Bayesian approaches, which are able to uncover network clusters through maximizing the posterior probability in terms of edge density within predefined blocks (clusters). Community through directed affiliations (CoDA) [47], Deep autoencoder-like nonnegative matrix factorization (DANMF) [48], and Discrete nonnegative matrix factorization (DNMF) [49] are popular models which are based on non-negative matrix factorization (NMF). Assuming the vertex-cluster preference is hidden in the low-dimensional latent space, these NMF-based approaches may perform the task of network clustering via minimizing the errors between the original adjacency matrix and the one constructed by the latent space. Other typical approaches, including Normalized cut (Ncut) [35], and Affinity propagation [10], can perform the task of network clustering utilizing edge weights that reveal the local similarity between connected vertices.

More recent methods attempt to discover network clusters as well as learn the descriptive features of them. Considering edge structure and vertex features as two sources of data, these approaches therefore formulate network clustering as a co-learning problem. And it can be effectively solved by inferring the low-dimensional latent space shared by edge structure and vertex features. Many machine learning techniques have been successfully adapted to solve the mentioned co-learning problem in network data. Inspired by topic modeling [4], several probabilistic models, including Relational topic model [7] and Itopic model [38] have been proposed to learn a

low-dimensional latent space hidden in network topology and vertex features. Communities from edge structure and node attributes (CESNA) [46] and Circles [26] are two prevalent probabilistic models, which can learn a latent space compactly representing the cluster preference for each vertex, regarding structural and feature similarity. Besides probabilistic modeling, other learning techniques, e.g., NMF, spectral analysis, and evolutionary computation, have also been used to solve the co-learning problem in network data. For example, Semantic community identification (SCI) [44], Adaptive semantic community detection (ASCD) [34], Mining interesting sub-graphs [14], and Structural fuzzy pattern analysis for attributed graphs [12] are four prevalent NMF-based approaches which perform the task of network clustering by learning the cluster preference for each vertex from edge structure and vertex features. Multi-view spectral clustering [19], [41] is able to discover network clusters by simultaneously performing spectral analysis in the matrices representing structural and feature similarities between pairwise vertices. More recently, Multiobjective genetic algorithms for attributed graph [23], [31], [39] are proposed to uncover network clusters, which performs the task by maximizing the connectivity and similarity of vertex features in clusters.

Though effective in uncovering clusters in various network data, none of the previous approaches attempts to model the mentioned latent preference between neighboring vertices. As a result, the interpretability and effectiveness of previous methods are limited as the impact of cluster allocation of vicinal vertex brought by the mentioned latent preference is always overlooked. Note that VVAMo is also different from prevalent clustering approaches considering learning adaptive neighbors [28], [42] as they neither model the vertex-wise topological/feature inclinations, nor they are particularly designed for network data. The above observations motivate us to propose a novel model to fill the void.

III. VICINAL VERTEX ALLOCATED PROBABILISTIC MATRIX FACTORIZATION

In this section, we elaborate the proposed Vicinal Vertex Allocated Matrix Factorization (VVAMo). The mathematical preliminaries and notations used throughout this paper are firstly introduced. How VVAMo formulates a network clustering problem as a unified likelihood function is then presented. At last, we derive a series of iterative algorithms for learning VVAMo.

A. Preliminaries and notations

1) *Mathematical notations:* Given a network composed of N vertices, $|E|$ edges, M vertex features, and assumed to have K clusters, we use two binary matrices $\mathbf{Y} \in \{0, 1\}^{N \times N}$ and $\mathbf{F} \in \{0, 1\}^{M \times N}$, to represent whether two vertices are connected and whether a vertex has a corresponding feature, respectively. We denote $\mathbf{Z} \in R_+^{N \times N}$ and $\mathbf{G} \in R_+^{N \times N}$ as the structural and feature correlations between connected vertices. We denote $\mathbf{V} \in R_+^{N \times K}$, each entry of which is parameterized by half-normal distribution, as the vertex-cluster preference in the network. We use $\mathbf{U} \in R_+^{M \times K}$, each column

TABLE I
NOTATIONS USED BY VVAMo

Notation	Meaning
N	Number of vertices
M	Number of features
$ E $	Number of edges
K	Number of clusters to be discovered
$\mathbf{Y}$	Vertex adjacency matrix
$\mathbf{F}$	Node-feature matrix
$\mathbf{Z}$	Structural correlation matrix
$\mathbf{G}$	Feature correlation matrix
$\mathbf{X}$	Vertex-vertex latent preference
$\mathbf{H}$	Regression parameters for vicinal vertex allocation
$\mathbf{S}$	Inclinations w.r.t. topology and vertex features
$\mathbf{V}$	Latent space representing vertex-cluster membership
$\mathbf{U}$	Latent space representing cluster themes
λ	Hyper parameters for half-normal distribution

of which contains the multi-nomial parameters, to represent the themes of the K clusters. $\mathbf{X} \in R_+^{N \times N}$, each row of which are multi-nomial parameters, is denoted as latent preference between vertices and their neighbors. It is noted that VVAMo attempts to learn the latent preference only between those connecting vertices. We denote $\mathbf{H} \in R_+^{N \times K}$ as the non-negative regression parameters used to allocate cluster labels of each vertex, according to $\mathbf{X}$. Last but not the least, we use $\mathbf{S} \in R_+^{N \times 2}$ to represent the inclinations w.r.t. topology and feature when each vertex is interacting with others. Table I summarizes the notations used in this paper.

2) *Computation of $\mathbf{Z}$ and $\mathbf{G}$:* As mentioned, $\mathbf{Z}$ and $\mathbf{G}$ represent the topological and feature correlations between pairwise vertices. Given $\mathbf{Y}$ and $\mathbf{F}$ containing binary values, there are many approaches, e.g., cosine similarity, that can be used to compute the correlations between pairwise vertices. To effectively capture the mutual relationship, in this paper, we use Normalized shifted point-wise mutual information (NSPMI) [15], [22] to respectively compute $\mathbf{Z}$ and $\mathbf{G}$:

$$\mathbf{Z}_{ij} = \begin{cases} \frac{PMI(y_i, y_j)}{H(y_i, y_j)} & \text{If } PMI(y_i, y_j) \geq \log \delta \text{ and } \mathbf{Y}_{ij} = 1 \\ 0 & \text{otherwise,} \end{cases}$$

$$PMI(y_i, y_j) = \log \frac{obs(y_i, y_j) \cdot |obs|_{\mathbf{Y}}}{obs(y_i, +) \cdot obs(y_j, +)},$$

$$H(y_i, y_j) = -\log \frac{obs(y_i, y_j)}{|obs|_{\mathbf{Y}}},$$

$$\mathbf{G}_{ij} = \begin{cases} \frac{PMI(f_i, f_j)}{H(f_i, f_j)} & \text{If } PMI(f_i, f_j) \geq \log \delta \text{ and } \mathbf{Y}_{ij} = 1 \\ 0 & \text{otherwise,} \end{cases}$$

$$PMI(f_i, f_j) = \log \frac{obs(f_i, f_j) \cdot |obs|_{\mathbf{F}}}{obs(f_i, +) \cdot obs(f_j, +)},$$

$$H(f_i, f_j) = -\log \frac{obs(f_i, f_j)}{|obs|_{\mathbf{F}}}, \quad (1)$$

where $obs(y_i, y_j)$, $obs(f_i, f_j)$, $obs(y_i, +)$, $obs(f_i, +)$, $|obs|_{\mathbf{Y}}$, and $|obs|_{\mathbf{F}}$ denote the number of vertices connecting both i th and j th vertex, the number of observations of features associated with both i th and j th vertex, the number of vertices connecting i th vertex, the number of features associated with

Fig. 2. Graphical representation of VVAMo. Shaded, black, and blank circles represent observed data, hyper parameters, and latent variables, respectively.

i th vertex, and the total number of edges or features in the network, respectively. Previous works have shown that by varying the value of δ , the obtained correlation is equivalent to the value inferred via word2vec (word embedding) [27] with a negative sampling value of δ [22]. As word embedding is a well-recognized technique used to capture the correlations between sequence of discrete values, we use NSPMI, which has shown to be equivalent to word embedding, to compute topological and feature correlations between pairwise vertices. In this paper, we typically set δ to 2 to allow VVAMo to capture more correlations.

B. Model structure

VVAMo contains four sub-components, which are responsible for feature modeling, edge modeling, latent preference modeling between vicinal vertices, and vicinal vertex allocation for cluster learning. Among them, edge and feature modeling are basic components for network clustering. What distinguishes VVAMo from other approaches is the modeling of latent preference between vicinal vertices and utilizing which to identify better cluster allocations for these proximal vertices. The graphical representation for VVAMo generating the network data is depicted in Fig. 2. The details of the model structure are elaborated as follows.

1) *Feature modeling*: Based on PLSA [16], VVAMo assumes that each feature associated with a vertex is generated by the product of $\mathbf{U}$ and $\mathbf{V}^T$:

$$p(\mathbf{F}_{mi}|\mathbf{U}, \mathbf{V}) = (\mathbf{U}\mathbf{V}^T)_{mi}^{\mathbf{F}_{mi}}. \quad (2)$$

Given Eq. (2), it is seen that VVAMo only models those features which are observed to be associated with the vertex. Such a modeling method allows VVAMo to only consider the latent distribution ($\mathbf{U}$) behind the observed features, and to avoid the negative effect brought by the noise, i.e., those non-observed features. Besides, modeling vertex features using $\mathbf{V}$ allows VVAMo to infer similar cluster preferences for those disconnected vertices which share analogous features. The cluster preference inferred by VVAMo is thereby implanted with features characterizing the vertices in the network.

