Research Article

Social Network Community Detection Using Agglomerative Spectral Clustering

Ulzii-Utas Narantsatsralt and Sanggil Kang

Department of Computer Engineering, Inha University, Incheon, Republic of Korea

Correspondence should be addressed to Sanggil Kang; sgkang@inha.ac.kr

Received 18 April 2017; Revised 24 July 2017; Accepted 23 August 2017; Published 7 November 2017

Academic Editor: Katarzyna Musial

Copyright © 2017 Ulzii-Utas Narantsatsralt and Sanggil Kang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Community detection has become an increasingly popular tool for analyzing and researching complex networks. Many methods have been proposed for accurate community detection, and one of them is spectral clustering. Most spectral clustering algorithms have been implemented on artificial networks, and accuracy of the community detection is still unsatisfactory. Therefore, this paper proposes an agglomerative spectral clustering method with conductance and edge weights. In this method, the most similar nodes are agglomerated based on eigenvector space and edge weights. In addition, the conductance is used to identify densely connected clusters while agglomerating. The proposed method shows improved performance in related works and proves to be efficient for real life complex networks from experiments.

1. Introduction

In recent years, community detection in a network has become one of the main topics of fields, such as biology, computer science, physics, and applied mathematics [1–3]. In a network, $G(V, E)$, where V is a set of nodes and E is the edges (relation) between the nodes, a community is a group of nodes with tightly connected edges with each other and the nodes of community show similar characteristics. For example, in social network, people in a community show similar interest to a trend in a community, for example, buying the same products in online marketing. In a biology network, proteins in a community show similar specific functions, and, in the World Wide Web, sites clustered together show the same topic in their web page. Scientists in many fields made significant contributions to detecting communities by a number of different methods, such as graph partitioning [4, 5], hierarchical clustering [6, 7], and spectral clustering [8, 9].

In graph partitioning, a network is divided into clusters in such a way that the number of edges connecting the clusters is minimum, that is, the edges of a cluster are denser

inside than outside (also referred as conductance [10]). In addition, the number of lowest sized clusters needs to be specified. Girvan and Newman [3] introduced a popular graph partitioning algorithm. Girvan and Newman [3] use modularity (also referred as conductance) to cluster communities but the method is slower than other community detection algorithms [11, 12]. Later, Djidjev [13] proposed a computationally faster version of the algorithm. However, the definition of conductance is not always definite, and the definition can be false in some cases [5]. Therefore, graph partitioning still needs further inference. A number of methods have been proposed to solve this problem. One of the famous methods is introduced by Newman [5]. They use spectral clustering algorithm with modularity maximization, in which the modularity function is implemented for only possible clusters of network and the result proved that spectral clustering with conductance can efficiently cluster communities.

Hierarchical clustering is used for complex networks because they often have a hierarchical structure [14]. Hierarchical clustering consists of a division [15] and agglomeration stage [16, 17]. In the division stage, a network is

deemed to be one cluster in the beginning and the network is then divided into clusters in each iteration, where the most dissimilar nodes are separated. In the agglomeration stage, similar nodes are agglomerated together until the termination criteria are met or the clusters agglomerate into a single community. However, hierarchical clustering needs a well-defined similarity function and the clustering can be inaccurate if all nodes are similar to each other.

However, the problem of similar nodes and similarity function can be solved by projecting the nodes into high dimensional feature space using spectral clustering because the projected eigenvectors significantly distinguish the similar nodes into more distanced positions in feature space. The reason for using eigenvector space instead of using original point is that the properties of original clustering are made more distinct by the eigenvector space. In spectral clustering, original points are transformed into a set of points in eigenvector space and clustering is done by analyzing eigenvector space. One technique for clustering eigenvector space is to use k -means algorithm [18] where similar nodes are clustered together. However, traditional spectral clustering has a problem with model selection which depends on heuristics. The problem can be solved using weighted kernel spectral clustering (KSC) with primal and dual representations [19–21]. KSC [21] focuses on the principle that projections of similar nodes are clustered together in eigenvector space. In another work of KSC, an agglomeration technique is introduced to the KSC which is called agglomerative hierarchical kernel spectral clustering (AH-KSC) [22]. AH-KSC uses eigenvector space to find distance between nodes and it agglomerates close distanced nodes. The main purpose of AH-KSC is to get hierarchical clustering but accuracy of AH-KSC does not improve significantly from KSC because AH-KSC allows indirectly connected nodes to be agglomerated together and also there are no termination criteria for satisfied community. The problems of KSC and AH-KSC are choosing eigenvector, improving accuracy of detected communities, and using only data generated by hand which usually do not show same characteristics as real life networks.

