Incorporating User's Preference into Attributed Graph Clustering

Wei Ye, Dominik Mautz, Christian Böhm, Ambuj Singh, and Claudia Plant

Abstract—Graph clustering has been studied extensively on both plain graphs and attributed graphs. However, all these methods need to partition the whole graph to find cluster structures. Sometimes, based on domain knowledge, people may have information about a specific target region in the graph and only want to find a single cluster concentrated on this local region. Such a task is called local clustering. In contrast to global clustering, local clustering aims to find only one cluster that is concentrating on the given seed vertex (and also on the designated attributes for attributed graphs). Currently, very few methods can deal with this kind of task. To this end, we propose two quality measures for a local cluster: Graph Unimodality (GU) and Attribute Unimodality (AU). The former measures the homogeneity of the graph structure while the latter measures the homogeneity of the subspace that is composed of the designated attributes. We call their linear combination as Compactness. Further, we propose LOCLU to optimize the Compactness score. The local cluster detected by LOCLU concentrates on the region of interest, provides efficient information flow in the graph and exhibits a unimodal data distribution in the subspace of the designated attributes.

Index Terms—Local clustering, user's preference, attributed graphs, dip test, unimodal, NCut, power iteration

1 Introduction

Data can be collected from multiple sources and modeled as attributed graphs (networks), in which vertices represent entities, edges represent their relations and attributes describe their own characteristics. For example, proteins in a protein-protein interaction network may be associated with gene expressions in addition to their interaction relations; users in a social network may be associated with individual attributes such as interests, residence and demographics in addition to their friendship relations.

One of the major data mining tasks in graphs (networks) is the detection of clusters. Existing methods for cluster detection in attributed graphs can be divided into two categories, i.e., full space attributed graph clustering methods [1], [2] and subspace attributed graph clustering methods [3], [4], [5]. The methods belonging to the first category treat all attributes equally important to the graph structure, while the methods belonging to the second category consider varying relevance of attributes to the graph structure. All these methods need to partition the whole graph to find cluster structures. However, based on domain knowledge, sometimes people may have information about a specific target region in the graph and are only interested in finding a cluster surrounding this local region. Such a task is called local cluster detection, which has aroused a great deal of attention in many applications, e.g., targeted ads, medicine, etc. Without considering scalability, one may think we can first use full space or subspace attributed graph clustering techniques and then return the cluster that contains the target region. However, it is hard to set the number of clusters in real-world graphs. And the cluster content depends on the chosen number of clusters.

To deal with this task, several recent works [6], [7] use short random walks starting from the target region to find the local cluster. Also, some approaches [8], [9] focus on using the graph diffusion methods to find the local cluster. However, these methods are only suitable for detecting local clusters in plain graphs whose vertices have no attributes. Recently, FocusCO [10] has been proposed to find a local cluster of interest to users in attributed graphs. Given an examplar set, it first exploits a metric learning method to learn a projection vector that makes the vertex in the examplar set similar to each other in the projected attribute subspace, then updates the graph weight and finally performs the focused cluster extraction. FocusCO cannot infer the projection vector if the examplar set has only one vertex.

In this paper, given user's preference, i.e., the seed vertex and the designated attributes, we develop a method that can automatically find the vertices that are similar to the given seed vertex. The similarity is measured by the homogeneity both in the graph structure and the subspace that is composed of the designated attributes. To this end, we first propose Compactness to measure the unimodality[1] of the

- *W. Ye and A. Singh are with the Department of Computer Science, University of California, Santa Barbara, CA 93106 USA. E-mail: {weiye, ambuj}@cs.ucsb.edu.*
- *D. Mautz and C. Böhm are with the Institut für Informatik, Ludwig-Maximilians-Universität München, 81377 Munich, Germany. E-mail: {mautz, boehm}@dbs.ifi.lmu.de.*
- *C. Plant is with the Faculty of Computer Science, University of Vienna, 1010 Vienna, Austria. E-mail: claudia.plant@univie.ac.at.*

Manuscript received 11 June 2019; revised 13 Jan. 2020; accepted 14 Feb. 2020. Date of publication 24 Feb. 2020; date of current version 5 Nov. 2021. (Corresponding author: Wei Ye.)
Recommended for acceptance by C. Li.
Digital Object Identifier no. 10.1109/TKDE.2020.2976063

1. In this work, unimodality/unimodal and homogeneity/homogeneous can be used interchangeably.

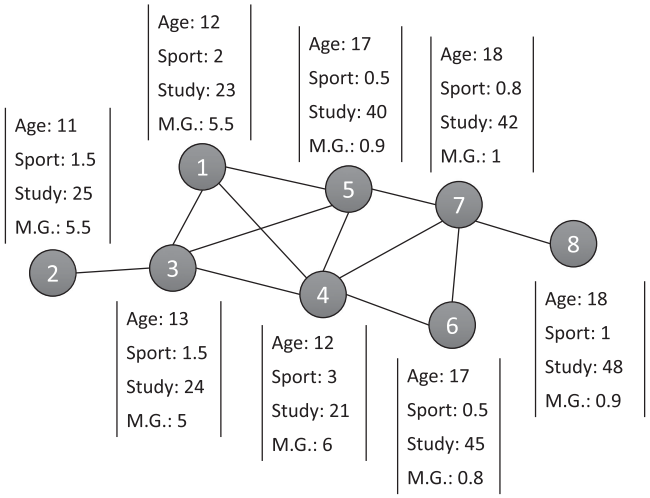


Fig. 1. An example social network.

clusters in attributed graphs. COMPACTNESS is composed of two measures: Graph Unimodality (GU) and Attribute Unimodality (AU). GU measures the unimodality of the graph structure, and AU measures the unimodality of the subspace that is composed of the designated attributes. To consider both the graph structure and attributes, we first embed the graph structure into vector space. Then we consider the graph embedding vector as another designated attribute and apply the local clustering technique separately on each designated attribute. We call the procedure to find a local cluster as LOCLU.

Let us use a simple example to demonstrate our motivation. Fig. 1 shows an example social network, in which the vertices represent students in a middle school, the edges represent their friendship relations, and the attributes associated to each vertex are age (year), sport time (hour) per week, studying time (hour) per week and playing mobile game (M.G.) time (hour) per week. Given vertex 4 and the designated attribute M.G., the task is to find a local cluster around the vertex 4. (This task is of interest to mobile game producers.) Conventional diffusion-based local clustering method such as HK [9] finds a cluster $\mathcal{C}_1 = \{1, 3, 4, 5, 6, 7\}$. However, this cluster is not homogeneous in the subspace of the M.G. attribute. Compared with $\mathcal{C}_1$, the cluster $\mathcal{C}_2 = \{1, 2, 3, 4\}$ is more local, which is concentrated on the vertex 4 and the M.G. attribute.

The main contributions are as follows:

- We introduce the univariate statistic hypothesis test called Hartigans' dip test [11] to a new user-centric problem setting: incorporating user's preference into attributed graph clustering.
- We propose COMPACTNESS, a new quality measure for clusters in attributed graphs. COMPACTNESS measures the homogeneity (unimodality) of both the graph structure and subspace that is composed of the designated attributes.
- We propose LOCLU to optimize the COMPACTNESS score.
- We demonstrate the effectiveness and efficiency of LOCLU by conducting experiments on both synthetic and real-world attributed graphs.

2 PRELIMINARIES

2.1 Notation

In this work, we use lower-case Roman letters (e.g., a, b) to denote scalars. We denote vectors (column) by boldface lower case letters (e.g., $\mathbf{x}$) and denote its ith element by $\mathbf{x}(i)$. Matrices are denoted by boldface upper case letters (e.g., $\mathbf{X}$). We denote entries in a matrix by non-bold lower case letters, such as x_{ij}. Row i of matrix $\mathbf{X}$ is denoted by $\mathbf{X}(i,:)$, column j by $\mathbf{X}(:,j)$. A set is denoted by calligraphic capital letters (e.g., $\mathcal{S}$). An undirected attributed graph is denoted by $\mathsf{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ is a set of graph vertices with number $n = |\mathcal{V}|$ of vertices, $\mathcal{E}$ is a set of graph edges with number $e = |\mathcal{E}|$ of edges and $\mathbf{X} \in \mathbb{R}^{n \times d}$ is a data matrix of attributes associated to vertices, where d is the number of attributes. An adjacency matrix of vertices is denoted by $\mathbf{A} \in \mathbb{R}^{n \times n}$ with $a_{ij} = 1(i \neq j)$ and $a_{ij} = 0(i = j)$. The degree matrix $\mathbf{D}$ is a diagonal matrix associated with $\mathbf{A}$ with $d_{ii} = \sum_j a_{ij}$. The random walk transition matrix $\mathbf{W}$ is defined as $\mathbf{D}^{-1}\mathbf{A}$. The Laplacian matrix is denoted as $\mathbf{L} = \mathbf{I} - \mathbf{W}$, where $\mathbf{I}$ is the identity matrix. An attributed graph cluster is a subset of vertices $\mathcal{C} \subseteq \mathcal{V}$ with attributes. The indicator function is denoted by $\mathbb{1}(x)$.

2.2 The Dip Test

Before introducing the concept of the dip test, let us first clarify the definitions of unimodal distribution and multimodal distribution. In statistics, a unimodal distribution refers to a probability distribution that only has a single mode (i.e., peak). If a probability distribution has multiple modes, it is called multimodal distribution. From the behavior of the cumulative distribution function (CDF), unimodal distribution can also be defined as: if the CDF is convex for $x < m$ and concave for $x > m$ (m is the mode), then the distribution is unimodal.