2) *Edge modeling*: To model the edge, VVAMo assumes that the generation of each edge in the network following Poisson distribution. In other words, each edge is assumed to be generated by the Poisson matrix factorization parameterized by the product of $\mathbf{V}$ and $\mathbf{V}^T$:

$$p(\mathbf{Y}_{ij}|\mathbf{V}) \propto (\mathbf{V}\mathbf{V}^T)_{ij}^{\mathbf{Y}_{ij}} \cdot \exp(-\mathbf{V}\mathbf{V}^T)_{ij}. \quad (3)$$

In previous works, Poisson matrix factorization has shown to have better performance than Gaussian matrix factorization [11], [32], which encourages us to use the matrix factorization based on Poisson likelihood, rather than the Gaussian.

3) *Prior probability of $\mathbf{V}$* : Introducing a prior probability over a latent variable to allow it to meet some empirical assumption may make a probabilistic model more Bayesian [3]. As each variable in $\mathbf{V}$ should be non-negative, we assume it to follow the half-normal distribution, which is a widely-used prior distribution for non-negative latent variables:

$$p(\mathbf{V}_{ik}|\lambda_i) \propto \frac{\sqrt{2\lambda_i}}{\sqrt{\pi}} \exp(-\frac{\lambda_i \mathbf{V}_{ik}^2}{2}), \quad (4)$$

where λ_i is the hyper parameter controlling the scale of the above density function.

4) *Vicinal vertex allocation for learning clusters*: Influencing the cluster allocations of vicinal vertices, the learning of latent preference, $\mathbf{X}$ is an essential component, which distinguishes VVAMo from other related approaches. To effectively capture such latent preference, VVAMo assumes the relationship among $\mathbf{X}$, $\mathbf{S}$, $\mathbf{Z}$, and $\mathbf{G}$ follows the Boltzmann distribution. Given the mentioned latent and observed variables, the energy function of Boltzmann distribution is defined as follows:

$$\mathbf{E}_{ij} = -[\mathbf{X}_{ij} \cdot (\mathbf{S}_{iz}\mathbf{S}_{jz}\mathbf{Z}_{ij} + \mathbf{S}_{ig}\mathbf{S}_{jg}\mathbf{G}_{ij})], \quad (5)$$

where $\mathbf{S}_{iz}$ and $\mathbf{S}_{ig}$ stand for the inclinations w.r.t. topology and features when i th vertex interacts with others. Given Eq. (5), the likelihood function representing all the latent preferences is defined as:

$$p(\mathbf{X}, \mathbf{S}, \mathbf{Z}, \mathbf{G}) \propto \frac{\exp(-\sum_{i,j, \mathbf{Y}_{ij}=1} \mathbf{E}_{ij})}{A}, \quad (6)$$

$$A = \sum_{i,j, \mathbf{Y}_{ij}=1} \exp(-\mathbf{E}_{ij}).$$

Given Eqs. (5) and (6), it is seen that the multi-nomial parameter $\mathbf{X}_{ij}$ is inferred according to heterogeneous correlations ($\mathbf{Z}$ and $\mathbf{G}$) and corresponding inclinations ($\mathbf{S}$), not a simple mean of $\mathbf{Z}_{ij}$ and $\mathbf{G}_{ij}$. The likelihood function in Eq. (6) increases when those pairs of vertices sharing similar inclinations and relatively high correlations are assigned with higher $\mathbf{X}_{ij}$ s. Optimizing Eq. (6) therefore allows VVAMo to quantitatively rank the neighbors for each vertex in the network, concerning heterogeneous correlations and corresponding inclinations. Those pairs of vertices having higher rankings, i.e., $\mathbf{X}_{ij}$ s, can be seen as truly vicinal vertices inferred by VVAMo.

Intuitively, vicinal vertices are highly possible to allocate into the same cluster. Given $\mathbf{X}$, VVAMo achieves the allocation of vicinal vertex for cluster learning via assuming that the relationship between cluster preference ($\mathbf{V}$) and the regressing latent preference also follows Boltzmann distribution. Thus, the corresponding energy function can be defined as:

$$\mathbf{R}_{ik} = -[(\mathbf{X}\mathbf{H})_{ik} \cdot \mathbf{V}_{ik}]. \quad (7)$$

Given Eq. (7), the Boltzmann distribution interpreting the relationship between the regressing latent preference and all cluster labels is defined as:

$$p(\mathbf{X}, \mathbf{H}, \mathbf{V}) \propto \frac{\exp(-\sum_{i,k} \mathbf{R}_{ik})}{B}, B = \sum_{i,k} \exp(-\mathbf{R}_{ik}). \quad (8)$$

As seen, the likelihood function in Eq. (8) increases when a high regressing latent preference of a vertex $((\mathbf{X}\mathbf{H})_{ik})$ meets a high corresponding cluster preference $(\mathbf{V}_{ik})$. According to Eqs. (7) and (8), it is also known that the regressing latent preferences for a cluster obtained by different vertices can be similarly large if their mutual latent preferences are relatively high. Optimizing Eq. (8) can thereby regularize VVAMo to allocate those vertices having high mutual preferences into the same cluster.

Appropriately integrating the learning of $\mathbf{X}$ with the inference of cluster preference, the modeling of vicinal vertex allocation is another critical module distinguishing VVAMo from other approaches. The learning of $\mathbf{X}$ enables VVAMo to effectively identify which neighbors are more latently preferred by each vertex in the network. Then, through Eq. (8), the regression of such latent preference can regularize the proposed model to assign similar cluster label to those vertices which are highly correlated. Thus, clusters that are interpretable from both structural and feature aspects can be learned by VVAMo. Note that, given the flexibility of the modeling method shown in Eqs. (5) - (8), one may immediately allow VVAMo to capture the latent features between higher-order neighbors, e.g., 2 or 3-hop neighbors by using a proper method to compute $\mathbf{Z}$ and $\mathbf{G}$.

5) *The joint likelihood function:* Based on the above descriptions, the joint likelihood function utilized by VVAMo to discover clusters in the network can be defined as follows:

$$p(\mathbf{Y}, \mathbf{F}, \mathbf{V}, \mathbf{U}, \mathbf{H}, \mathbf{X}, \mathbf{S}|\lambda) = p(\mathbf{V}|\lambda) \cdot \prod_{i \neq j} p(\mathbf{Y}_{ij}|\mathbf{V}) \cdot \prod_{i,m} p(\mathbf{F}_{mi}|\mathbf{U}, \mathbf{V}) \cdot p(\mathbf{X}, \mathbf{S}, \mathbf{Z}, \mathbf{G}) \cdot p(\mathbf{X}, \mathbf{H}, \mathbf{V}). \quad (9)$$

Taking the logarithm of Eq. (9), we may obtain the following unified log-likelihood function for VVAMo to learn optimal cluster membership for each vertex in the network:

$$L = \sum_{i \neq j} \mathbf{Y}_{ij} \log(\mathbf{V}\mathbf{V}^T)_{ij} - \sum_{i \neq j} (\mathbf{V}\mathbf{V})_{ij}^T + \sum_{i,m} \mathbf{F}_{mi} \log(\mathbf{U}\mathbf{V}^T)_{mi} - \sum_{i,j, \mathbf{Y}_{ij}=1} \mathbf{E}_{ij} - \sum_{i,k} \mathbf{R}_{ik} - \log A - \log B - \sum_{i,k} \frac{\lambda_i}{2} \mathbf{V}_{ik}^2 + \frac{K}{2} \sum_i \log \lambda_i + \text{const}, \quad (10)$$

where *const* represents the terms that are irrelevant to model optimization. Given Eq. (10), we may observe that VVAMo is a model of probabilistic matrix factorization and is fundamentally different from previous methods for network clustering. Besides considering modeling edges and vertex features in the network, VVAMo further attempts to learn the optimal latent preference, according to the correlations of topology and features, and corresponding inclinations between vicinal vertices. Simultaneously, such latent preference can regularize VVAMo

to allocate those vicinal vertices sharing higher correlations, and similar inclinations into the same group. Interpretable clusters implanted with latent information on the proximal vertices can therefore be uncovered by the proposed model.

C. Learning VVAMo

Given the structure of Eq. (10), it is known that directly optimizing VVAMo is intractable. Fortunately, several frameworks, e.g., gradient descent [2], Expectation-Maximization (EM) [9], and variational inference [7], have been proposed to optimize such a complex likelihood function. In this paper, we derive a series of algorithms for learning VVAMo based on Maximum a posteriori (MAP). MAP attempts to optimize the log-likelihood function via iteratively maximizing the posterior probability of each latent variable. Within a finite number of iterations, VVAMo is able to achieve the local optima by performing MAP.