The above-mentioned methods focus on decreasing the computation time or improving the accuracy of community detection. Methods for improving the computational time have been well studied and it can be solved by advances in technology and techniques [23–25], such as parallel computing and GPU programming. Improving the accuracy of community detection has been challenging task because networks are usually structured with great complexity with millions of nodes and edges. Hence, this paper proposes an agglomerative spectral clustering method with conductance and edge weights to improve the accuracy of community detection. The characteristics of the proposed method are well suited for accurate community detection in complex networks because the eigenvector space from spectral clustering provides well distinct points that are used for the similarity function in agglomeration. The conductance is used for the sensitive termination criteria of agglomeration and the edge weight is a major factor for evaluating a more accurate similarity. In addition, performance of the proposed

method was compared with that of AH-KSC and KSC using real life social network data with a ground-truth, which are the LiveJournal and Orkut network. This method can help improve the community detection performance from previous works [21, 22].

The remainder of this paper is as follows: Section 2 introduces the problem statement and background, which helps understand the proposed method. The core algorithm of the proposed method is explained in Section 3. The experiment is outlined in Section 4 and Section 5 reports the conclusions.

2. Fundamental Concepts

2.1. Problem Statement. In KSC, the data are divided into training, validating, and test sets. In the training stage, the eigenvector space of the training data is signed, which is used for clustering the nodes in a network. The sign of the eigenvector points in the same cluster is identical. In the validating stage, model selection is performed to identify the clustering parameters. The eigenvector space of the test data is used to evaluate the clusters obtained from the training data using the hamming distance function. The problem with the KSC is that clustering depends on encoding/decoding eigenvectors space. The encoded values are all signed in KSC [21] and distinction between the two elements of the eigenvector is only “1” or “0” so that similar eigenvector points become noisy. For example, if in eigenvector space, $e_1^{(l)} = 0.001$ and $e_2^{(l)} = -0.001$, the two values can be binarized as “1” and “0,” respectively. Although, $e_1^{(l)}$ and $e_2^{(l)}$ are projected in similar feature space, the results of encoding show a different outcome. This problem can be solved by agglomerative hierarchical KSC (AH-KSC). In AH-KSC [22], instead of signing eigenvector space, the space is used as the data points to obtain the distance between nodes in a network and close distanced nodes are agglomerated together until there are only k clusters or less.

KSC and AH-KSC still have certain disadvantages. Both methods calculate the kernel matrix Ω by counting the number of edges connecting the common neighbors between two nodes, $i \wedge j$: $\Omega_{ij} = \sum_{k,l \in N_{ij}} A_{kl}$, where N_{ij} is a set of common neighbors of $i \wedge j$, k and l are common neighbors, and A_{kl} is adjacency matrix of the graph. However, the common neighbors between nodes can cause indirectly connected nodes to be clustered together so that the nodes in different clusters can be clustered. To solve this problem, the adjacency matrix is used as a kernel matrix so agglomerated nodes can be connected directly. In addition, KSC and AH-KSC use only the first $k - 1$ eigenvectors for encoding/decoding but the remaining eigenvectors can still provide correlated information for clustering. To take this into consideration, in this study, all eigenvector space was used to evaluate the similarity between nodes. Furthermore, in AH-KSC, there were no termination criteria for agglomerating clusters. Therefore, the conductance was used as termination criterion during the agglomeration of satisfied clusters.

2.2. Background. In general, KSC is described by a primal-dual formulation. Given a network, $G(V, E)$, where V denotes the vertices and E the edges, and the training data $V_{tr} = \{x_{tr}\}_{i=1}^{N_{tr}}$, the primal problem [21] is

$$\begin{aligned} \min_{w^{(l)}, e^{(l)}, b_l} \quad & \frac{1}{2} \sum_{l=1}^{k-1} w^{(l)T} w^{(l)} - \frac{1}{2} \sum_{l=1}^{k-1} \gamma_l e^{(l)T} D_{\Omega}^{-1} e^{(l)} \\ \text{subject} \quad & e^{(l)} = \Phi w^{(l)} + b_l 1_{N_{tr}}, \quad l = 1, \dots, k-1, \end{aligned} \quad (1)$$