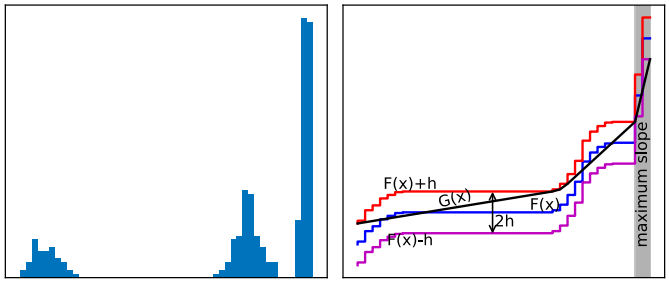
Now let us introduce a univariate statistic hypothesis test which is called Hartigans' dip test [11] as follows:

Theorem 1 [11]. *Let $F(x)$ be a distribution function. Then $D(F) = 2h$ (h is the dip test value) only if there exists a nondecreasing function $G(x)$ such that for some $x_l \leq x_u$:*

- *$G(x)$ is the greatest convex minorant (g.c.m.) of $F(x) + h$ in $(-\infty, x_l)$.*
- *$G(x)$ has constant maximum slope in $[x_l, x_u]$ (modal interval).*
- *$G(x)$ is the least concave majorant (l.c.m.) of $F(x) - h$ in (x_u, ∞).*
- *$h = \sup_{x \notin [x_l, x_u]} |F(x) - G(x)| \geq \sup_{x \in [x_l, x_u]} |F(x) - G(x)|$.*

The g.c.m. of $F(x)$ in $(-\infty, x_l]$ is $\sup(L(x))$ for $x \leq x_l$, where the $\sup(\cdot)$ is taken over all functions $L(x)$ that are convex in $(-\infty, x_l]$ and nowhere greater than $F(x)$. The l.c.m. of $F(x)$ in $[x_u, \infty)$ is $\inf(L(x))$ for $x \geq x_u$, where the $\inf(\cdot)$ is taken over all functions $L(x)$ that are concave in $[x_u, \infty)$ and nowhere less than $F(x)$.

The dip test is the infimum among the supremum computed between the cumulative distribution function (CDF) of $F(x)$ and the CDF of $G(x)$ from the set of unimodal distributions. The dip test measures the departure of $F(x)$ from unimodality. As pointed out in [11], the class of uniform



(a) Histogram. (b) Dip solution.

Fig. 2. The demonstration of the dip test.

distributions is the most suitable for the null hypothesis, because their dip test values are stochastically larger than those of other unimodal distributions. Note that the higher the dip test value, the more multimodal the distribution. Also note that the dip test value is in the range $[0, 0.25)$ [12].

Let us use Fig. 2 to demonstrate the main idea behind the dip test. Fig. 2a shows the histogram of the x-axis projection of the data shown in Fig. 3a. The blue curve ($F(x)$) in Fig. 2b is the CDF of the x-axis projection of the data. Note that the histogram is for visual comparison only. The dip test only needs $F(x)$. To measure the unimodality of $F(x)$, the dip test tries to fit a piecewise-linear function $G(x)$ onto it. Then, the twice of the dip test value $2h$ is defined as the maximum achievable vertical offset for two copies of $F(x)$ (the red and magenta curves, i.e., $F(x)+h$ and $F(x)-h$ in Fig. 2b) such that $G(x)$ does not violate its unimodal rules (convex up to the modal interval (the shade area in Fig. 2b) and concave after it). The farther $F(x)$ strays from unimodality, the larger the required offset between the two copies of $F(x)$. For more details, please refer to [11], [12], [13].

The p-value for the dip test is then computed by comparing $D(F(x))$ with $D(G(x)^{(q)})$ b times, each time with a different n observations from $G(x)$, and the proportion $\sum_{1\le q\le b} \mathbb{1}(D(F(x)) \le D(G(x)^{(q)}))/b$ is the p-value. If the p-value is greater than a significance level α, say 0.05, the null hypothesis H_0 that $F(x)$ is unimodal is accepted. Otherwise H_0 is rejected in favor of the alternative hypothesis H_1 that $F(x)$ is multimodal.

3 Method LOCLU

3.1 Objective Function

The problem of incorporating user's preference into attributed graph clustering can be defined as follows:

Incorporating User's Preference into Attributed Graph Clustering. Given an attributed graph $\mathbf{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, a seed vertex v_q, and the indexes of the designated attributes $\mathcal{I} = \{a_1, a_2, \ldots, a_u\}$, find a cluster $\mathcal{C} = \{v_1, v_2, \ldots, v_q, \ldots\}$ around the seed vertex v_q, such that the cluster $\mathcal{C}$ is not only unimodal in the graph structure but also in the subspace that is composed of the designated attributes.

In order to find the local cluster that satisfies the definition, we need to consider the information from both the graph structure and attributes. First, we consider the graph structure. We propose Graph Unimodality (GU) to measure the unimodality of the graph structure.

Definition 1 (Graph Unimodality). *For a local cluster $\mathcal{C} = \{v_1, v_2, \ldots, v_q, \ldots\}$ around the given seed vertex v_q, its graph unimodality is defined as*

$$GU(\mathcal{C}) = \frac{1}{r}\sum_{i=1}^{r} D(F(\mathbf{e}_i(\mathcal{C}))), \tag{1}$$

where r is the dimension of the graph embedding $\mathbf{E}$, $\mathbf{e}_i$ is the ith embedding vector, and $F(\mathbf{e}_i(\mathcal{C}))$ is the CDF of $\mathbf{e}_i(\mathcal{C})$.

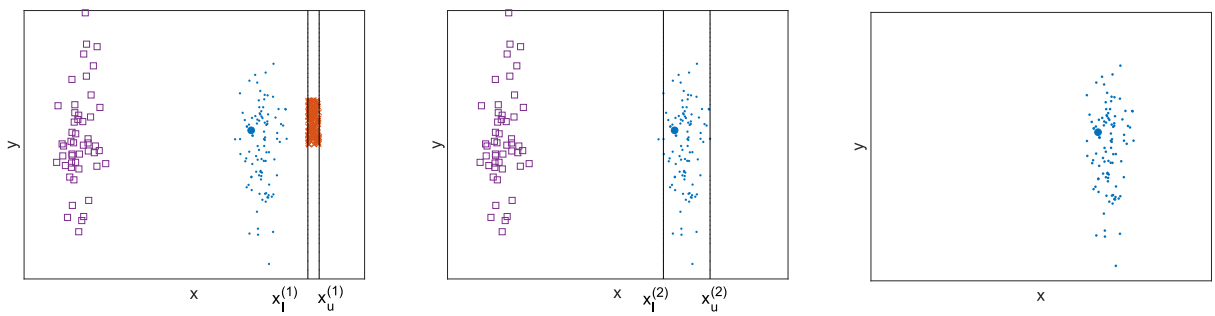
Graph Unimodality (GU) measures the unimodality of the graph structure in a detected local cluster. The lower the GU value, the more unimodal a local cluster in the graph structure. We use the spectral embedding method, especially normalized cut (NCut) [15], to find the embedding $\mathbf{E}$ of the graph structure. The definition of the NCut is

$$\mathrm{NCut}(\mathcal{C}) = \frac{\mathrm{cut}(\mathcal{C}, \overline{\mathcal{C}})}{\mathrm{vol}(\mathcal{C})}, \tag{2}$$

where $\mathrm{cut}(\mathcal{C}, \overline{\mathcal{C}}) = \sum_{v_i \in \mathcal{C}, v_j \in \overline{\mathcal{C}}} a_{ij}$ and $\mathrm{vol}(\mathcal{C}) = \sum_{v_i \in \mathcal{C}, v_j \in \mathcal{V}} a_{ij}$.

Equation (2) can be equivalently rewritten as (for a more detailed explanation, please refer to [16])

$$\begin{aligned} &\mathrm{NCut}(\mathcal{C}) = \mathbf{eLe} \\ &s.t.\ \mathbf{eDe} = \mathrm{vol}(\mathbf{G}) \\ &\mathbf{De} \perp \mathbf{1}, \end{aligned} \tag{3}$$



(a) The x-axis projection is multimodal with the modal interval $[x_l^{(1)}, x_u^{(1)}]$.

(b) The x-axis projection is multimodal with the modal interval $[x_l^{(2)}, x_u^{(2)}]$.

(c) The x-axis projection is unimodal and the local clustering stops.

Fig. 3. The demonstration for the local clustering technique. Assume that the attributes x and y are associated with a local graph cluster. Given the seed vertex (the big blue dot) and the index of the designated attribute (x), we want to find a local cluster around the blue dot, which is unimodal in the subspace that is composed of the designated attribute x.

where $\mathbf{e}$ is the cluster indicator vector (embedding vector) and $\mathbf{eLe}$ is the cost of the cut and $\mathbf{1}$ is a constant vector whose entries are all 1. Note that finding the optimal solution is known to be NP-hard [17] when the values of $\mathbf{e}$ are constrained to $\{1, -1\}$. But if we relax the objective function to allow it take values in $\mathbb{R}$, a near optimal partition of the graph G can be derived from the eigenvector having the second smallest eigenvalue of $\mathbf{L}$. More generally, embedding $\mathbf{E}$ that is composed of k eigenvectors with the k smallest eigenvalues partition the graph into k subgraphs with near optimal normalized cut value.

Second, we consider the attributes. We propose Attribute Unimodality (AU) to measure the unimodality of the subspace that is composed of the designated attributes.

Definition 2 (Attribute Unimodality). *For a local cluster $\mathcal{C} = \{v_1, v_2, \ldots, v_q, \ldots\}$ around the given seed vertex v_q, its attribute unimodality is defined as*

$$AU(\mathcal{C}) = \frac{1}{u}\sum_{i=1}^{u} D(F(\mathbf{x}_i(\mathcal{C}))), \tag{4}$$

where u is the number of the designated attributes, $\mathbf{x}_i$ is the a_ith designated attribute, and $F(\mathbf{x}_i(\mathcal{C}))$ is the CDF of $\mathbf{x}_i(\mathcal{C})$.

Attribute Unimodality (AU) measures the unimodality of the subspace that is composed of the designated attributes in a detected local cluster. The lower the AU value, the more unimodal a local cluster in the subspace.

To measure the unimodality of both the graph structure and attributes of a local cluster, our objective function integrates both GU and AU into one framework, which is called COMPACTNESS

$$Compactness(\mathcal{C}) = \mathrm{GU}(\mathcal{C}) + \mathrm{AU}(\mathcal{C}). \tag{5}$$

In the following, we will elaborate the optimization method LOCLU. It employs a dip test based local clustering technique on the embedding of the graph structure and the designated attributes. The detected local cluster is unimodal both in the graph structure and the designated attributes.