1) *Learning V:* Based on the MAP framework, updating $\mathbf{V}$ is equivalent to maximize the corresponding Lagrangian of the log posterior probability of $\mathbf{V}$:

$$\begin{aligned} \mathbf{V} &= \arg \max_{\mathbf{V}} [\log p(\mathbf{V}|\mathbf{Y}, \mathbf{F}, \mathbf{U}, \mathbf{H}, \mathbf{X}, \mathbf{S}, \lambda)] \\ &\propto \arg \max_{\mathbf{V}} [\log p(\mathbf{V}|\lambda) + \log p(\mathbf{Y}|\mathbf{V}) + \log p(\mathbf{X}, \mathbf{H}, \mathbf{V})] \\ &= \arg \max_{\mathbf{V}} [\sum_{i \neq j} \mathbf{Y}_{ij} \log(\mathbf{V}\mathbf{V}^T)_{ij} - \sum_{i \neq j} (\mathbf{V}\mathbf{V})_{ij}^T \\ &\quad + \sum_{i,m} \mathbf{F}_{mi} \log(\mathbf{U}\mathbf{V}^T)_{mi} - \sum_{i,k} \mathbf{R}_{ik} \\ &\quad - \log B - \sum_{i,k} \frac{\lambda_i}{2} \mathbf{V}_{ik}^2 - \sum_{i,k} \Gamma_{ik} \mathbf{V}_{ik}], \end{aligned} \quad (11)$$

where Γ_{ik} represents the Lagrangian multiplier for the non-negative constraint for each $\mathbf{V}_{ik}$. Based on the KKT conditions for constrained optimization, we may derive the following element-wise equation system:

$$\begin{aligned} \frac{\partial \log p(\cdot)}{\partial \mathbf{V}_{ik}} &= 2 \sum_j \frac{\mathbf{Y}_{ij}}{(\mathbf{V}\mathbf{V}^T)_{ij}} \mathbf{V}_{jk} + \sum_m \frac{\mathbf{F}_{mi}}{(\mathbf{U}\mathbf{V}^T)_{mi}} \mathbf{U}_{mk} \\ &\quad + (\mathbf{X}\mathbf{H})_{ik} - \frac{(\mathbf{X}\mathbf{H})_{ik} \exp(-\mathbf{R}_{ik})}{B} - 2 \sum_j \mathbf{V}_{jk} \\ &\quad - \lambda_i \mathbf{V}_{ik} - \Gamma_{ik} = 0, \Gamma_{ik} \cdot \mathbf{V}_{ik} = 0, \mathbf{V}_{ik} \geq 0, \Gamma_{ik} \geq 0. \end{aligned} \quad (12)$$

Solving the first equation in terms of Γ_{ik} in the above system and substituting Γ_{ik} in the second equation with the solution, we have:

$$\begin{aligned} \mathbf{V}_{ik} \cdot [2 \sum_j \frac{\mathbf{Y}_{ij}}{(\mathbf{V}\mathbf{V}^T)_{ij}} \mathbf{V}_{jk} + \sum_m \frac{\mathbf{F}_{mi}}{(\mathbf{U}\mathbf{V}^T)_{mi}} \mathbf{U}_{mk} + (\mathbf{X}\mathbf{H})_{ik} \\ - \frac{(\mathbf{X}\mathbf{H})_{ik} \exp(-\mathbf{R}_{ik})}{B} - 2 \sum_j \mathbf{V}_{jk} - \lambda_i \mathbf{V}_{ik}] = 0. \end{aligned} \quad (13)$$

Solving the above equation, we are able to derive the updating rule for $\mathbf{V}_{ik}$:

$$\mathbf{V}_{ik} \leftarrow \mathbf{V}_{ik} \cdot \frac{2(\frac{\mathbf{Y}}{\mathbf{V}\mathbf{V}^T} \mathbf{V})_{ik} + ((\frac{\mathbf{F}}{\mathbf{U}\mathbf{V}^T})^T \mathbf{U})_{ik} + (\mathbf{X}\mathbf{H})_{ik}}{(\frac{\mathbf{X}\mathbf{H}}{\mathbf{B}})_{ik} \exp(-\mathbf{R}_{ik}) + 2 \sum_j \mathbf{V}_{jk} + \lambda_i \mathbf{V}_{ik}}, \quad (14)$$

where $(\frac{\mathbf{Y}}{\mathbf{V}\mathbf{V}^T} \mathbf{V})_{ik}$, and $((\frac{\mathbf{F}}{\mathbf{U}\mathbf{V}^T})^T \mathbf{U})_{ik}$ stand for $\sum_j \frac{\mathbf{Y}_{ij}}{(\mathbf{V}\mathbf{V}^T)_{ij}} \mathbf{V}_{jk}$, and $\sum_m \frac{\mathbf{F}_{mi}}{(\mathbf{U}\mathbf{V}^T)_{mi}} \mathbf{U}_{mk}$, respectively.

2) *Learning U*: To derive the updating rule for $\mathbf{U}$, we may maximize the log posterior probability of $\mathbf{U}$:

$$\begin{aligned} \mathbf{U} &= \arg \max_{\mathbf{U}} [\log p(\mathbf{U}|\mathbf{F}, \mathbf{V})] \propto \arg \max_{\mathbf{U}} [\log p(\mathbf{F}|\mathbf{U}, \mathbf{V})] \\ &= \arg \max_{\mathbf{U}} \left[\sum_{i,m} \mathbf{F}_{mi} \log(\mathbf{U}\mathbf{V}^T)_{mi} \right. \\ &\quad \left. - \sum_k \nu_k \left(\sum_m \mathbf{U}_{mk} - 1 \right) \right], \end{aligned} \quad (15)$$

where ν_k represents the Lagrange multiplier for the unity constraint of each column of $\mathbf{U}$. Based on the KKT conditions for constrained optimization, we may derive the following equation system:

$$\begin{aligned} \frac{\partial \log p(\cdot)}{\partial \mathbf{U}_{mk}} &= \sum_i \frac{\mathbf{F}_{mi}}{(\mathbf{U}\mathbf{V}^T)_{mi}} \mathbf{V}_{ik} - \nu_k = 0, \\ \sum_m \mathbf{U}_{mk} - 1 &= 0. \end{aligned} \quad (16)$$

The first equation in the above system can be rewritten as:

$$\sum_m \mathbf{U}_{mk} \cdot \sum_i \frac{\mathbf{F}_{mi}}{(\mathbf{U}\mathbf{V}^T)_{mi}} \mathbf{V}_{ik} - \nu_k \sum_m \mathbf{U}_{mk} = 0. \quad (17)$$

As the sum of each column of $\mathbf{U}$ is equal to one, we are able to obtain the solution of ν_k and then derive the updating rule for each variable in $\mathbf{U}$:

$$\mathbf{U}_{mk} \leftarrow \mathbf{U}_{mk} \cdot \frac{(\frac{\mathbf{F}}{\mathbf{U}\mathbf{V}^T} \mathbf{V})_{mk}}{\sum_m \mathbf{U}_{mk} \cdot (\frac{\mathbf{F}}{\mathbf{U}\mathbf{V}^T} \mathbf{V})_{mk}}, \quad (18)$$

where $(\frac{\mathbf{F}}{\mathbf{U}\mathbf{V}^T} \mathbf{V})_{mk}$ represents $\sum_i \frac{\mathbf{F}_{mi}}{(\mathbf{U}\mathbf{V}^T)_{mi}} \mathbf{V}_{ik}$.

3) *Learning other model parameters*: Similar to how to deduce the updating rules for $\mathbf{V}_{ik}$ and $\mathbf{U}_{mk}$, we may obtain

the rules for updating $\mathbf{X}_{ij}$, $\mathbf{H}_{jk}$, $\mathbf{S}_{id}$, and λ_i . The updating rule for $\mathbf{X}_{ij}$ is:

$$\begin{aligned} \mathbf{X}_{ij} &\leftarrow \frac{\Delta_{ij} \sum_j \frac{\mathbf{X}_{ij}}{\Lambda_{ij}} + \mathbf{X}_{ij}}{\Lambda_{ij} \sum_j \frac{\mathbf{X}_{ij}}{\Lambda_{ij}} + \sum_j \frac{\mathbf{X}_{ij} \Delta_{ij}}{\Lambda_{ij}}}, \\ \Delta_{ij} &= [(\mathbf{V}\mathbf{H}^T)_{ij} + \sigma_{ij}] \mathbf{X}_{ij}, \\ \Lambda_{ij} &= \left(\left(\frac{\exp(-\mathbf{R}_{ik})}{B} \cdot \mathbf{V} \right) \mathbf{H}^T \right)_{ij} + \sigma_{ij} \cdot \frac{\exp(-\mathbf{E}_{ij})}{A}, \\ \sigma_{ij} &= \mathbf{S}_{iz} \mathbf{S}_{jz} \mathbf{Z}_{ij} + \mathbf{S}_{ig} \mathbf{S}_{jg} \mathbf{G}_{ij}, \end{aligned} \quad (19)$$

where $((\frac{\exp(-\mathbf{R}_{ik})}{B} \cdot \mathbf{V}) \mathbf{H}^T)_{ij}$ stands for $\sum_k \frac{\exp(-\mathbf{R}_{ik})}{B} \cdot \mathbf{V}_{ik} \cdot \mathbf{H}_{jk}$.

The updating rule for $\mathbf{H}_{jk}$ is:

$$\mathbf{H}_{jk} \leftarrow \mathbf{H}_{jk} \cdot \frac{(\mathbf{X}^T \mathbf{V})_{jk}}{(\mathbf{X}^T (\frac{\exp(-\mathbf{R}_{ik})}{B} \cdot \mathbf{V}))_{jk}}, \quad (20)$$

where $(\mathbf{X}^T (\frac{\exp(-\mathbf{R}_{ik})}{B} \cdot \mathbf{V}))_{jk}$ represents $\sum_i \mathbf{X}_{ij} \frac{\exp(-\mathbf{R}_{ik})}{B} \cdot \mathbf{V}_{ik}$.