where $e^{(l)} = [e_1^{(l)}, \dots, e_{N_{tr}}^{(l)}]$ is the projection, which is the mapped points of training data in feature space with respect to the direction, $w^{(l)}$, l indicates the number of score variables, which is needed to encode the k clusters, D_{Ω}^{-1} is the inverse matrix of the degree matrix of the kernel matrix, Ω , Φ is the $N_{tr} \times d_h$ feature matrix, where $\Phi = [\varphi(x_1)^T; \dots; \varphi(x_{N_{tr}})^T]$, γ_l is the regularization constant, and $1_{N_{tr}}$ is the $N_{tr} \times N_{tr}$ matrix of ones. The primal form of the data point is expressed as

$$e_i^{(l)} = w^{(l)T} \varphi(x_i) + b_l, \quad i = 1, \dots, N_{tr}, \quad (2)$$

where $\varphi : R^n \rightarrow R^{d_h}$ is the map to high dimensional feature space, where n is the number of nodes in graph, G , d_h is the number of eigenvectors, and b_l is the bias term. The dual problem related to the primal problem is

$$D_{\Omega}^{-1} M_D \Omega \alpha^l = \lambda_l \alpha^{(l)}, \quad (3)$$

where Ω is the kernel matrix with ij th entry, $\Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$; D_{Ω} is the diagonal matrix of Ω with elements of $d_{ii} = \sum_j \Omega_{ij}$; M_D is a centering matrix defined as $M_D = I_{N_{tr}} - (1/1_{N_{tr}}^T D_{\Omega}^{-1} 1_{N_{tr}})(1_{N_{tr}} 1_{N_{tr}}^T D_{\Omega}^{-1})$, where $I_{N_{tr}}$ is the $N_{tr} \times N_{tr}$ identity matrix; $\alpha^{(l)}$ is the dual variable; and the kernel function K is the similarity function of the graph. The parameters, such as the number of community k , are estimated using the training data, V_{tr} , and validating data, V_{va} . In addition, all the nodes are clustered in the training and validating stage. The eigenvector space is used to find unique code-word for all clusters A_p , $p = 1, \dots, k$. The codebook, $C = \{c_p\}_{p=1}^k$, can be obtained from rows of the binarized eigenvector matrix. Finally, the eigenvector space of the test data, V_{te} , is decoded using the hamming distance [21] and the clustered result is evaluated. Therefore, eigenvector space is used to derive the similarity among nodes, which will be explained in detail in the following section.

3. Proposed Community Detection Algorithm

This section presents details of agglomerative spectral clustering with the conductivity method. The eigenvector space is used to find the similarity among nodes and agglomerate the most similar nodes to make a new combined node in a network graph. The new combined node is added to the graph after agglomeration and the changed graph is iterated until the termination criteria are satisfied.

To agglomerate two nodes, a similarity function is modified from the correlation distance function among the nodes as follows:

$$(i, j) = \text{CorDis} = 1 - p = 1 - \frac{\sum_{n=1}^N (x_n^i * x_n^j) - (1/N) \sum_{n=1}^N x_n^i \sum_{n=1}^N x_n^j}{\sqrt{\sum_{n=1}^N (x_n^i)^2 - (1/n) \left(\sum_{n=1}^N x_n^i\right)^2} \sqrt{\sum_{n=1}^N (x_n^j)^2 - (1/n) \left(\sum_{n=1}^N x_n^j\right)^2}}, \quad (4)$$

where (i, j) is the similarity between nodes i and j in the range of $[0, 2]$ with 0 being perfect similarity and 2 being perfect dissimilarity. x_n^i is the value of the eigenvector, $e_i^{(n)}$, in eigenvector space, i th row, and n th column.