3.2 Optimizing Attribute Unimodality

There are many clustering techniques for the numerical attributes, e.g., k-means, EM, DBSCAN [14], etc. One possible idea is inputing proper parameters (such as the number of clusters for k-means, and MinPts and ϵ for DBSCAN) to the clustering techniques and letting them return the cluster that includes the given seed vertex. However, the disadvantage is that the parameters are difficult to set. The cluster assignments change with different numbers of clusters. Moreover, many clustering techniques need to partition the whole dataset, which is very time- and resource-consuming. Alternatively, we can consider the attributes and graph structure simultaneously and apply some attributed graph clustering technique. The problems we face are the same as described above. In this paper, we would like to develop a local clustering technique. The technique does not require the number of clusters k, which is difficult to set in the real world datasets.

Note that the dip test returns a modal interval, in which the distribution of data is unimodal. Our idea is to employ the modal interval to find a local cluster around the given seed vertex. Our perspective is that data points in the modal interval belong to one cluster. We cluster vertices according to their positions and the position of the modal interval. Thus, other statistical tests that do not return the modal interval cannot be adopted. We use Fig. 3 to elaborate the main idea of our local clustering technique. Assume that Fig. 3a shows two numerical attributes x and y associated with a local graph cluster. The two attributes have three clusters inside. The purple and blue clusters follow Gaussian distributions. The orange cluster follows a uniform distribution. The big blue dot in the blue cluster reveals the numerical values of the given seed vertex and we want to find a local cluster concentrating on this big blue dot and the x-axis.

We input the x-axis projection of the data into the dip test and it returns a p-value and a modal interval $[x_l^{(1)}, x_u^{(1)}]$. In this case, the p-value is 0 that is below the significance level $\alpha = 0.05$, which means the x-axis projection of the data is multimodal. If the given seed point is on the left side of $x_l^{(1)}$, we remove the data points that situate on the right side of $x_l^{(1)}$ and dip over the x-axis projection of the remaining data. If the given seed point is on the right side of $x_u^{(1)}$, we remove the data points that situate on the left side of $x_u^{(1)}$ and dip over the x-axis projection of the remaining data. Otherwise, we dip over the x-axis projection of the data situated in the modal interval $[x_l^{(1)}, x_u^{(1)}]$. We repeat the process until the x-axis projection of the remaining data that contains the given seed point is unimodal.

In our case, since $[x_l^{(1)}, x_u^{(1)}]$ does not contain the big blue dot, we remove the data points that situate on the right side of $x_l^{(1)}$ and continue to dip over the x-axis projection of the remaining data. The dip test on the x-axis projection of the remaining data (shown in Fig. 3b) returns a p-value of 0, which indicates that the remaining data is still multimodal. Because the given seed point is within the new modal interval $[x_l^{(2)}, x_u^{(2)}]$, we extract the data points situated in this modal interval and dip over their x-axis projection. Since the p-value returned by the dip test is 0.937 that is greater than the significance level $\alpha = 0.05$, which means the x-axis projection of the data points (shown in Fig. 3c) is unimodal, we terminate the recursive process and return the found local cluster (shown in Fig. 3c).

In the above, we recursively dip over the x-axis projection of the data to find the local cluster. If given the indexes of attributes $\mathcal{I} = \{a_1, a_2, \ldots, a_u\}$, we will first compute the dip test value of each attribute and then dip over the attributes according to their dip test values, from the highest to the lowest. In this way, the most multimodal attribute will be first explored. The insight is that the directions which depart the most from unimodality are promising for clustering.

Theorem 2. *For a random variable $X = [x_1, x_2, \ldots, x_n]$, the local clustering method will find a unimodal cluster whose values are in the interval $[x_l, x_u]$. The probability distribution function (PDF) of the data points in the interval $[x_i, x_j]$ $(\forall i, j, x_l \le x_i < x_j \le x_u)$ is unimodal.*

Proof. If the PDF of the data points in the interval $[x_i, x_j]$ $(\forall i, j, x_l \le x_i < x_j \le x_u)$ is multimodal, the PDF of the data points in the interval $[x_l, x_u]$ should also be multimodal. Thus, the cluster in the interval $[x_l, x_u]$ is not unimodal and the local clustering method will continue

dipping over the interval $[x_l, x_u]$ until the PDF of the remaining data points is unimodal. □

3.3 Optimizing Graph Unimodality

A good partition of the graph structure is achieved by using the local clustering method in the embedding vector space. In this work, the graph embedding just contains one eigenvector. Each eigenvector bisects the graph into two clusters. We can consider the cluster that contains the seed vertex as the required local cluster. We can first compute the second smallest eigenvector $\mathbf{e}_2$ of the graph Laplacian matrix $\mathbf{L}$. Then, we use the proposed local clustering technique on it to find a local cluster that contains the seed vertex v_q. For large-scale graphs, the eigen-decomposition of $\mathbf{L}$ is $\mathcal{O}(n^3)$ which is impractical. Instead, we use the power iteration method [18] to compute an approximate eigenvector.

The power iteration (PI) is a fast method to compute the dominant eigenvector of a matrix. Note that the k largest eigenvector of $\mathbf{W}$ are also the k smallest eigenvector of $\mathbf{L}$. The power iteration method starts with a randomly generated vector $\mathbf{v}^0$ and iteratively updates as follows:

$$\mathbf{v}^t = \frac{\mathbf{W}\mathbf{v}^{t-1}}{\|\mathbf{W}\mathbf{v}^{t-1}\|_1}. \tag{6}$$

Suppose $\mathbf{W}$ has eigenvectors (embedding vectors) $\mathbf{E} = [\mathbf{e}_1; \mathbf{e}_2; \ldots; \mathbf{e}_n]$ with eigenvalues $\mathbf{\Lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_n]$, where $\lambda_1 = 1$ and $\mathbf{e}_1$ is constant. We have $\mathbf{WE} = \mathbf{\Lambda E}$ and in general $\mathbf{W}^t\mathbf{E} = \mathbf{\Lambda}^t\mathbf{E}$. When ignoring renormalization, Equation (6) can be written as

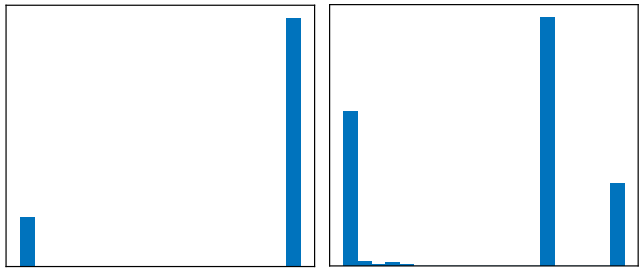
$$\begin{aligned}\mathbf{v}^t &= \mathbf{W}\mathbf{v}^{t-1} = \mathbf{W}^2\mathbf{v}^{t-2} = \cdots = \mathbf{W}^t\mathbf{v}^0 \\ &= \mathbf{W}^t(c_1\mathbf{e}_1 + c_2\mathbf{e}_2 + \cdots + c_n\mathbf{e}_n) \\ &= c_1\mathbf{W}^t\mathbf{e}_1 + c_2\mathbf{W}^t\mathbf{e}_2 + \cdots + c_n\mathbf{W}^t\mathbf{e}_n \\ &= c_1\lambda_1^t\mathbf{e}_1 + c_2\lambda_2^t\mathbf{e}_2 + \cdots + c_n\lambda_n^t\mathbf{e}_n,\end{aligned} \tag{7}$$

where $\mathbf{v}^0$ can be denoted by $c_1\mathbf{e}_1 + c_2\mathbf{e}_2 + \cdots + c_n\mathbf{e}_n$ which is a linear combination of all the original orthonormal eigenvectors. Since the orthonormal eigenvectors form a basis for $\mathbb{R}^n$, any vector can be expanded by them.

From Equation (7), we have the following:

$$\frac{\mathbf{v}^t}{c_1\lambda_1^t} = \mathbf{e}_1 + \frac{c_2}{c_1}\left(\frac{\lambda_2}{\lambda_1}\right)^t\mathbf{e}_2 + \cdots + \frac{c_n}{c_1}\left(\frac{\lambda_n}{\lambda_1}\right)^t\mathbf{e}_n. \tag{8}$$

So the convergence rate of PI towards the dominant eigenvector $\mathbf{e}_1$ depends on the significant terms $(\frac{\lambda_i}{\lambda_1})^t (2 \leqslant i \leqslant n)$. If we let the power iteration method run long enough, it will converge to the dominant eigenvector $\mathbf{e}_1$ which is of little use in clustering. If we define the velocity at t to be the vector $\boldsymbol{\delta}^t = \mathbf{v}^t - \mathbf{v}^{t-1}$ and define the acceleration at t to be the vector $\boldsymbol{\epsilon}^t = \boldsymbol{\delta}^t - \boldsymbol{\delta}^{t-1}$, we can stop the power iteration when $\|\boldsymbol{\epsilon}^t\|_{max}$ is below a threshold $\hat{\epsilon}$. We use $\mathbf{v}^t$ as the graph embedding vector. Fig. 4 shows $\mathbf{e}_2$ and $\mathbf{v}^t$ of the graph Laplacian matrix of the data shown in Fig. 3. Compared with the PDF of $\mathbf{e}_2$, the PDF of $\mathbf{v}^t$ is more multimodal and thus is more promising for clustering.



(a) Histogram of $\mathbf{e}_2$. (b) Histogram of $\mathbf{v}^t$.

Fig. 4. Comparison between the PDFs of $\mathbf{e}_2$ and $\mathbf{v}^t$.