The updating rule for $\mathbf{S}_{iz}$ and $\mathbf{S}_{ig}$ is:

$$\begin{aligned} \mathbf{S}_{iz} &\leftarrow \frac{\mathbf{S}_{iz} (\frac{\mathbf{S}_{iz}}{(\Psi\mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi\mathbf{S})_{ig}}) (\Delta\mathbf{S})_{iz} + \mathbf{S}_{iz}}{(\Psi\mathbf{S})_{iz} (\frac{\mathbf{S}_{iz}}{(\Psi\mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi\mathbf{S})_{ig}}) + (\frac{\mathbf{S}_{iz}(\Delta\mathbf{S})_{iz}}{(\Psi\mathbf{S})_{iz}} + \frac{\mathbf{S}_{iz}(\Lambda\mathbf{S})_{ig}}{(\Phi\mathbf{S})_{ig}})}, \\ \mathbf{S}_{ig} &\leftarrow \frac{\mathbf{S}_{ig} (\frac{\mathbf{S}_{iz}}{(\Psi\mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi\mathbf{S})_{ig}}) (\Lambda\mathbf{S})_{ig} + \mathbf{S}_{ig}}{(\Phi\mathbf{S})_{ig} (\frac{\mathbf{S}_{iz}}{(\Psi\mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi\mathbf{S})_{ig}}) + (\frac{\mathbf{S}_{iz}(\Delta\mathbf{S})_{iz}}{(\Psi\mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}(\Lambda\mathbf{S})_{ig}}{(\Phi\mathbf{S})_{ig}})}, \end{aligned} \quad (21)$$

$$\Delta_{ij} = \mathbf{X}_{ij} \mathbf{Z}_{ij},$$

$$\Lambda_{ij} = \mathbf{X}_{ij} \mathbf{G}_{ij},$$

$$\Psi_{ij} = \mathbf{X}_{ij} \mathbf{Z}_{ij} \frac{\exp\{-\mathbf{E}_{ij}\}}{A},$$

$$\Phi_{ij} = \mathbf{X}_{ij} \mathbf{G}_{ij} \frac{\exp\{-\mathbf{E}_{ij}\}}{A}.$$

Finally, the updating rule for λ_i is:

$$\lambda_i = \frac{K}{\sum_k \mathbf{V}_{ik}^2}. \quad (22)$$

By iteratively updating the variables and parameters according to the above rules, VVAMo will converge to local optima in a finite number of iterations. The process of learning VVAMo has been summarized in Algorithm 1.

IV. ALGORITHMIC ANALYSIS OF VVAMo

In this section, we analyze the algorithmic properties of the proposed method, including the algorithm convergence and the computational complexity.

A. Convergence analysis

To prove the convergence of the proposed model, we utilize the idea for proving the convergence of the EM algorithm. That is, before each iteration of updating, we first derive the lower bound of the log-likelihood in terms of all the latent variables. Then, we show that maximizing the derived lower bound is equivalent to update all the model parameters according to the rules derived in the previous section. As the maximization of the lower bound in each iteration is non-decreasing, the convergence of the derived algorithm for learning the variables in VVAMo is guaranteed.

Algorithm 1: Vicinal Vertex Allocated Matrix Factorization (VVAMo)

Input: Network Data: $\mathbf{Y}$, $\mathbf{F}$, $\mathbf{Z}$, $\mathbf{G}$

Output: Cluster preference for each vertex $\{\mathbf{V}_i\}_{i=1}^N$;
Cluster themes $\{\mathbf{U}_k\}_{k=1}^K$; Regression variables $\{\mathbf{H}_k\}_{k=1}^K$; Latent preference $\{\mathbf{X}_{ij}\}_{i \neq j, \mathbf{Y}_{ij}=1}^N$; Vertex-wise topology/feature inclination $\{\mathbf{S}_i\}_{i=1}^N$

```

1 Initialize  $\mathbf{U}$ ,  $\mathbf{V}$ ,  $\mathbf{X}$ ,  $\mathbf{S}$ ,  $\mathbf{H}$ ,  $\lambda_i$ ;
2  $t \leftarrow 0$ ;
3 while  $t < T_{max}$  do
4    $t \leftarrow t + 1$ ;
5   Updating  $\mathbf{V}_{ik}$ ,  $\mathbf{U}_{mk}$ ,  $\mathbf{X}_{ij}$ ,  $\mathbf{H}_{jk}$ ,  $\mathbf{S}_i$ , and  $\lambda_i$  by
     Eqs. (14), (18), (19)-(22);
6   Compute Log likelihood  $L^{(t)}$  by Eq. (10);
7   if  $L^{(t)} - L^{(t-1)} \leq \epsilon$  then
8     break;
9   end
10 end
11 Identify cluster label for each vertex using  $\mathbf{V}$ .
```

1) *Lower bound of the log-likelihood:* To lower bound Eq. (10), we may utilize the following two properties of logarithmic functions.

$$\begin{aligned} \log\left(\sum_k x_k\right) &\geq \sum_k a_k \log\left(\frac{x_k}{a_k}\right), \\ \text{if } \sum_k a_k &= 1, x_k > 0; \\ \sum_k x_k \log \frac{x_k}{a_k} &\geq x \log \frac{x}{a}, \\ \text{if } a_k, x_k > 0, x &= \sum_k x_k, a = \sum_k a_k. \end{aligned} \quad (23)$$

Based on the first property shown in Eq. (23), the terms related to edge and feature modeling in Eq. (10) can be lower bounded:

$$\begin{aligned} L &\geq \sum_{i \neq j} \mathbf{Y}_{ij} \sum_k \theta_{ij,k} \log \frac{\mathbf{V}_{ik} \mathbf{V}_{jk}}{\theta_{ij,k}} - \sum_{i \neq j} (\mathbf{V} \mathbf{V})_{ij}^T \\ &+ \sum_{i,m} \mathbf{F}_{mi} \sum_k \phi_{mi,k} \log \frac{\mathbf{U}_{mk} \mathbf{V}_{ik}}{\phi_{mi,k}} - \sum_{i,j} \mathbf{E}_{ij} - \sum_{i,k} \mathbf{R}_{ik} \\ &- \log A - \log B - \sum_{i,k} \frac{\lambda_i}{2} \mathbf{V}_{ik}^2 + \frac{K}{2} \sum_i \log \lambda_i + \text{const}, \\ \theta_{ij,k} &= \frac{\mathbf{V}_{ik} \mathbf{V}_{jk}}{(\mathbf{V} \mathbf{V})_{ij}^T}, \phi_{mi,k} = \frac{\mathbf{U}_{mk} \mathbf{V}_{ik}}{(\mathbf{U} \mathbf{V})_{mi}^T}, \end{aligned} \quad (24)$$

where $\theta_{ij,k}$ and $\phi_{mi,k}$ are auxiliary variables for ensuring the lower bound related to the terms of edge and feature modeling. Based on the second property shown in Eq. (23), the terms related to log-sum, i.e., $\log A$ and $\log B$ in Eq. (10), can be lower bounded as follows:

$$\begin{aligned} L &\geq Q = \sum_{i \neq j} \mathbf{Y}_{ij} \sum_k \theta_{ij,k} \log \frac{\mathbf{V}_{ik} \mathbf{V}_{jk}}{\theta_{ij,k}} - \sum_{i \neq j} (\mathbf{V} \mathbf{V})_{ij}^T \\ &- \sum_{i,k} \mathbf{R}_{ik} + \sum_{i,k} \eta_{ik} \mathbf{R}_{ik} \\ &+ \sum_{i,m} \mathbf{F}_{mi} \sum_k \phi_{mi,k} \log \frac{\mathbf{U}_{mk} \mathbf{V}_{ik}}{\phi_{mi,k}} - \sum_{i,j} \mathbf{E}_{ij} \\ &+ \sum_{i,j} \delta_{ij} \mathbf{E}_{ij} - \sum_{i,k} \frac{\lambda_i}{2} \mathbf{V}_{ik}^2 + \frac{K}{2} \sum_i \log \lambda_i + \text{const}, \\ \eta_{ik} &= \frac{\exp(-\mathbf{R}_{ik})}{B}, \delta_{ij} = \frac{\exp(-\mathbf{E}_{ij})}{A}, \end{aligned} \quad (25)$$

where η_{ik} and δ_{ij} are the corresponding auxiliary variables for ensuring the lower bound related to the log-sum. As Eq. (10) is no lower than the presented Q (Eq. (25)), pushing up Q is equivalent to maximize the log-likelihood function.

2) *Convergence related to $\mathbf{V}$:* Showing the model convergence related to $\mathbf{V}$ is equivalent to show that the rule for maximizing the corresponding lower bound is the same as the updating rule for learning $\mathbf{V}$. Thus, we have:

$$\begin{aligned} \mathbf{V}^{(new)} &= \arg \max_{\mathbf{V}} [Q(\mathbf{V})] = \arg \max_{\mathbf{V}} \left[- \sum_{i \neq j} (\mathbf{V} \mathbf{V})_{ij}^T \right. \\ &+ \sum_{i \neq j} \mathbf{Y}_{ij} \sum_k \theta_{ij,k} \log \frac{\mathbf{V}_{ik} \mathbf{V}_{jk}}{\theta_{ij,k}} - \sum_{i,k} \mathbf{R}_{ik} + \sum_{i,k} \eta_{ik} \mathbf{R}_{ik} \\ &\left. + \sum_{i,m} \mathbf{F}_{mi} \sum_k \phi_{mi,k} \log \frac{\mathbf{U}_{mk} \mathbf{V}_{ik}}{\phi_{mi,k}} - \sum_{i,k} \frac{\lambda_i}{2} \mathbf{V}_{ik}^2 \right]. \end{aligned} \quad (26)$$

Taking the partial derivative of $Q(\mathbf{V})$ w.r.t. $\mathbf{V}_{ik}$ and setting it to zero, we have:

$$\begin{aligned} \frac{\partial Q(\mathbf{V})}{\partial \mathbf{V}_{ik}} &= 2 \sum_j \mathbf{Y}_{ij} \theta_{ij,k} - 2 \mathbf{V}_{ik} \sum_j \mathbf{V}_{jk} + \sum_m \mathbf{F}_{mi} \phi_{mi,k} \\ &+ \mathbf{V}_{ik} (\mathbf{X} \mathbf{H})_{ik} - \mathbf{V}_{ik} \eta_{ik} (\mathbf{X} \mathbf{H})_{ik} - \lambda_i \mathbf{V}_{ik}^2 = 0 \Rightarrow \\ \mathbf{V}_{ik}^{(new)} &\leftarrow \frac{2 \sum_j \mathbf{Y}_{ij} \theta_{ij,k} + \sum_m \mathbf{F}_{mi} \phi_{mi,k} + \mathbf{V}_{ik} (\mathbf{X} \mathbf{H})_{ik}}{2 \sum_j \mathbf{V}_{jk} + \eta_{ik} (\mathbf{X} \mathbf{H})_{ik} + \lambda_i \mathbf{V}_{ik}}. \end{aligned} \quad (27)$$

Substituting $\theta_{ij,k}$, $\phi_{mi,k}$, and η_{ik} in Eq. (27), we may observe that the rule for maximizing Q w.r.t. each $\mathbf{V}_{ik}$ is equivalent to what is depicted in Eq. (14). Thus, Eq. (27) shows that updating $\mathbf{V}$ according to Eq. (14) may increase the log-likelihood till convergence.