The eigenvector space is not enough to fully express the similarity among agglomerated nodes because the nodes connected to each other are projected into a similar place in feature space and it is very difficult to distinguish similar projections. On the other hand, these similar projections can be distinguished using the disparity of the edge connections between the nodes. Agglomerated nodes can have more than one connecting edge with each other. Therefore, more tightly connected nodes have more similarity. For example, in Figure 1, similar nodes are combined in the 1st iteration and node n_6 has two connections to the agglomerated node of n_4

and n_5 and has one connection to the agglomerated node of n_7 and n_8 , so that n_6 is more likely to agglomerate with n_5 and n_4 . In the 2nd iteration, new agglomerated nodes are used to find new eigenvector space and agglomerate similar nodes. In doing so, the number of edges in the graph is unchanged and some nodes have more than one edge between them. As mentioned in the example of node n_6 , the edges between two nodes are used as a mean to give a similarity score between nodes to improve the accuracy of the algorithm. On the other hand, the number of edges between nodes can be varied too much and the value of the similarity function in (4) is too different. Therefore, the similarity function will be overemphasized on a number of edges; that is, disregard the eigenvector space score. In the present study, a sigmoid function is used to normalize the edge values to solve the above-mentioned problem.

FIGURE 1: Example of agglomerating nodes.

Equation (4) is modified, as expressed in

$$(i, j) = \frac{\left(1 - \left(\left(\sum_{n=1}^N (x_n^i * x_n^j) - (1/N) \sum_{n=1}^N x_n^i \sum_{n=1}^N x_n^j \right) / \sqrt{\sum_{n=1}^N (x_n^i)^2 - (1/n) \left(\sum_{n=1}^N x_n^i \right)^2} \sqrt{\sum_{n=1}^N (x_n^j)^2 - (1/n) \left(\sum_{n=1}^N x_n^j \right)^2} \right) \right)}{0.5 / (1 + e^{((-5/E_{\max}) * E(i, j))})}, \quad (5)$$

where $E_{\max}$ is the maximum number of edges and $E(i, j)$ is the number of edges between nodes $i \wedge j$. The vertical value of the sigmoid graph is deemed to be the edge similarity score and starts from 0.5 to 1 while the horizontal value, which is the number of edges, ranges from 0 to the maximum number of edges. Equation (5) is used to find the most similar node of node i from the other nodes in graph G . At the first iteration, the first node becomes a candidate and if there is a more similar node than the candidate, the candidate is then replaced with it. The process continues until the similarity of all nodes is evaluated. Thus, the most similar node to node i is determined by

$$n_{ms}^i = \min_k (i, k), \quad (6)$$

where $k \in N$, N is all the nodes in graph G , and n_{ms}^i is the most similar node of node i .

Furthermore, to obtain a more accurate clustering result, this study considers the definition of a good community,

which is “density of the edge connection should be higher inside than outside” [10]. The similar nodes are agglomerated together in every iteration and the agglomerated nodes become a clustered community after a few iterations. If the cluster is connected tightly inside and sparsely connected to outside, there is no need for further agglomeration because the cluster is sufficiently satisfied to be a good community and agglomeration for this community is terminated. In addition, two communities are agglomerated when they are tightly connected to each other. For example, in Figure 1(c), where the inside edges are straight lines and the outside edges are dotted lines, the graph is clustered into three agglomerated communities, such as C_1 , C_2 , and C_3 . In the case of community C_1 , it has three inside edges and two outside edges connected to both C_2 and C_3 so that C_1 has a denser connection inside than outside. Consequently, no further agglomeration is needed. To consider the ratio of the inside and outside edges into (5), the two possible cases are

divided when node j is a candidate as the most similar node for node i :

- (1) $N_i < N_j$
- (2) $N_i > N_j$

where N_i is the number of nodes inside node i and N_j is the number of nodes inside node j .

In the first case, the number of inside edges of node i is at most equal to the number of outside edges connecting to node j . However in the next case, the number of inside edges of node i is more than the number of outside edges connecting to node j . Therefore, to agglomerate only tightly connected nodes together, (5) can be modified using the inside and outside edges:

$$n_{ms}^i = \min_k(i, k) \longrightarrow \begin{cases} E_i < E_{ij} * \mu, N_i < N_j \\ E_j < E_{ij} * \mu, N_i > N_j \end{cases}, \quad (7)$$

where E_i is the inside edges of node i , E_j is the inside edges of node j , E_{ij} is the edges connecting node $i \wedge j$, and μ is the community density parameter.

After finding the similarity between nodes using (7), agglomeration of the most similar nodes starts. In every iteration, the most similar node for each node was found and if the most similar node of n_i is n_j , the opposite is not definite for n_j . Therefore, only the case in which both nodes choose each other is agglomerated as the most similar node. Thus, the agglomerated node n_{ag} is

$$n_{ag} = n_i n_j \longrightarrow \begin{cases} n_{ms}^i = j \\ n_{ms}^j = i \end{cases}, \quad (8)$$

where $k \in N$ and N is all the nodes in graph G .