3.4 Implemetation Details and Analysis

We can consider the embedding vector $\mathbf{v}^t$ as another designated attribute and perform local clustering on it. Let us elaborate the algorithmic details of LOCLU whose pseudo-code is given in Algorithm 1. Line 3 computes the random walk transition matrix, which costs $\mathcal{O}(e)$ where e represents the number of edges in the graph. Line 4 initializes the starting vector for the power iteration method. Lines 5–9 use the power iteration method to compute an embedding vector for the graph structure. The power iteration method is guaranteed to converge (please refer to [18]). The time complexity for the power iteration method is $\mathcal{O}(e)$ [19]. Line 10 considers the embedding vector $\mathbf{v}^t$ as another designated attribute and concatenates it to the data matrix $\mathbf{X}$. Lines 11–16 perform local clustering separately on the designated attributes. Lines 11–13 compute the dip test values and p-values for the designated attributes. At lines 15–16, we dip over the designated attributes according to their dip test values, from the highest to the lowest. Line 16 uses the local clustering method to find a local cluster around the given seed vertex on each designated attribute. The time complexity of the dip test at line 12 is $\mathcal{O}(n)$ [11]. Note that the dip test method first sorts the input data, which costs $\mathcal{O}(n \cdot \log(n))$ time. Thus, the time complexity of lines 11–14 in Algorithm 1 is $\mathcal{O}(u \cdot n \cdot \log(n))$, where u is the number of the designated attributes and n is the number of vertices.

The local clustering method is given in Algorithm 2. It recursively dips over the designated attribute until finding the local cluster around the given seed vertex. Line 3 dips over the designated attribute. At lines 4–9, if the given seed vertex does not belong to the current modal interval, we extract the vertices whose attribute values are on the left side of x_l (line 5) or on the right side of x_u (line 7) and update the cluster $\mathcal{C}$; at line 9, if the given seed vertex belongs to the current modal interval, we update the cluster $\mathcal{C}$ with the vertices whose attribute values are inside the modal interval. The time complexity of the dip test at line 3 is $\mathcal{O}(n \cdot \log(n))$. Thus, the time complexity of the local clustering procedure is bounded by $\mathcal{O}(k \cdot n \cdot \log(n))$, where $k \ll n$ is the number of modes in the data. We remark that the local clustering method is guaranteed to converge. The worst case is that each vertex is a mode and local clustering method finds the given seed vertex as the local cluster, which will lead to the termination of the dip test. Thus, LOCLU is also guaranteed to converge. The time complexity of LOCLU is $\mathcal{O}((u+k) \cdot n \cdot \log(n) + e)$.

Theorem 3. *The local cluster detected by LOCLU is unimodal in each designated attribute and the graph structure.*

Proof. For the data matrix $\mathbf{X} = [\mathbf{X}, \mathbf{v}^t]$ at line 10 in Algorithm 1, we first apply the local clustering method on the attribute with the highest dip test value and it will find a unimodal cluster in the interval $[x_{l_1}^{(1)}, x_{u_1}^{(1)}]$. Then we apply it on the attribute with the second largest dip test value and it will find a unimodal cluster in the interval $[x_{l_2}^{(2)}, x_{u_2}^{(2)}]$, where $l_1 \leq l_2 < u_2 \leq u_1$. From Theorem 2, we know that the data points in the interval $[x_{l_2}^{(1)}, x_{u_2}^{(1)}]$ of the attribute with the highest dip test value remains unimodal. If we apply the local clustering method on each column of $\mathbf{X}$, from the highest to the lowest with respect to their dip test values, the final local cluster detected by LOCLU is unimodal in each designated attribute and the graph structure. □

Algorithm 1. LOCLU

Input: Adjacency matrix $\mathbf{A}$, data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, the seed vertex index q, the indexes of the designated attributes $\mathcal{I} = \{a_1, a_2, \ldots, a_u\}$
Output: Local cluster $\mathcal{C}$

1 $\hat{\epsilon} \leftarrow 0.001, t \leftarrow 0$, iter $\leftarrow 1000$;
2 $\mathcal{C} \leftarrow [1:n]$ / * $\mathcal{C}$ contains the indexes of vertices. * /
3 compute the random walk transition matrix $\mathbf{W}$;
4 $\mathbf{v}^0 \leftarrow$ **randn** $(n, 1)$;
/ *power iteration * /
5 **repeat**
6 $\mathbf{v}^{t+1} \leftarrow \frac{\mathbf{W}\mathbf{v}^t}{\|\mathbf{W}\mathbf{v}^t\|_1}$;
7 $\boldsymbol{\delta}^{t+1} \leftarrow |\mathbf{v}^{t+1} - \mathbf{v}^t|$;
8 $t \leftarrow t + 1$;
9 **until** $\|\boldsymbol{\delta}^{t+1} - \boldsymbol{\delta}^t\|_{\max} \leq \hat{\epsilon}$ **or** $t \geq$ iter;
10 $\mathbf{X} \leftarrow [\mathbf{X}, \mathbf{v}^t]$, $a_{u+1} \leftarrow d+1$, $\mathcal{I} \leftarrow \mathcal{I} \cup a_{u+1}$ / * $[\cdot,\cdot]$ means concatenation. * /
/ * Perform local clustering separately on the embedding vector of the graph structure and designated attributes. * /
11 **for** $i \leftarrow 1$ **To** $u+1$ **do**
12 $[\text{dip}, p\text{-value}, \mathbf{t}, x_l, x_u] \leftarrow$ **DipTest**$(\mathbf{X}(:, a_i))$;
13 $\mathbf{d}(i) \leftarrow \text{dip}$;
14 $[\mathbf{d}, \mathbf{s}] \leftarrow$ **sort**$(\mathbf{d})$ / * descending sort, $\mathbf{s}$ contains the indexes of attributes sorted by their dip test values. * /
15 **for** $i \leftarrow 1$ **To** $u+1$ **do**
16 $\mathcal{C} \leftarrow$ LocalClustering $(\mathcal{C}, \mathbf{X}, q, \mathbf{s}(i))$;
17 **return** $\mathcal{C}$;

4 Experimental Evaluation

4.1 Experiment Settings

We thoroughly evaluate LOCLU on cluster quality and runtime using both synthetic and real-world attributed graphs. We compare LOCLU with baseline methods whose descriptions are as follows:

- FocusCO [10] identifies the relevance of vertex attributes that makes the user-provided examplar vertices similar to each other. Then it reweighs the graph edges and extracts the focused cluster.
- SG-Pursuit [4] is a generic and efficient method for detecting subspace clusters in attributed graphs. The main idea is to iteratively identify the intermediate solution that is close-to-optimal and then project it to the feasible space defined by the topological and sparsity constraints.
- UNCut [5] proposes unimodal normalized cut to find cohesive clusters in attributed graphs. The homogeneity of attributes is measured by the proposed unimodality compactness which also exploits Hartigans' dip-test.
- AMEN [20], [21] develops a measure called Normality to quantify both internal consistency and external separability of a graph cluster. Then, the graph cluster that has the best Normality score is extracted.
- AGC [22] is an adaptive graph convolution method for attributed graph clustering, using spectral convolution filters on the vertex attributes.
- HK [9] is a local and deterministic method to accurately compute a heat kernel diffusion in a graph. Then, it finds small conductance community around a given seed vertex. HK only considers the graph structure.

We use the Normalized Mutual Information (NMI) [23] and the F_1 score [9], [24] to evaluate the cluster quality. NMI is a widely used metric for computing clustering accuracy of a method against the desired ground truth. NMI is defined as $NMI(\mathcal{C}^*, \mathcal{C}) = \frac{2 \times I(\mathcal{C}^*; \mathcal{C})}{H(\mathcal{C}^*) + H(\mathcal{C})}$, where $\mathcal{C}^*$ is ground truth, $\mathcal{C}$ is the detected cluster, $I(\cdot;\cdot)$ is mutual information, $H(\cdot)$ is entropy. F_1 score is the harmonic mean of precision P and recall R and is defined as $F_1 = 2 \cdot \frac{P \cdot R}{P + R}$, where $P = \frac{|\mathcal{C} \cap \mathcal{C}^*|}{|\mathcal{C}|}$, $R = \frac{|\mathcal{C} \cap \mathcal{C}^*|}{|\mathcal{C}^*|}$. The higher the NMI and F_1 score, the better the clustering.

Algorithm 2. LocalClustering

Input: Cluster $\mathcal{C}$, data matrix $\mathbf{X}$, the seed vertex index q, the index s of the designated attribute
Output: Local cluster $\mathcal{C}$

1 **repeat**
2 $\mathbf{x} \leftarrow \mathbf{X}(\mathcal{C}, s)$;
3 $[\text{dip}, p\text{-value}, x_l, x_u] \leftarrow$ **DipTest**$(\mathbf{x})$;
4 **if** $\mathbf{x}(q) < x_l$ **then**
5 $\mathcal{C} \leftarrow \{\sigma_1, \sigma_2, \ldots\}$ / * $\{\mathbf{x}(\sigma_1), \mathbf{x}(\sigma_2), \ldots\} < x_l$ * /
6 **else if** $\mathbf{x}(q) > x_u$ **then**
7 $\mathcal{C} \leftarrow \{\sigma_1, \sigma_2, \ldots\}$ / * $\{\mathbf{x}(\sigma_1), \mathbf{x}(\sigma_2), \ldots\} > x_u$ * /
8 **else**
9 $\mathcal{C} \leftarrow \{\sigma_1, \sigma_2, \ldots\}$ / * $\{\mathbf{x}(\sigma_1), \mathbf{x}(\sigma_2), \ldots\} \in [x_l, x_u]$ * /
10 **until** $p\text{-}value > 0.05$;
11 **return** $\mathcal{C}$;

For the experiments on the synthetic graphs, we give the correct number of clusters to SG-Pursuit, UNCut, and AGC. We also give the correct size of each cluster to SG-Pursuit. We compute the NMI and the F_1 score for each combination of the detected cluster and the ground truth cluster and report the best NMI and F_1 score. Note that the selection of the seed vertex and the indexes of the designated attributes depend on user's preferences. In the experiments, we randomly sample a vertex as the seed vertex, and only dip over the most multimodal attribute whose dip test value is the

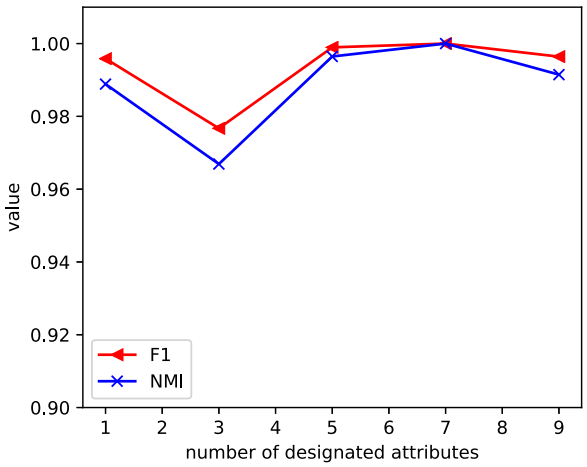


Fig. 5. Clustering results of LOCLU with the increasing number of designated attributes.

highest. FocusCO needs to compute the relevant attribute weight vector β which is then used to weight each edge in the graph. For a fair comparison, the entry in β that corresponds to the attribute whose dip test value is the highest is set to one and the other entries are set to zero. AMEN also needs to compute the relevant attribute weight vector. Analogous to FocusCO, we set the corresponding entry to one and other entries zero. Since HK is designed for plain graphs and cannot handle attribute information, we incorporate the attribute information by weighing the edges of the graph using the weighting vector β. We also report the results of HK on the graph structure. We call these two versions as weighted HK (w) and unweighted HK (uw). We run each experiment 50 times and at each time we randomly sample a seed vertex.