3) *Convergence related to other variables:* Similarly, we may derive the rules for maximizing Q w.r.t. other latent variables. For pushing up the lower bound w.r.t. $\mathbf{U}_{mk}$, we have:

$$\begin{aligned} \mathbf{U}_{mk}^{(new)} &= \arg \max_{\mathbf{U}_{mk}} [Q(\mathbf{U}_{mk})] \\ &= \arg \max_{\mathbf{U}_{mk}} \left[\sum_i \mathbf{F}_{mi} \sum_k \phi_{mi,k} \log \frac{\mathbf{U}_{mk} \mathbf{V}_{ik}}{\phi_{mi,k}} \right. \\ &\left. - \nu_k (\sum_m \mathbf{U}_{mk} - 1) \right] \Rightarrow \mathbf{U}_{mk}^{(new)} \leftarrow \frac{\sum_i \mathbf{F}_{mi} \phi_{mi,k}}{\sum_{i,m} \mathbf{F}_{mi} \phi_{mi,k}}. \end{aligned} \quad (28)$$

Eq. (28) is equivalent to Eq. (18) if $\phi_{mi,k}$ is substituted as it is shown in Eq. (24).

The rule for pushing up Q w.r.t. $\mathbf{H}_{jk}$ can be derived as the following:

$$\begin{aligned} \mathbf{H}_{jk}^{(new)} &= \arg \max_{\mathbf{H}_{jk}} [Q(\mathbf{H}_{jk})] \\ &= \arg \max_{\mathbf{H}_{jk}} \left[- \sum_{i,k} \mathbf{R}_{ik} + \sum_{i,k} \eta_{ik} \mathbf{R}_{ik} \right] \\ &\Rightarrow \mathbf{H}_{jk}^{(new)} \leftarrow \mathbf{H}_{jk} \cdot \frac{\sum_i \mathbf{X}_{ij} \mathbf{V}_{ik}}{\sum_i \eta_{ik} \cdot \mathbf{V}_{ik} \cdot \mathbf{X}_{ij}}. \end{aligned} \quad (29)$$

To maximize the presented lower bound related to the latent preference, $\mathbf{X}_{ij}$, we derive the following updating rule:

$$\begin{aligned} \mathbf{X}_{ij}^{(new)} &= \arg \max_{\mathbf{X}_{ij}} [Q(\mathbf{X}_{ij})] = \arg \max_{\mathbf{X}_{ij}} \left[- \sum_{i,j} \mathbf{E}_{ij} \right. \\ &- \sum_{i,k} \mathbf{R}_{ik} + \sum_{i,j} \delta_{ij} \mathbf{E}_{ij} + \sum_{i,k} \eta_{ik} \mathbf{R}_{ik} + \nu_i (\sum_j \mathbf{X}_{ij} - 1) \left. \right] \\ &\Rightarrow \mathbf{X}_{ij}^{(new)} \leftarrow \frac{\Delta_{ij} \sum_j \frac{\mathbf{X}_{ij}}{\Lambda_{ij}} + \mathbf{X}_{ij}}{\Lambda_{ij} \sum_j \frac{\mathbf{X}_{ij}}{\Lambda_{ij}} + \sum_j \frac{\mathbf{X}_{ij} \Delta_{ij}}{\Lambda_{ij}}}, \end{aligned} \quad (30)$$

$$\begin{aligned} \Delta_{ij} &= [(\mathbf{V} \mathbf{H}^T)_{ij} + \sigma_{ij}] \mathbf{X}_{ij}, \\ \sigma_{ij} &= \mathbf{S}_{iz} \mathbf{S}_{jz} \mathbf{Z}_{ij} + \mathbf{S}_{ig} \mathbf{S}_{jg} \mathbf{G}_{ij}, \\ \Lambda_{ij} &= \sum_k \eta_{ik} \cdot \mathbf{V}_{ik} \cdot \mathbf{H}_{jk} + \sigma_{ij} \cdot \delta_{ij}. \end{aligned}$$

Finally, the rule for pushing up the lower bound related to the inclinations w.r.t. topology and vertex features ($\mathbf{S}_{id}$,

$d \in \{z, g\}$) can be derived as follows:

$$\begin{aligned}
\mathbf{S}_{iz}^{(new)} &= \arg \max_{\mathbf{S}_{iz}} [Q(\mathbf{S}_{iz})] \\
&= \arg \max_{\mathbf{S}_{iz}} \left[- \sum_{i,j} \mathbf{E}_{ij} + \sum_{i,j} \delta_{ij} \mathbf{E}_{ij} - \nu_i (\mathbf{S}_{iz} + \mathbf{S}_{ig} - 1) \right] \\
&\Rightarrow \mathbf{S}_{iz}^{(new)} \leftarrow \frac{\mathbf{S}_{iz} \left(\frac{\mathbf{S}_{iz}}{(\Psi \mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi \mathbf{S})_{ig}} \right) (\Delta \mathbf{S})_{iz} + \mathbf{S}_{iz}}{(\Psi \mathbf{S})_{iz} \left(\frac{\mathbf{S}_{iz}}{(\Psi \mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi \mathbf{S})_{ig}} \right) + \left(\frac{\mathbf{S}_{iz} (\Delta \mathbf{S})_{iz}}{(\Psi \mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig} (\Delta \mathbf{S})_{ig}}{(\Phi \mathbf{S})_{ig}} \right)}, \\
\mathbf{S}_{ig}^{(new)} &= \arg \max_{\mathbf{S}_{ig}} [Q(\mathbf{S}_{ig})] \\
&= \arg \max_{\mathbf{S}_{ig}} \left[- \sum_{i,j} \mathbf{E}_{ij} + \sum_{i,j} \delta_{ij} \mathbf{E}_{ij} - \nu_i (\mathbf{S}_{iz} + \mathbf{S}_{ig} - 1) \right] \\
&\Rightarrow \mathbf{S}_{ig}^{(new)} \leftarrow \frac{\mathbf{S}_{ig} \left(\frac{\mathbf{S}_{iz}}{(\Psi \mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi \mathbf{S})_{ig}} \right) (\Delta \mathbf{S})_{ig} + \mathbf{S}_{ig}}{(\Phi \mathbf{S})_{ig} \left(\frac{\mathbf{S}_{iz}}{(\Psi \mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig}}{(\Phi \mathbf{S})_{ig}} \right) + \left(\frac{\mathbf{S}_{iz} (\Delta \mathbf{S})_{iz}}{(\Psi \mathbf{S})_{iz}} + \frac{\mathbf{S}_{ig} (\Delta \mathbf{S})_{ig}}{(\Phi \mathbf{S})_{ig}} \right)}, \\
\Delta_{ij} &= \mathbf{X}_{ij} \mathbf{Z}_{ij}, \Lambda_{ij} = \mathbf{X}_{ij} \mathbf{G}_{ij}, \\
\Psi_{ij} &= \mathbf{X}_{ij} \mathbf{Z}_{ij} \cdot \delta_{ij}, \Phi_{ij} = \mathbf{X}_{ij} \mathbf{G}_{ij} \cdot \delta_{ij}.
\end{aligned} \tag{31}$$

It is easy to show that the above rules for maximizing the lower bound are equivalent to what are used for updating the corresponding latent variables, when the auxiliary variables are substituted according to Eqs. (24) and (25). Since Eq. (10) is no lower than Q and updating Q according to the above rules is non-decreasing, the convergence of the derived learning algorithms is theoretically verified.

B. Computational complexity

The model complexity can be analyzed according to the updating rules shown in Eq. (14)-(22). In each iteration, updating all the latent variables in $\mathbf{V}$ follows the order of $O(2N^2K + NMK + 2NK^2 + eNK)$, where e represents the average degree of each vertex. Updating all the latent variables in $\mathbf{U}$ follows the order of $O(NMK + M^2K + 2MK^2)$. Updating all the latent variables in $\mathbf{X}$ and $\mathbf{S}$ follows the order of $O(3|E|e + |E|K)$ and $O(4Ne)$, respectively. And updating the hyper-parameter λ follows the order of $O(NK)$. Therefore, learning all the latent variables and model parameters approximately follows the order of $O(N^2K + NMK + M^2K + |E|e)$.

V. EXPERIMENTS AND ANALYSIS

In this section, we conduct a series of experiments on real-world datasets, including social network, document network, and biological network to validate the effectiveness of VVAMo against both classical and state-of-the-art methods.

A. Experimental setup

1) *Baselines for comparison:* Ten approaches, which are either classical or state-of-the-art, are selected as baselines. They can be categorized into two classes. Ncut [35], SBM [30], CoDA [47], DANMF [48], and DNMF [49] are five effective methods utilizing network topology to uncover clusters. MVSC [19], CESNA [46], VGAE [18], SCI [44], and ASCD [34] are state-of-the-art approaches to network clustering, which utilize both network structure and vertex features to unfold clusters in the network.

TABLE II
STATISTICS OF DATASETS USED IN THE EXPERIMENTS. SOC, DOC, OR BIO REPRESENTS WHETHER THE DATASET IS A SOCIAL, DOCUMENT, OR BIOLOGICAL NETWORK.

Dataset	Type	N	$ E $	M	K
Twitter	Soc	3687	49881	20905	242
Ego	Soc	4039	88234	1283	191
Flickr	Soc	7575	239738	12047	9
Gplus	Soc	8725	972899	5913	130
Blog	Doc	5196	171743	8189	6
UAI	Doc	3363	33300	4971	19
Wiki	Doc	2405	17981	4973	17
Biogrid	Bio	5640	59748	4286	200

NCut is a spectral-based method for network clustering, which performs the task via assigning vertices sharing higher structural similarity into the same cluster. SBM is a Bayesian model for network clustering. It can uncover the community structures in graphs taking advantage of a multi-stage maximum likelihood approach. CoDA is an effective method based on probabilistic matrix factorization, which can identify clusters via learning a low-dimensional space from network topology. DANMF is a deep NMF-based approach, which uses deep encoder and decoder to learn the hierarchical mappings between the original network and the final community membership space. DNMF is also an NMF based approach, which seeks for a discrete community membership matrix directly and incorporates a pseudo supervision module to exploit the discriminative information in an unsupervised manner.

MVSC is a prevalent spectral clustering framework that finds consistent clusters from the structure and feature similarity between pairwise vertices, through the co-regularization of clustering hypotheses across heterogeneous similarity spaces. CESNA is a probabilistic generative model, which learns a shared latent space as cluster preference for each vertex from edge structure and vertex features. VGAE is a prevalent graph convolution network (GCN) for discovering latent modules in the network data. It defines a multi-layer variational encoder to learn graph representations from normalized graph adjacency and node features. The cluster preference for each vertex can be inferred via minimizing the divergence between graph adjacency and that constructed by the graph representations. SCI is an effective NMF-based approach to network clustering. It performs the task by learning the community membership for each vertex from edge structure and discriminated features. ASCD integrates topological structure and node attributes of social networks under the framework of NMF and introduces an adaptive parameter to control the trade-off between topology and content according to the identified mismatch degree.

In our experiments, we used the source codes of all the baselines provided by the authors for implementation and configured the baselines by using either default settings or recommended ones. Specifically, CESNA does not need any predefined parameter. For the rest of the baselines, including NCut, SBM, CoDA, DANMF, DNMF, MVSC, VGAE, SCI, and ASCD, we used the settings recommended by the authors.

For the number of clusters, i.e., K , which has to be determined by the proposed model, and all the baselines except CESNA, we set it to be equal to the number of ground-truth clusters of the testing dataset. All of the experiments were performed on a workstation with 6-core 3.4GHz CPU and 32GB RAM and all approaches were executed 10 times to obtain a statistically steady performance.

2) *Dataset description*: We used eight real-world networks with verified ground-truth clusters as testing datasets, including four social networks, *Twitter* [21], *Ego-facebook* [21], *Flickr* [17], and *Google+* [46]; three document networks, *BlogCatalog* [17], *UAI* [24], and *Wiki* [24]; and one biological network *Biogrid* [36].

Twitter, *Ego-facebook* (*Ego*), *Flickr*, and *Google+* (*Gplus*) are four social networks, which are constructed based on snapshots of the users from different online social networking sites. The vertices, edges, and vertex features in these datasets represent social players using the social networking services, social-ties among social players, and their descriptive information, e.g., social tags, locations, and user profiles, respectively.

BlogCatalog (*Blog*), *UAI*, and *Wiki*, are three widely-used document network datasets. In the mentioned three datasets, online documents such as blogs, articles, and web pages, document-document relationships, e.g., citations, and hyperlinks, and descriptive information, such as keywords are respectively represented as vertices, edges, and vertex features.

Different from the aforementioned two categories of network data, *Biogrid* is one that is extracted from vital activities. It is used to describe the interactions between proteins in *Saccharomyces cerevisiae*. In this dataset, vertices, edges, and vertex features are used to represent proteins, protein-protein interactions, and biological fingerprints of the proteins, respectively. As these networks have different sizes and they are extracted from various real-world scenarios, they can be used to test the robustness of different approaches to network clustering. The statistics of these testing datasets are summarized in Table II, where N is the number of vertices, $|E|$ is the number of edges, M is the number of vertex features, and K is the number of ground-truth clusters, respectively.

3) *Evaluation metrics*: To evaluate the performance, two widely used metrics, Normalized Mutual Information (NMI) [14] and Accuracy (Acc) [34], are selected in our experiments to evaluate the performance of different approaches.

The NMI measures the overall accuracy of the matches between detected clusters and the ground-truth. It is defined as:

$$NMI = \frac{\sum_{C_i, C_j^*} Pr(C_i, C_j^*) \log \frac{Pr(C_i, C_j^*)}{Pr(C_i)Pr(C_j^*)}}{\max(H(C), H(C^*))}, \quad (32)$$

$$H(C) = - \sum_i Pr(C_i) \log Pr(C_i),$$

$$H(C^*) = - \sum_j Pr(C_j^*) \log Pr(C_j^*),$$

where $Pr(C_i, C_j^*)$ denotes the probability that the vertices are shared in both the detected cluster i and the ground-truth

cluster j , and $Pr(C_i)$ denotes the probability that a vertex belongs to cluster i .

Different from NMI , Acc measures the accuracy of individually detected clusters. It is defined as follows:

$$Acc = \sum_i \frac{|C_i|}{|C|} f(C_i, C^*), \quad (33)$$

where $|C_i|$ denotes the size of detected cluster i and $f(\cdot, \cdot)$ is defined as the maximum overlap between the detected cluster i and a cluster in the ground-truth database. According to the definition, Acc evaluates the best matching of each individual cluster.

The properties of the two evaluation metrics enable them to evaluate the effectiveness of an approach in a complementary manner, so that a clustering approach can be evaluated comprehensively. Larger values of NMI and Acc indicate a better matching between the detected clusters and the ground-truth.

B. Clustering performance comparison

Social community detection, document segmentation, and biological module identification are typical applications of network clustering. In our experiment, we used the aforementioned four social networks, three document networks, and one biological network to test the effectiveness of all the eleven approaches. Using the ground-truth clusters of the testing datasets which have been verified in previous studies, the clusters discovered by different approaches against the ground-truth are validated in terms of NMI and Acc . The experimental results of all algorithms are summarized in Tables III and IV.

When the detected clusters are evaluated by NMI , VVAMo outperforms all the other baselines in seven out of eight testing datasets, and ranks the second-best on *Biogrid* dataset. In five datasets out of the seven, VVAMo is better than the second-best approach by at least 15%. Specifically, in *Flickr* dataset, VVAMo performs better than ASCD by 22.69%. In *Gplus*, the proposed model performs better than CESNA by 20.83%. In *Blog*, VVAMo outperforms SCI by 17.19%. In *UAI* and *Wiki*, VVAMo is better than the second-best DANMF by 41.73% and 55.75%, respectively.

When Acc is considered, VVAMo still performs robustly when compared with other baselines. VVAMo can obtain the best performance in terms of Acc in seven out of the eight testing datasets, except *Blog*, where it ranks the second best. VVAMo outperforms the second-best approaches by at least 5% in six out of the seven datasets. In *Ego* and *Gplus*, VVAMo is better than DNMF by 5.94% and 11.24%, respectively. In *Flickr*, VVAMo outperforms ASCD by 12.08%. In *UAI* and *Biogrid*, VVAMo outperforms SCI by 25.11% and 5.65%, respectively. In *Wiki*, VVAMo outperforms the second-best DANMF by up to 33.57%.

From the experimental results in terms of NMI and Acc , we can observe that VVAMo is able to obtain a robust performance in uncovering clusters in various networks. When compared with prevalent shallow models, e.g., CoDA, SBM, DNMF, SCI, and ASCD, VVAMo can obtain satisfying clustering performance in various types of network data. In

TABLE III
CLUSTERING PERFORMANCE EVALUATED BY *NMI* (% MEAN $\pm$ STD. DEVIATION). THE BEST PERFORMANCE ON EACH DATASET IS HIGHLIGHTED IN BOLD.