All the experiments are run on the same machine with the Ubuntu 18.04.1 LTS operating system and an Intel Core Quad i7-3770 with 3.4 GHz and 32 GB RAM. LOCLU is written in Java. The code of LOCLU and all the synthetic and real-world graphs used in this work are publicly available at Github.[2]

4.2 Synthetic Graphs

4.2.1 Clustering Quality

To study the clustering performance, we generate synthetic graphs with varying numbers of vertices n, attributes d, varying ratio of relevant attribute and variable cluster size range. For the case of varying n, we fix the attribute dimension $d = 20$ and the ratio of relevant attributes 50 percent. For the case of varying d, we fix the number of vertices $n = 1000$ and the ratio of relevant attributes 50 percent. For the case of varying the ratio of relevant attribute, we fix the attribute dimension $d = 20$ and the number of vertices $n = 1000$. For varying the cluster size range, we fix the attribute dimension $d = 20$ and the ratio of relevant attributes 50 percent.

All the graphs are generated based on the planted partitions model [25] which is also used in FocusCO and SG-Pursuit. Given the desired number of vertices in each cluster,

2. https://github.com/yeweiysh/LOCLU

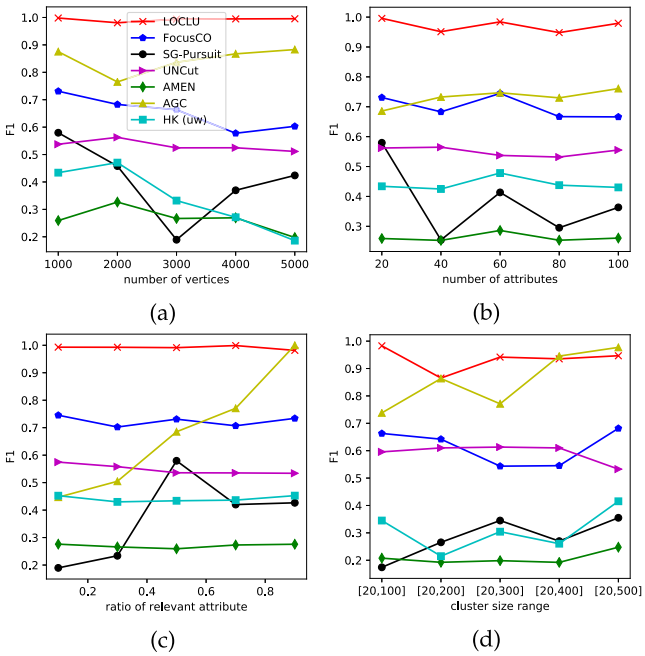


Fig. 6. Clustering results (F_1) on synthetic graphs.

we define a block for the cluster on the diagonal of the adjacency matrix and randomly asign a 1 (an edge) for each entry in the block with probability 0.35 (density of edges in each cluster). For the blocks that are not on the diagonal of the adjacency matrix, we randomly assign an edge for each entry in the block with a probability of 0.01 (density of edges between clusters). We further bisect each graph cluster into two new clusters and then assign attributes to each new cluster. In this case, a method that is only applicable for graph structure cannot detect the "real cluster" (unimodal both in the graph structure and designated attributes). To add vertex attributes, for each new graph cluster, we generate the values of relevant attributes according to a Gaussian distribution with the mean value of each attribute randomly sampled from the range $[0, 10]$, and the variance value of each attribute 0.001. Following FocusCO [10], the variance is specifically chosen to be small such that the clustered vertices "agree" on their relevant attributes. To make the other attributes of clusters irrelevant to the graph structure, we first randomly permute the vertex labels and then generate each cluster's irrelevant attribute values according to a Gaussian distribution with mean randomly sampled from the range $[10, 20]$ and variance 1.

To study how the number of designated attributes affect the performance of LOCLU, we use our generative model to generate a synthetic data with $n = 1000$ vertices, $d = 20$ attributes, and the ratio of relevant attributes 50 percent. Fig. 5 shows that LOCLU can almost detect the ground truth. When increasing the number of designated attributes, the performance of LOCLU does not change much. LOCLU considers the data points that situate in the modal interval as the cluster. However, for some boundary data points of Gaussian clusters, the modal interval may not include them. This is the reason why the performance curves of LOCLU have some small vibrations.

Figs. 6 and 7 show the quality results. Since HK(w) and HK (uw) have similar performance, we only show one of

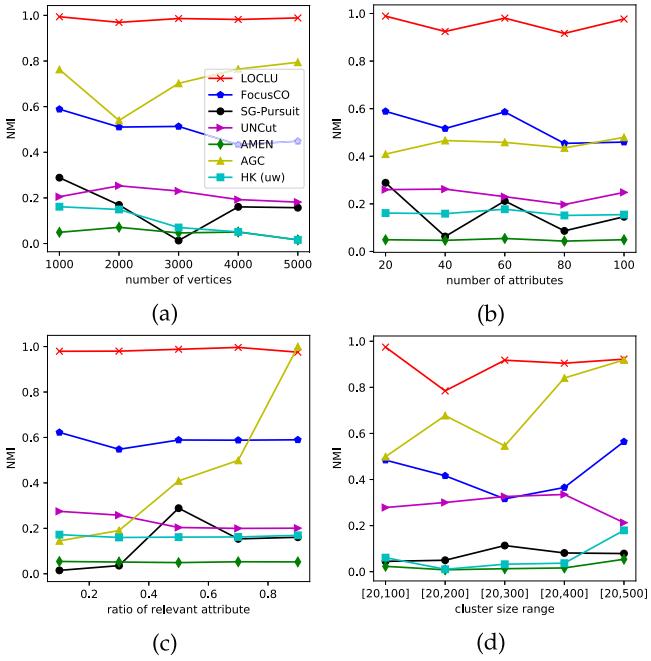


Fig. 7. Clustering results (NMI) on synthetic graphs.

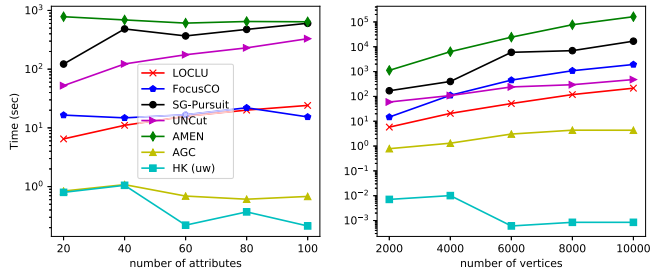


Fig. 8. Runtime experiments.

TABLE 1
The Statistics of the Real-World Attributed Graphs

Datasets	vertex#	edge#	attribute#	cluster#[3], [5]
DISNEY	124	333	28	9
4AREA	26,144	108,550	4	50
ARXIV	856	2,660	30	19
IMDB	862	4,388	21	30
ENRON	13,533	176,967	18	40
PATENTS	100,000	188,631	5	150

them. Fig. 6d shows the results of each method when varying the cluster size range. We let the graph contains clusters with variable sizes and increase the variance of the cluster sizes. The cluster size is randomly drawn from the variable ranges. In Fig. 6, we can see that LOCLU outperforms the most comparison methods. In most cases, LOCLU beats all the competitors with a large margin, although we provide them with the correct parameters. Fig. 6 also shows that SG-Pursuit is the most unstable method compared with the other methods in all these scenarios. AGC is a deep learning method. We can see that AGC is the best in all the comparison methods. Fig. 6c demonstrates that the performance of AGC is dramatically increasing with the increasing ratio of relevant attribute. In Fig. 7, we have similar conclusions. As pointed out above, for some boundary data points of Gaussian clusters, the modal interval may not include them. Thus, the curves of LOCLU has some small vibrations. In addition, we randomly generate the mean values of the attributes of the graph cluster. If the mean values of the attributes of two graph clusters are very close, the dip test may think these two clusters' attributes follow a unimodal distribution. Therefore, LOCLU cannot separate them. This is another reason that the curves of LOCLU have some small vibrations.

4.2.2 Scalability

In this section, we study the scalability of all the methods. We still use the generative model to generate synthetic graphs. For the case of varying the number of attributes, we fix the number of vertices $n = 2000$ and the ratio of the relevant attributes 50 percent. For the case of varying the number of vertices, we fix the attribute dimension $d = 20$ and the ratio of the relevant attributes 50 percent. Because the running time of the weighted and unweighted versions of the baseline HK are similar, we only give the results of the unweighted version. The runtime of each method is demonstrated in Fig. 8. Since HK (uw) only considers the graph structure, its running time is the lowest. AGC has the second lowest running time. LOCLU outperforms FocusCO, SG-Pursuit, UNCut, and AMEN in most cases.