Dataset	Social				Document			Biological
	Twitter	Ego	Flickr	Gplus	Blog	UAI	Wiki	Biogrid
Ncut	48.421 ± 0.367	53.646 ± 0.162	2.457 ± 0.000	12.274 ± 0.588	3.709 ± 0.003	16.855 ± 0.272	8.638 ± 0.412	83.458 ± 0.262
CoDA	60.878 ± 0.559	62.384 ± 1.066	3.277 ± 0.415	29.341 ± 1.044	5.203 ± 0.238	27.632 ± 0.516	30.203 ± 0.950	76.225 ± 0.601
SBM	12.276 ± 1.505	63.527 ± 0.403	2.848 ± 0.565	27.075 ± 0.436	16.739 ± 0.108	28.693 ± 0.276	22.67 ± 0.008	88.868 ± 0.734
DANMF	57.726 ± 1.296	56.447 ± 0.601	5.202 ± 0.994	27.583 ± 1.157	5.527 ± 0.453	31.812 ± 1.497	33.889 ± 0.803	70.885 ± 0.468
DNMF	67.418 ± 0.596	63.221 ± 0.321	20.936 ± 0.354	35.508 ± 1.093	20.501 ± 0.452	28.721 ± 0.813	22.518 ± 0.565	90.471 ± 1.228
MVSC	54.827 ± 0.367	12.71 ± 0.318	3.191 ± 0.002	26.482 ± 0.394	5.862 ± 0.005	13.624 ± 0.190	16.736 ± 0.306	77.401 ± 0.249
CESNA	46.588 ± 0.155	57.513 ± 1.119	12.332 ± 0.968	38.675 ± 0.937	25.413 ± 1.592	17.536 ± 0.566	9.374 ± 0.185	80.088 ± 0.353
SCI	56.889 ± 0.196	44.025 ± 0.509	14.805 ± 0.504	26.938 ± 0.080	33.367 ± 0.396	29.673 ± 1.397	32.204 ± 0.618	92.028 ± 0.195
VGAE	59.873 ± 0.961	60.888 ± 0.895	22.908 ± 0.607	32.468 ± 0.753	19.073 ± 0.327	12.110 ± 0.401	18.590 ± 0.339	85.604 ± 0.386
ASCD	52.636 ± 1.269	45.857 ± 0.249	25.359 ± 1.409	32.590 ± 0.413	21.598 ± 0.521	24.688 ± 1.256	26.965 ± 1.256	86.296 ± 0.314
VVAMo	67.916 ± 0.207	65.422 ± 0.203	31.113 ± 1.228	46.733 ± 0.628	39.104 ± 2.118	45.087 ± 0.839	52.783 ± 1.345	90.983 ± 0.276

contrast, those methods solely utilizing topology, e.g., CoDA, SBM, and DNMF, are hard to obtain a robust performance in those document networks. Unlike networks from social and biological scenarios, the structure of document networks is relatively sparse and naturally vertex features play a more important role in the clustering task. Although methods like CESNA, SCI, and ASCD take into consideration both network topology and vertex features when performing the task of network clustering, they overlook the profound information underlying the intricate network. Thus, there is still some possibility to further improve these shallow models. Besides modeling the cluster preference making use of both network topology and vertex features, VVAMo further considers inferring the latent vertex-vertex preference based on heterogeneous correlations between pairwise vertices and the inclinations leading each vertex to choose corresponding correlations to interact with others. Then, VVAMo leverages such latent preference to allocate those vertices which are truly correlated into the same cluster. Thus, its clustering performance is better.

When compared with deep learning models for network clustering, e.g., DANMF and VGAE, VVAMo is still able to obtain a satisfying performance. Leveraging the power provided by deep learning architecture, methods like DANMF can perform robustly in various network data, although they solely utilize topological information. However, their performance can be further improved if vertex features can be appropriately used. Taking advantage of the convolution operator, graph convolution networks, e.g., VGAE can discover

network clusters via mapping the feature aggregated from 1-hop neighbors into the low dimensional latent space through a multi-layer learning architecture. Though performing robustly, these methods always confront two challenges. One is the topological data carried by the network are not fully utilized as they are only used for the feature aggregation of neighbors. The other is the interpretability of these methods decreases as the model structure goes deeper, which brings many difficulties in understanding the learned results. As a shallow model for network clustering, VVAMo can make use of correlations w.r.t. structure and features to dig out latent preference between pairwise vertices, which is then appropriately used to regularize the model to infer interpretable clusters in various network data. It is seen that VVAMo is able to discover easily understandable clusters as well as ensure its clustering performance to be comparable to both shallow and deep learning approaches.

C. Model convergence test

In addition to the theoretical analysis on the model convergence, we investigate the convergent speed of the proposed model on real-world network data. Specifically, we recorded the values of log-likelihood function of VVAMo for the first 400 iterations on all the eight testing datasets. The curves of the recorded function values are depicted in Fig. 3. As the figure shows, the value of log-likelihood function converges to a stable value in less than 400 iterations on all the testing

TABLE IV
CLUSTERING PERFORMANCE EVALUATED BY Acc (% MEAN $\pm$ STD. DEVIATION). THE BEST PERFORMANCE ON EACH DATASET IS HIGHLIGHTED IN BOLD.

Dataset	Social				Document			Biological
	Twitter	Ego	Flickr	Gplus	Blog	UAI	Wiki	Biogrid
Ncut	42.121 ± 0.122	44.689 ± 0.520	11.921 ± 0.000	26.991 ± 0.184	24.538 ± 0.008	29.527 ± 0.164	17.588 ± 0.208	3.245 ± 0.098
CoDA	66.537 ± 2.545	50.218 ± 1.597	19.091 ± 0.748	39.176 ± 2.221	29.051 ± 0.966	40.557 ± 1.228	45.267 ± 0.873	4.954 ± 0.263
SBM	27.556 ± 1.546	52.959 ± 0.186	17.413 ± 0.785	55.255 ± 0.138	33.237 ± 0.231	43.622 ± 0.104	36.715 ± 0.748	8.528 ± 0.363
DANMF	57.851 ± 1.951	43.352 ± 1.493	17.201 ± 1.320	10.147 ± 0.699	26.135 ± 0.736	43.816 ± 0.834	50.148 ± 1.201	8.865 ± 0.227
DNMF	70.411 ± 0.959	55.239 ± 0.366	38.028 ± 0.551	62.589 ± 1.263	37.954 ± 0.308	43.301 ± 0.835	35.459 ± 0.705	12.130 ± 1.029
MVSC	54.136 ± 0.122	14.236 ± 0.272	17.228 ± 0.001	12.860 ± 0.195	25.808 ± 0.115	27.059 ± 0.327	19.958 ± 0.187	4.609 ± 0.098
CESNA	51.340 ± 0.350	46.124 ± 1.118	33.817 ± 1.425	22.994 ± 0.749	61.366 ± 1.695	23.839 ± 0.570	24.738 ± 0.144	2.570 ± 0.112
SCI	58.120 ± 0.151	35.113 ± 0.124	33.474 ± 0.985	28.169 ± 0.040	51.212 ± 0.722	44.643 ± 1.091	38.843 ± 0.108	12.720 ± 0.063
VGAE	61.893 ± 0.714	47.338 ± 0.645	39.729 ± 0.837	59.903 ± 0.766	39.434 ± 0.801	22.866 ± 0.545	46.611 ± 0.839	6.152 ± 0.057
ASCD	42.792 ± 3.508	35.746 ± 0.087	43.934 ± 2.231	21.767 ± 0.281	40.127 ± 0.519	30.370 ± 0.504	32.952 ± 0.083	6.986 ± 0.124
VVAMo	70.624 ± 0.399	58.519 ± 0.244	49.241 ± 2.276	69.626 ± 1.240	60.931 ± 2.331	55.853 ± 1.037	66.985 ± 1.299	13.439 ± 0.259

Fig. 3. Model convergence on testing datasets

datasets, which showcases its capability of efficiently acquiring the satisfying clustering results.

D. Running time on learning parameters

Besides the previous experiment on the model convergence, we further tested the computational efficiency of VVAMo by recording the running time on model fitting on all the testing datasets. The running time of VVAMo is then compared with that obtained by other network clustering approaches based on matrix factorization, including CESNA, DNMF, and DANMF. The corresponding results are shown in Fig. 4. It is observed that VVAMo may obtain a competitive

Fig. 4. Scalability comparison among different approaches

performance, when compared with other approaches. On most testing datasets, the proposed model is more efficient than CESNA and DNMF, but slightly slower than DANMF, which is the most efficient approach in the experiment. Specifically, VVAMo takes 4433, 920, 2079, 2477, 1043, 650, 320, and 2315 seconds to complete the model fitting on *Twitter*, *Ego*, *Flickr*, *Gplus*, *Blog*, *UAI*, *Wiki*, and *Biogrid*. In contrast, DANMF costs 845, 1688, 1086, 7657, 322, 87, 50, and 6807 seconds on model optimization on the mentioned

Fig. 5. Allocated vicinal vertices in *Ego – facebook* dataset.

testing datasets. Based on the analysis conducted in Section IV-B, the computational complexity of the proposed model follows the order of $O(N^2K + NMK + M^2K + |E|e)$, which indicates VVAMo has to cost some time on the learning of latent preference, and vertex inclinations in each iteration of model fitting. As for other approaches, they do not consider such a modeling method, and perform the task of network clustering through factorizing the data matrix directly carried by the network. This is the reason that VVAMo is a bit slower than the most efficient factorization-based approach, i.e., DANMF. Taking into consideration the clustering performance and computational efficiency, VVAMo is seen as an efficient model that can effectively perform the task of uncovering clusters in various network data.

E. Case study on allocated vicinal vertices

To verify whether VVAMo is able to allocate vicinal vertices sharing high latent preference into the same cluster, we conducted a detailed analysis on the clusters discovered by VVAMo and provide a concrete example. Figure 5 illustrates a pair of cluster members correctly detected by VVAMo in *Ego – facebook* dataset. Why they are correctly detected in the same cluster can be explained using the depicted latent information learned by VVAMo. It is observed that the mutual latent preferences between vertex 58 and 2814 are both larger than 0.1, which is a relatively high value in view of the size of the dataset ($N=4039$), and the topology/feature inclinations of these two vertices are highly similar. Given the learned latent preferences, vertex inclinations w.r.t. topology and features, and the observed correlations, VVAMo deduces that this pair of vicinal vertices has a high probability of being allocated into the same cluster, based on the Boltzmann distribution formulated in Eq. (8). Attempting to assign analogous cluster allocations to the vicinal vertices like the exemplified, VVAMo is capable of learning interpretable clusters in the network, its clustering performance is thereby guaranteed.