4.3 Real-World Graphs

We conduct experiments on six real-world attributed graphs whose statistics are given in Table 1. Their details are described in the following. For the real-world graphs, if their attributes are not numeric, i.e., categorical, we use one-hot encoding to tranform the categorical values to numeric ones.

- DISNEY [26]: This network is the Amazon co-purchase network of Disney movies. The network has 124 vertices and 333 edges. Vertices represent movies and edges represent their co-purchase relationships. Each movie has 28 attributes.
- 4AREA [10]: This network is a co-authorship network of computer science authors. The attributes represent the relevance scores of the publications of an author to the conferences. The categories of conferences are "databases", "data mining", "information retrieval", and "machine learning". The network has 26,114 vertices and 108,550 edges.
- ARXIV [3]: This network is a citation network whose vertices represent papers and edges represent citation relationships. Attributes denote how often a specific keyword appears in the abstract of the paper. The network has 856 vertices, 2,660 edges, and 30 attributes.
- IMDB [3]: This network is extracted from Internet Movie Database. Each vertex represents a movie with at least 200 rankings and an average ranking of at least 6.5. Two movies are connected if they have the same actors. Attributes denote 21 movie genres. The network has 862 vertices and 4,388 edges.

TABLE 2
The AU/GU/Compactness Scores of Each Method on the Real-World Attributed Graphs

Algorithms	Disney	4Area	ARXIV	IMDb	Enron	Patents
LOCLU	**0.010**/0.062/**0.072**	**0.002**/**0.009**/**0.011**	**0**/**0.027**/**0.027**	**0**/**0.015**/**0.015**	**0.006**/**0.001**/**0.007**	**0**/**0.001**/**0.001**
FocusCO	0.012/0.080/0.092	0.021/0.075/0.096	0.074/0.055/0.129	0.024/0.079/0.103	0.088/0.001/0.089	0.012/0.067/0.079
SG-Pursuit	0.088/0.087/0.167	N/A	0.075/0.074/0.149	0.051/0.058/0.109	0.126/0.001/0.127	N/A
UNCut	0.094/0.079/0.173	0.175/**0.009**/0.184	0.172/0.051/0.223	0.148 / 0.023 / 0.171	0.149/0/0.149	0.162/0.012/0.174
AMEN	0.071/**0.055**/0.126	0.022/0.033/0.055	N/A	0.106/0.060/0.166	N/A	N/A
AGC	0.138/0.094/0.232	0.008/0.011/0.019	0.138/0.057/0.195	0.132/0.029/0.161	0.156/0.001/0.157	0.007/0.007/0.014
HK (uw)	0.123/0.070/0.193	0.050/0.078/0.128	0.168/0.050/0.218	0.143/0.019/0.162	0.167/0.002/0.169	0.027/0.031/0.058
HK (w)	0.123/0.070/0.193	0.050/0.078/0.128	0.168/0.050/0.218	0.143/0.019/0.162	0.167/0.002/0.169	0.027/0.031/0.058

N/A means the results are not available because the method: 1) is not applicable on the unconnected graphs, 2) runs out of memory, or 3) does not finish in a week.

- Enron [26]: This network is the communication network with email transmission as edges between email addresses. Each vertex has 18 attributes which describe aggregated information about average content length, average number of recipients, or time range between two mails. The network has 13,533 vertices and 176,967 edges.
- Patents [3]: This network is a citation network of patents with 100,000 vertices, 188,631 edges, and five attributes which are "assignee code", "claims", "patent class", "year" and "country".

Since the real-world graphs do not have a ground truth, we use the proposed AU, GU, and Compactness as cluster quality measures. The lower the scores of these three measures, the higher the cluster quality. In addition to these three measures, we also report the Normality [20], [21] score. The Normality score is a generalization of Newman's modularity and assortativity [27], [36] to attributed graphs. Normality measures both the internal consistency and external separability of an attributed graph cluster. The higher the Normality score, the better the cluster quality. Note that the Normality score can be negative. We give the average scores over 50 runs in Tables 2 and 3, each time with a randomly sampled vertex as the seed vertex. For SG-Pursuit, UNCut and AGC, we give them the same number of clusters as used in [3], [5]. For each seed vertex, we first decide which cluster contains it and then compute the scores of these measures.

We can see from Table 2 that LOCLU achieves the best AU and Compactness scores on all six real-world datasets. On the dataset Disney, AMEN achieves the best GU score. Table 3 shows the Normality score of each method. We can see that LOCLU has the best Normality score on four datasets. On dataset Disney, UNCut has the best Normality score. On dataset Patents, FocusCO has the best Normality score. For case studies, we interprete the results of LOCLU and its competitor FocusCO on Disney and 4Area datasets. For FocusCO, we set the entries in β that correspond to the designated attributes to one and other entries to zero.

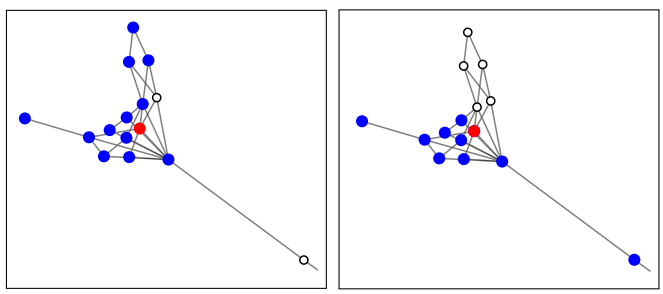
TABLE 3
The Normality Score of Each Method on the Real-World Attributed Graphs

Algorithms	Disney	4Area	ARXIV	IMDb	Enron	Patents
LOCLU	-1.326	**-0.420**	**-0.567**	**0.013**	**-0.800**	-1.001
FocusCO	-1.567	-0.760	-0.832	-0.979	-1.000	**-0.920**
SG-Pursuit	-1.898	N/A	-0.914	-0.979	-0.999	N/A
UNCut	**-1.182**	-1.000	-0.999	-0.996	-1.000	-1.001
AMEN	-2.403	-0.974	N/A	-0.990	N/A	N/A
AGC	-1.235	-1.000	-0.987	-0.998	-1.000	-1.000
HK (uw)	-1.687	-0.780	-0.858	-0.958	-0.926	-1.001
HK (w)	-1.687	-0.780	-0.858	-0.958	-0.926	-1.001

N/A means the results are not available because the method: 1) is not applicable on the unconnected graphs, 2) runs out of memory, or 3) does not finish in a week.

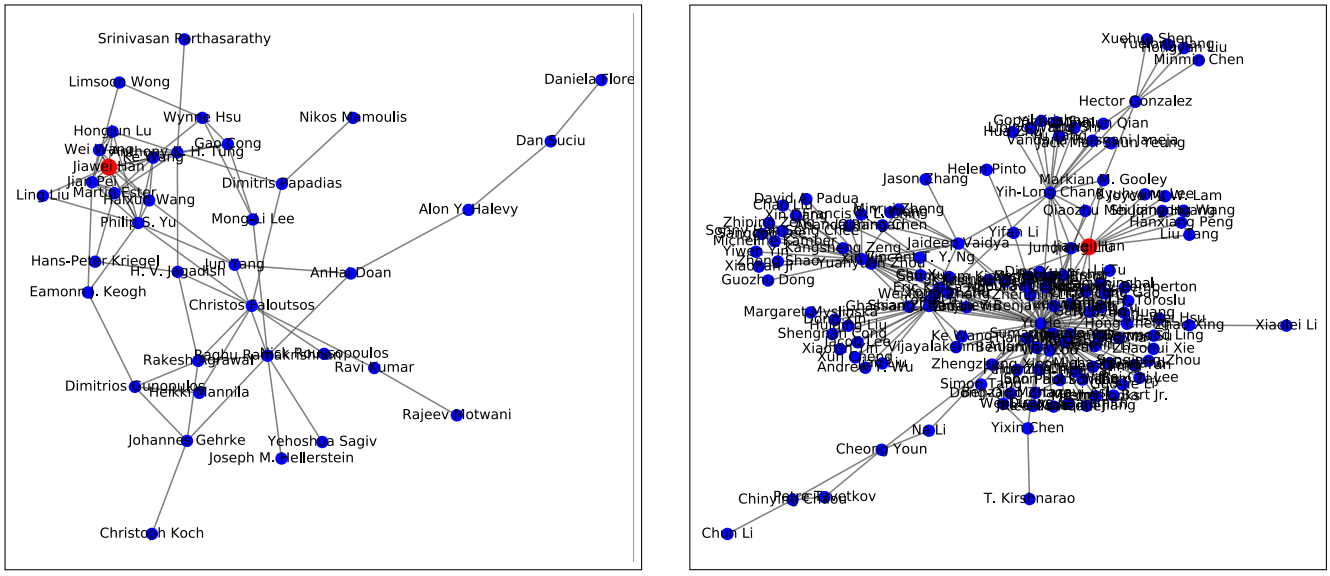
Disney. Disney is a subgraph of the Amazon co-purchase network. Each movie (vertex) is described by 28 attributes, such as "Average Vote", "Product Group", and "Price". Given the seed vertex and one designated attributes "Amazon Price", we want to find a local cluster concentrating on this seed vertex and the designated attribute. All the 15 vertices in Fig. 9 are read-along movies that are rated as PG (Parental Guidance Suggested) and attributed as "Action & Adventure", e.g., "Spy Kids", "Inspector Gadget" and "Mighty Joe Young". We show the local clusters detected by LOCLU and its competitor FocusCO in Fig. 9. In Fig. 9, the vertex in red is the given seed vertex, and the vertices in blue are the detected vertices. Fig. 9a shows the local cluster detected by LOCLU. The GU score is 0.087 and the AU score is 0.110. The Compactness score is 0.197. The Normality score is -1.454. Fig. 9b shows the local cluster detected by FocusCO. The GU score is 0.050 and the AU score is 0.100. The Compactness score is 0.150. The Normality score is -1.711. FocusCO is better than LOCLU if considering the Compactness score. LOCLU is superior to FocusCO when considering the Normality score.

4Area. 4Area is a co-authorship network of computer science authors. The attributes represent the relevance scores of the publications of an author to the conferences "databases", "data mining", "information retrieval", and "machine learning". Given the seed vertex *Jiawei Han* and two attributes "data mining" and "machine learning", we



(a) LOCLU (b) FocusCO

Fig. 9. Local clusters found in the Disney dataset by LOCLU and FocusCO.



(a) LOCLU (b) FocusCO

Fig. 10. Local clusters found in the 4AREA dataset by LOCLU and FocusCO. To reduce clutter, we only show a subgraph of 4AREA, which consists of the seed vertex and the detected vertices.

want to find a local cluster concentrating on this seed vertex and the two designated attributes. Fig. 10a shows the main component of the local cluster that includes the given seed vertex (in red) and the detected vertices (in blue) by LOCLU. This subgraph has 37 authors and is unimodal in the two designated attributes "data mining" and "machine learning". This subgraph contains authors that belong to the "data mining" field, but not the "machine learning" field. It contains authors such as *Jian Pei*, *Philip S. Yu*, *Hans-Peter Kriegel*, and *Christos Faloutsos* who focus primarily on "data Mining". The GU score is 0.046 and the AU score is 0.052. The COMPACTNESS score is 0.098. The NORMALITY score is -1.424. The detected cluster (shown in Fig. 10b) by FocusCO has 134 authors. This local cluster is not unimodal in the designated attribute "data mining". The cluster contains authors such as *Chu Xu* and *Liping Wang* who focus primarily on "information retrieval", and authors such as *Joseph C. Pemberton* and *Zhao Xing* who focus primarily on "machine learning". The GU score is 0.018 and the AU score is 0.110. The COMPACTNESS score is 0.128. The NORMALITY score is -2.000. Thus, the subgraph shown in Fig. 10a has a higher quality than that shown in Fig. 10b.

5 RELATED WORK AND DISCUSSION

5.1 Plain Graph Clustering

Plain graphs are those graphs whose vertices have no attributes. Clustering on plain graphs has been well studied in literatures. METIS [28] and spectral clustering [15], [29], [30] are typically and widely used methods, which compute a k-way partitioning of a graph. METIS is a multi-constraint graph partitioning method, which are based on the multilevel graph partitioning paradigm. Spectral clustering aims to partition the graph into k subgraphs such that the normalized cut criterion is minimized. Instead of optimizing the normalized cut criterion, MODULE [27] optimizes a quality function known as "modularity" over the possible divisions of a graph. The authors showed that the modularity was superior to the normalized cut criterion in the task of community detection. Markov Cluster Algorithm (MCL) [31] is a fast and scalable graph clustering method that is based on simulation of stochastic flow in graphs. Infomap [32] is an information theoretic approach that uses the probability flow of random walks on a network as a proxy for information flows and decomposes the network into modules by Minimum Description Length (MDL) principle. Attractor [33] automatically detects communities in a network by using the concept of distance dynamics, i.e., the network is treated as an adaptive dynamical system where each vertex interacts with its neighbors. Cluster-driven Low-rank Matrix Completion (CLMC) [51] performs community detection and link prediction simultaneously. It first decomposes the adjacency matrix of a graph as three additive matrices: clustering matrix, noise matrix and supplement matrix. Then, the community-structure and low-rank constraints are imposed on the clustering matrix to remove noisy edges between communities.

5.2 Attributed Graph Clustering

Differing from the plain graph clustering which groups vertices only taking the graph structure into account, attributed graph clustering achieves detecting clusters in which the vertices have dense edge connectivity and homogeneous attribute values. PICS [1] exploits the MDL principle to automatically decide the parameters to detect meaningful and insightful patterns in attributed graphs. SA-Cluster [2] first designs a unified neighborhood random walk distance to measure the vertex similarity on an augmented graph. It then uses k-medoids to partition the graph into clusters with cohesive intra-cluster structures and homogeneous attribute values. BAGC [34] develops a Bayesian probabilistic model for attributed graphs. Clustering on attributed graphs is transformed into a probabilistic inference problem, which is then solveld by an efficient variational method.

The above methods consider all attributes for clustering. However, the irrelevant attributes may be contradicting with the graph structure. In this case, clusters only exist in the subset (subspace) of attributes. For the subspace clustering in attributed graphs, some methods have been proposed. SSCG [3] proposes Minimum Normalized Subspace Cut and detects an individual set of relevant features for each cluster. It needs to update the subspace dependent weight matrix in every iteration, which is very time-consuming. CDE [52] formulates the community detection in attributed graphs as a nonnegative matrix factorization problem. It first develops a structural embedding method for the graph structure. Then, it integrates community structure embedding matrix and vertex attribute matrix for subsequent nonnegative matrix factorization. CDE is only applicable on graphs with nonnegative vertex attributes.

UNCut [5] proposes unimodal normalized cut to find cohesive clusters in attributed graphs. The detected cohesive clusters have densely connected edges and have as many homogeneous (unimodal) attributes as possible. The homogeneity or unimodality of attributes is measured by the proposed unimodality compactness which also exploits Hartigans' dip test. The dip test used in UNCut is to measure the unimodality of each attribute. However, in our method LOCLU, the dip test is used to generate modal interval on which the local clustering technique is based. SG-Pursuit [4] is a generic and efficient method for detecting subspace clusters in attributed graphs. The main idea is to iteratively identify the intermediate solution that is close-to-optimal and then project it to the feasible space defined by the topological and sparsity constraints. SG-Pursuit needs to specify the parameters such as the maximum number of vertices in the subspace cluster and the maximum size of selected features which are difficult to set in the real-world datasets.

Recently, deep learning techniques are adopted for attributed graph clustering. DAEGC [35] develops a graph attention-based autoencoder to effectively integrate both structure and attribute information for deep latent representation learning. Furthermore, soft labels for the graph representation are generated to supervise a self-training clustering process. The graph representation and self-training processes are unified in one framework. AGC [22] is an adaptive graph convolution method for attributed graph clustering. AGC first designs a k-order graph convolution that acts as a low-pass graph filter on vertex attributes to obtain smooth feature representations. Then, it utilizes spectral clustering to find clusters in the representation space.

Another research trend is to integrate anomaly detection into the clustering process. AMEN [20], [21] proposes a new quality measure called Normality for attributed neighborhoods, which utilizes the graph structure and attributes together to quantify both internal consistency and external separability. Normality is inspired by Newman's modularity and assortativity [27], [36]. Then, a community and anomaly detection algorithm that uses Normality is proposed to extract communities and anomalies in attributed graphs. Each community is assigned with a few characterizing attributes. PAICAN [37] is a probabilistic generative model that jointly models the attribute and graph space, as well as the latent group assignments and anomaly detection. All the methods discussed above need to partition the whole graph structure to find clusters and cannot incorporate user's preference into clustering.

5.3 Semi-Supervised Graph Clustering

In many applications, people may be only interested in finding clusters near a target local region in the graph. The methods for plain graph and attributed graph clustering cannot be applied in such a scenario. Several recent methods [6], [7], [38] focus on using short random walks starting from a small seed set of vertices to find local clusters. There are also some proposals focusing on using the graph diffusion methods to find local clusters, such as PPR [8], HK [9], PGDc [24], HOSPLOC [39], and MAPPR [40]. PPR [8] is an approximate method to compute the personalized PageRank vector which is used for the local graph partitioning. HK [9] is a local and deterministic method to accurately compute a heat kernel diffusion in a graph. There are also some methods based on spectral clustering and label propagation for local cluster detection, such as [41], [42], [43], [44], [45].

However, all these methods are only applicable on the task of local clustering on plain graphs. To the best of our knowledge, there are only two methods focusing on the local clustering on attributed graphs. FocusCO [10] incorporates user's preference into graph mining and outlier detection. It identifies the relevance of vertex attributes that makes the user-provided examplar vertices similar to each other. Then it reweighs the graph edges and extracts the focused clusters. FocusCO cannot infer the projection vector if the examplar set has only one vertex. LOCLU can find a local cluster around a given seed vertex. If given a set of vertices whose designated attribute values follow a unimodal distribution, LOCLU can also work. However, if their designated attribute values follow a multimodal distribution, LOCLU cannot find a local cluster that includes all these vertices. Like other clustering methods, LOCLU also has limitations. For example, the univariate projection for the dip test may cause information-loss in some cases. TCU-SA (Target Community Detection with User's Preference and Attribute Subspace) [46] first computes the similarities between vertices and then expand the query vertex set with its neighbors. Based on the expanded set, TCU-SA deduces the attribute subspace using an entroy method. Finally, the target community is extracted. The idea is very similar to that of FocusCO.

5.4 Community Search

Community search over attributed graphs in database research field is also related to our work. Given an input set of query vertices $\mathcal{V}_q$ and their corresponding attributes, find a community containing $\mathcal{V}_q$, in which vertices are densely connected and have homogeneous attributes. These methods include [47], [48], [49], [50]. Closest truss community (CTC) [47] is proposed to find a connected k-truss subgraph that has the largest k, contains $\mathcal{V}_q$, and has the minimum diameter. The problem is NP-hard and the authors develops a greedy algorithm to find a satisfied community. attribute truss community (ATC) [48] formulates the community search on attributed graphs as finding attributed truss communities. The detected communities are connected and

close k-truss subgraphs which contains $\mathcal{V}_q$ and has the largest attribute relevance score proposed by the authors. Attributed community query (ACQ) [49], [50] develops the CL-tree index structure and three algorithms based on it for efficient attributed community search. The CL-tree is devised to organize the vertex attribute data in a hierarchical structure. The community search methods are only applicable on categorical attributes. The detected vertices have the same attribute values to those of the query vertices. For continuous attributes, they cannot search a community that is unimodal in the subspace that is composed of the designated attributes. In addition, they are based on dense subgraph structures, such as quasi-clique, k-core, or k-truss, which are not commonly used in graph clustering.

6 Conclusion

In this work, we have proposed LOCLU for incorporating user's preference into attributed graph clustering. Currently, very few methods can deal with this kind of task. To achieve the goal, we first propose a new quality measure called Compactness that measures the unimodality of both the graph structure and the subspace that is composed of the designated attributes of a local cluster. Then, we propose LOCLU to optimize the Compactness score. Empirical studies prove that our method LOCLU is superior to the state-of-the-arts. In the future, we will further explore node embeddings for attributed graphs, which seamlessly integrate information from both the attributes and graph structure.