VI. CONCLUSIONS

In this paper, we propose a novel model, dubbed Vicinal Vertex Allocated Matrix Factorization (VVAMo), for discovering clusters in network data. Unlike the previous

approaches solely concerning network topology and vertex features, VVAMo is further capable of inferring the latent vertex-vertex preferences, which are jointly determined by topological and feature correlations, and corresponding inclinations of each vertex in the network. Taking advantage of such latent preference to allocate the vicinal vertices sharing high correlations, and similar inclinations with analogous cluster labels, VVAMo is able to discover interpretable clusters in the network. VVAMo has been tested on a number of real-world network datasets and compared with both classical and state-of-the-art approaches. The experimental results demonstrate that VVAMo is able to reveal clusters in the networks constructed by different types of real-world data with high accuracy, and also in a computationally efficient manner. In the future, we will further improve the efficiency of VVAMo by developing the model based on parallel computing, or stochastic inference. We will also try to improve the effectiveness of VVAMo by allowing it to consider correlations between higher-order neighbors.

ACKNOWLEDGMENT

The authors would like to thank the editors and reviewers for their constructive comments and suggestions.

REFERENCES

- [1] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, “Mixed membership stochastic blockmodels,” *Journal of machine learning research*, vol. 9, no. Sep, pp. 1981–2014, 2008.
- [2] L. Bai, Y.-S. Ong, T. He, and A. Gupta, “Multi-task gradient descent for multi-task learning,” *Memetic Computing*, vol. 12, no. 4, pp. 355–369, 2020.
- [3] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.
- [4] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.
- [5] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of statistical mechanics: theory and experiment*, vol. 2008, no. 10, p. P10008, 2008.
- [6] D. Bo, X. Wang, C. Shi, M. Zhu, E. Lu, and P. Cui, “Structural deep clustering network,” in *Proceedings of The Web Conference 2020*, 2020, pp. 1400–1410.
- [7] J. Chang and D. Blei, “Relational topic models for document networks,” in *Artificial Intelligence and Statistics*, 2009, pp. 81–88.
- [8] A. Clauset, M. E. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical review E*, vol. 70, no. 6, p. 066111, 2004.
- [9] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.
- [10] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” *science*, vol. 315, no. 5814, pp. 972–976, 2007.
- [11] P. K. Gopalan, L. Charlin, and D. Blei, “Content-based recommendations with poisson factorization,” in *Advances in Neural Information Processing Systems*, 2014, pp. 3176–3184.
- [12] T. He and K. C. Chan, “Discovering fuzzy structural patterns for graph analytics,” *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 5, pp. 2785–2796, 2018.
- [13] T. He and K. C. Chan, “Measuring boundedness for protein complex identification in ppi networks,” *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 16, no. 3, pp. 967–979, 2018.
- [14] T. He and K. C. Chan, “Misaga: An algorithm for mining interesting subgraphs in attributed graphs,” *IEEE transactions on cybernetics*, vol. 48, no. 5, pp. 1369–1382, 2018.
- [15] T. He, Y. Liu, T. H. Ko, K. C. Chan, and Y. S. Ong, “Contextual correlation preserving multiview featured graph clustering,” *IEEE transactions on cybernetics*, 2019.

- [16] T. Hofmann, "Learning the similarity of documents: An information-geometric approach to document retrieval and categorization," in *Advances in neural information processing systems*, 2000, pp. 914–920.
- [17] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017, pp. 731–739.
- [18] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *arXiv preprint arXiv:1611.07308*, 2016.
- [19] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Advances in neural information processing systems*, 2011, pp. 1413–1421.
- [20] J. B. Lee, G. Nguyen, R. A. Rossi, N. K. Ahmed, E. Koh, and S. Kim, "Dynamic node embeddings from edge streams," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2020.
- [21] J. Leskovec and J. J. McAuley, "Learning to discover social circles in ego networks," in *Advances in neural information processing systems*, 2012, pp. 539–547.
- [22] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in neural information processing systems*, 2014, pp. 2177–2185.
- [23] Z. Li, J. Liu, and K. Wu, "A multiobjective evolutionary algorithm based on structural and attribute similarities for community detection in attributed networks," *IEEE transactions on cybernetics*, vol. 48, no. 7, pp. 1963–1976, 2017.
- [24] Q. Lu and L. Getoor, "Link-based classification," in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 2003, pp. 496–503.
- [25] L. Ma, Y. Ma, Q. Lin, J. Ji, C. A. C. Coello, and M. Gong, "Snegan: Signed network embedding by using generative adversarial nets," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2020.
- [26] J. McAuley and J. Leskovec, "Discovering social circles in ego networks," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 8, no. 1, p. 4, 2014.
- [27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
- [28] F. Nie, X. Wang, and H. Huang, "Clustering and projected clustering with adaptive neighbors," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 977–986.
- [29] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *nature*, vol. 435, no. 7043, pp. 814–818, 2005.
- [30] C. Peng, Z. Zhang, K.-C. Wong, X. Zhang, and D. Keyes, "A scalable community detection algorithm for large graphs using stochastic block models," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.
- [31] C. Pizzuti and A. Socievoli, "Multiobjective optimization and local merge for clustering attributed graphs," *IEEE transactions on cybernetics*, 2019.
- [32] I. Psorakis, S. Roberts, M. Ebdon, and B. Sheldon, "Overlapping community detection using bayesian non-negative matrix factorization," *Physical Review E*, vol. 83, no. 6, p. 066114, 2011.
- [33] M. Qiao, J. Yu, W. Bian, Q. Li, and D. Tao, "Adapting stochastic block models to power-law degree distributions," *IEEE transactions on cybernetics*, vol. 49, no. 2, pp. 626–637, 2018.
- [34] M. Qin, D. Jin, K. Lei, B. Gabrys, and K. Musial-Gabrys, "Adaptive community detection incorporating topology and content in social networks," *Knowledge-Based Systems*, vol. 161, pp. 342–356, 2018.
- [35] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
- [36] C. Stark, B.-J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "Biogrid: a general repository for interaction datasets," *Nucleic acids research*, vol. 34, no. suppl_1, pp. D535–D539, 2006.
- [37] Y. Su, C. Liu, Y. Niu, F. Cheng, and X. Zhang, "A community structure enhancement-based community detection algorithm for complex networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019.
- [38] Y. Sun, J. Han, J. Gao, and Y. Yu, "itopicmodel: Information network-integrated topic modeling," in *2009 Ninth IEEE International Conference on Data Mining*. IEEE, 2009, pp. 493–502.
- [39] X. Teng, J. Liu, and M. Li, "Overlapping community detection in directed and undirected attributed networks using a multiobjective evolutionary algorithm," *IEEE transactions on cybernetics*, 2019.
- [40] Y. Tian, X. Su, Y. Su, and X. Zhang, "Emodmi: A multi-objective optimization based method to identify disease modules," *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2020.
- [41] H. Wang, Y. Yang, and B. Liu, "Gmc: graph-based multi-view clustering," *IEEE Transactions on Knowledge and Data Engineering*, 2019.
- [42] R. Wang, F. Nie, Z. Wang, H. Hu, and X. Li, "Parameter-free weighted multi-view projected clustering with structured graph learning," *IEEE Transactions on Knowledge and Data Engineering*, 2019.
- [43] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI*, vol. 17, 2017, pp. 203–209.
- [44] X. Wang, D. Jin, X. Cao, L. Yang, and W. Zhang, "Semantic community identification in large attribute networks," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
- [45] Z. Xu, Y. Ke, X. Cao, C. Zhou, P. Wei, and X. Gao, "A unified linear convergence analysis of k-svd," *Memetic Computing*, vol. 12, no. 4, pp. 343–353, 2020.
- [46] J. Yang, J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," in *Data Mining (ICDM), 2013 IEEE 13th international conference on*. IEEE, 2013, pp. 1151–1156.
- [47] J. Yang, J. McAuley, and J. Leskovec, "Detecting cohesive and 2-mode communities in directed and undirected networks," in *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014, pp. 323–332.
- [48] F. Ye, C. Chen, and Z. Zheng, "Deep autoencoder-like nonnegative matrix factorization for community detection," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 1393–1402.
- [49] F. Ye, C. Chen, Z. Zheng, R.-H. Li, and J. X. Yu, "Discrete overlapping community detection with pseudo supervision," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 708–717.
- [50] T.-F. Zhao, W.-N. Chen, S. Kwong, T.-L. Gu, H.-Q. Yuan, J. Zhang, and J. Zhang, "Evolutionary divide-and-conquer algorithm for virus spreading control over networks," *IEEE Transactions on Cybernetics*, 2020.

Tiantian He received the Ph.D. degree in Computer Science from the Hong Kong Polytechnic University in 2017. Currently he is a Research Fellow in Data Science & Artificial Intelligence Research Center, School of Computer Science and Engineering, Nanyang Technological University. His research interests include artificial intelligence, machine learning, data-centric transfer optimization, and data mining.

Lu Bai received the B.Eng. and M.Eng. degrees in automation from Xiamen University, Xiamen, China, in 2012 and 2015, respectively, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2020. She is currently a Research Fellow with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. Her research interests include distributed control, optimization, multi-task optimization, and multi-task learning.

Yew-Soon Ong (M'99-SM'12-F'18) received the Ph.D. degree in artificial intelligence in complex design from the University of Southampton, U.K., in 2003. He is President's Chair Professor in Computer Science at Nanyang Technological University (NTU), and holds the position of Chief Artificial Intelligence Scientist of the Agency for Science, Technology and Research Singapore. At NTU, he serves as Director of the Data Science and Artificial Intelligence Research and co-Director of the Singtel-NTU Cognitive & Artificial Intelligence Joint Lab.

His research interest is in artificial and computational intelligence. He is founding Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computational Intelligence and AE of IEEE Transactions on Neural Networks & Learning Systems, IEEE Transactions on Cybernetics, IEEE Transactions on Artificial Intelligence and others. He has received several IEEE outstanding paper awards and was listed as a Thomson Reuters highly cited researcher and among the World's Most Influential Scientific Minds.