Acknowledgments

The authors would like to thank anonymous reviewers for their constructive and helpful comments. This work was supported partially by the U.S. National Science Foundation (Grant # IIS-1817046) and by the U.S. Army Research Laboratory and the U.S. Army Research Office (Grant # W911NF-15-1-0577).

References

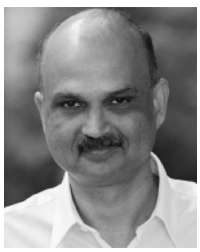
[1] L. Akoglu, H. Tong, B. Meeder, and C. Faloutsos, "PICS: Parameter-free identification of cohesive subgroups in large attributed graphs," in *Proc. SIAM Int. Conf. Data Mining*, 2012, pp. 439–450.
[2] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 718–729, 2009.
[3] S. Günnemann, I. Färber, S. Raubach, and T. Seidl, "Spectral subspace clustering for graphs with feature vectors," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 231–240.
[4] F. Chen, B. Zhou, A. Alim, and L. Zhao, "A generic framework for interesting subspace cluster detection in multi-attributed networks," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 41–50.
[5] W. Ye, L. Zhou, X. Sun, C. Plant, and C. Böhm, "Attributed graph clustering with unimodal normalized cut," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2017, pp. 601–616.
[6] R. Andersen and K. J. Lang, "Communities from seed sets," in *Proc. 15th Int. Conf. World Wide Web*, 2006, pp. 223–232.
[7] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Statistical properties of community structure in large social and information networks," in *Proc. 17th Int. Conf. World Wide Web*, 2008, pp. 695–704.
[8] R. Andersen, F. Chung, and K. Lang, "Local graph partitioning using pagerank vectors," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci.*, 2006, pp. 475–486.
[9] K. Kloster and D. F. Gleich, "Heat kernel based community detection," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 1386–1395.
[10] B. Perozzi, L. Akoglu, P. I. Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2014, pp. 1346–1355.
[11] J. A. Hartigan and P. Hartigan, "The dip test of unimodality," *Ann. Statist.*, vol. 13, pp. 70–84, 1985.
[12] S. Maurus and C. Plant, "Skinny-dip: Clustering in a sea of noise," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1055–1064.
[13] A. Krause and V. Liebscher, "Multimodal projection pursuit using the dip statistic," Universit at Greifswald, Tech. Rep. 13, 2005.
[14] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.
[15] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[16] U. Von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.
[17] D. Wagner and F. Wagner, "Between min cut and graph bisection," in *Proc. Int. Symp. Math. Found. Comput. Sci.*, 1993, pp. 744–750.
[18] F. Lin and W. W. Cohen, "Power iteration clustering," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 655–662.
[19] F. Lin and W. W. Cohen, "A very fast method for clustering big text datasets," in *Proc. 19th Eur. Conf. Artif. Intell.*, 2010, pp. 303–308.
[20] B. Perozzi and L. Akoglu, "Scalable anomaly ranking of attributed neighborhoods," in *Proc. SIAM Int. Conf. Data Mining*, 2016, pp. 207–215.
[21] B. Perozzi and L. Akoglu, "Discovering communities and anomalies in attributed graphs: Interactive visual exploration and summarization," *ACM Trans. Knowl. Discov. Data*, vol. 12, no. 2, 2018, Art. no. 24.
[22] X. Zhang, H. Liu, Q. Li, and X.-M. Wu, "Attributed graph clustering via adaptive graph convolution," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 4327–4333.
[23] C. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval," *Natural Lang. Eng.*, vol. 16, no. 1, pp. 100–103, 2010.
[24] T. Van Laarhoven and E. Marchiori, "Local network community detection with continuous optimization of conductance and weighted kernel k-means," *J. Mach. Learn. Res.*, vol. 17, no. 147, pp. 1–28, 2016.
[25] A. Condon and R. M. Karp, "Algorithms for graph partitioning on the planted partition model," *Random Struct. Algorithms*, vol. 18, no. 2, pp. 116–140, 2001.
[26] P. I. Sánchez, E. Müller, F. Laforet, F. Keller, and K. Böhm, "Statistical selection of congruent subspaces for mining attributed graphs," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 647–656.
[27] M. E. Newman, "Modularity and community structure in networks," *Proc. Nat. Academy Sci. United States America*, vol. 103, no. 23, pp. 8577–8582, 2006.
[28] G. Karypis and V. Kumar, "Multilevel algorithms for multi-constraint graph partitioning," in *Proc. ACM/IEEE Conf. Supercomput.*, 1998, pp. 1–13.
[29] A. Y. Ng *et al.*, "On spectral clustering: Analysis and an algorithm," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2001, pp. 849–856.
[30] W. Ye, S. Goebl, C. Plant, and C. Böhm, "FUSE: Full spectral clustering," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1985–1994.
[31] S. Van Dongen, "A cluster algorithm for graphs," *Rep.-Inf. Syst.*, vol. 10, no. 10, pp. 1–40, 2000.
[32] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proc. Nat. Academy Sci. United States America*, vol. 105, no. 4, pp. 1118–1123, 2008.
[33] J. Shao, Z. Han, Q. Yang, and T. Zhou, "Community detection based on distance dynamics," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1075–1084.
[34] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng, "A model-based approach to attributed graph clustering," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2012, pp. 505–516.
[35] C. Wang, S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, "Attributed graph clustering: A deep attentional embedding approach," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 3670–3676.
[36] M. E. Newman, "Assortative mixing in networks," *Phys. Rev. Lett.*, vol. 89, no. 20, 2002, Art. no. 208701.
[37] A. Bojchevski and S. Günnemann, "Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2738–2745.

[38] B. Yuchen, Y. Yaowei, C. Wei, W. Wei, L. Dongsheng, and Z. Xiang, "On multi-query local community detection," in *Proc. IEEE Int. Conf. Data Mining*, 2018, pp. 9–18.
[39] D. Zhou *et al.*, "A local algorithm for structure-preserving graph cut," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 655–664.
[40] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 555–564.
[41] K. He, Y. Sun, D. Bindel, J. Hopcroft, and Y. Li, "Detecting overlapping communities from local spectral subspaces," in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 769–774.
[42] Y. Li, K. He, D. Bindel, and J. E. Hopcroft, "Uncovering the small community structure in large networks: A local spectral approach," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 658–668.
[43] M. W. Mahoney, L. Orecchia, and N. K. Vishnoi, "A local spectral method for graphs: With applications to improving graph partitions and exploring data graphs locally," *J. Mach. Learn. Res.*, vol. 13, no. Aug., pp. 2339–2365, 2012.
[44] T. J. Hansen and M. W. Mahoney, "Semi-supervised eigenvectors for large-scale locally-biased learning," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3691–3734, 2014.
[45] J. P. Bagrow and E. M. Bollt, "Local method for detecting communities," *Phys. Rev. E*, vol. 72, no. 4, 2005, Art. no. 046108.
[46] H. Liu, H. Ma, Y. Chang, Z. Li, and W. Wu, "Target community detection with user's preference and attribute subspace," *IEEE Access*, vol. 7, pp. 46 583–46 594, 2019.
[47] X. Huang, L. V. Lakshmanan, J. X. Yu, and H. Cheng, "Approximate closest community search in networks," *Proc. VLDB Endowment*, vol. 9, no. 4, pp. 276–287, 2015.
[48] X. Huang and L. V. Lakshmanan, "Attribute-driven community search," *Proc. VLDB Endowment*, vol. 10, no. 9, pp. 949–960, 2017.
[49] Y. Fang, R. Cheng, S. Luo, and J. Hu, "Effective community search for large attributed graphs," *Proc. VLDB Endowment*, vol. 9, no. 12, pp. 1233–1244, 2016.
[50] Y. Fang, R. Cheng, Y. Chen, S. Luo, and J. Hu, "Effective and efficient attributed community search," *VLDB J.*, vol. 26, no. 6, pp. 803–828, 2017.
[51] J. Shao, Z. Zhang, Z. Yu, J. Wang, Y. Zhao, and Q. Yang, "Community detection and link prediction via cluster-driven low-rank matrix completion," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 3382–3388.
[52] Y. Li, R. Cheng, C. Sha, X. Huang, and Y. Zhang, "Community detection in attributed graphs: An embedding approach," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 338–345.

Wei Ye received the PhD degree in computer science from the Institut für Informatik, Ludwig-Maximilians-Universität München, Munich, Germany, in 2018. He is currently a postdoctoral researcher with the DYNAMO Lab, University of California, Santa Barbara. His research interests include graph-based machine learning, causal reasoning, and dynamic networks.

Dominik Mautz received the master of science degree in informatics from the Technical University of Munich, Munich, Germany, in 2016. He is currently working toward the doctoral degree and research assistant in the Research Group for Data Mining in Medicine, Ludwig-Maximilians-Universität München, Munich, Germany. His research interests include unsupervised deep learning, feature learning, dimensional reduction, and clustering.

Christian Böhm received the PhD degree, in 1998 and the habilitation degree, in 2001. He is currently a professor of computer science with Ludwig-Maximilians-Universität München, Munich, Germany. His research interests include database systems and data mining, particularly index structures for similarity search and clustering algorithms. He has received several research awards in the top-tier data mining conferences.



Ambuj Singh received the PhD degree from the University of Texas at Austin, Austin, Texas, in 1989. He is currently a professor of computer science with the University of California, Santa Barbara. He is currently on the editorial boards of three journals, and has served on program committees of several data mining conferences. His current research interests include network science, data mining, machine learning, bioinformatics, graph querying, and mining.

Claudia Plant received the PhD degree, in 2007. She is currently a professor of computer science with the University of Vienna, Vienna, Austria. Her research focuses on databases and data mining, especially clustering, information-theoretic data mining, and integrative mining of heterogeneous data. She has received several best paper awards in the top-tier data mining conferences.