The termination condition is met when all agglomerated nodes are connected more tightly inside than outside as seen in Pseudocode 1.

4. Experiment

This section presents the results of the proposed method and compares the data with that of conventional community detection works [21, 22] by varying the value of parameters. LiveJournal and Orkut are used for evaluation as the ground-truth social network. LiveJournal is a blogging and social networking site that has been around since 1999. LiveJournal data has 4 million nodes and 35 million edges. The LiveJournal ground-truth data has 287,512 communities. In order to show the change of detected community by varying the density parameter for different networks, we also use Orkut network because the density difference of two networks can clearly emphasize the importance of choosing optimal density parameter. Orkut is a free online social network where users form friendship with each other. Orkut data has 3 million nodes and 117 million edges. The Orkut ground-truth data has 6,288,363 communities. The network is massive and complex, which makes more difficult clustering task. The dataset is available at <https://snap.stanford.edu/data/>.

Input: Graph G , Nodes V , Edges E , density parameter μ
Output: Hierarchically clustered communities

- (1) Find the eigenvectors $\alpha^l = [\alpha_1^{(l)}, \dots, \alpha_{N_{\mu}}^{(l)}]$ of $D_{\Omega}^{-1} M_D \Omega$
- (2) Find the similarities of each node i and j using eigenvectors with Eq. (5)
- (3) Compute the most similar node i using Eq. (7)
- (4) Agglomerate the node i and j if the two nodes chose each other as the most similar node
- (5) Re-initialize the graph with the agglomerated nodes and start the next iteration
- (6) Agglomerate the nodes into hierarchical clusters when the iteration is finished

PSEUDOCODE 1

The evaluation is done using the measurement metrics, such as precision (P), recall (R), and F -score.

$$P = \frac{T_p}{T_p + F_p}, \quad (9)$$

where T_p is the number of nodes that are correctly clustered and F_p is the number of nodes that are falsely clustered.

$$R = \frac{T_p}{T_p + F_n}, \quad (10)$$

where F_n is the number of nodes that are supposed to be clustered but failed to do so.

$$Fscore = \frac{2 * P * R}{P + R}, \quad (11)$$

where $Fscore$ is the harmonic mean of precision and recall.

The results of clustering are shown by varying the value of μ , which is the community density parameter in Figure 2, where there are 3 communities. The right side community is the smallest with 61 nodes, the middle community has 109 nodes, and the left side community is the largest community with 331 communities. An optimal value was evaluated from the training data by a trial-and-error method. Beginning with $\mu = 1$, as shown in Figure 2(a), each community is clustered into small sized clusters internally. The clustering performance is improved by increasing the value of μ up to $\mu = 3$. The middle and right side communities are clustered successfully but the largest community on the left side failed to cluster because the left side community is very complex with many edges. In this case, the clustering performance can be improved by relaxing μ . The three communities are successfully clustered when $\mu = 4$, as shown in Figure 2(c). If the value of μ continues to increase, the clustering performance becomes worse than $\mu = 4$ because the clustering criteria are too relaxed as μ increases. With $\mu =$

FIGURE 2: Varying the results of the density parameter μ to obtain the optimal value.

5, the left side community is clustered into 4 communities, as shown in Figure 2(d), and when the value of μ reaches 7, the left side community is separated into smaller communities, as shown in Figure 2(e).

Figure 3 shows comparison by varying values of density parameter μ of Orkut network. Unlike LiveJournal network, Orkut is a more densely clustered network where the ratio of node and edges of LiveJournal is 1:8.6 whereas Orkut is 1:38.1. Therefore, the density parameter of Orkut requires being more strict compared to LiveJournal because the clusters are all densely clustered with each other. If the density parameter is not strict, it will allow the densely clustered nodes to be agglomerated together. First column of Figure 3 shows the ground-truth community which is colored in yellow and the following columns are detected networks by varying density parameters ranging from 0.1 to 4. As shown in the first row of Figure 3, the detected community with density parameter 0.1 shows well desired result but the accuracy has sharply dropped when we increased the density parameter because the detected community's size has

continuously increased. The second row of Figure 3 shows different characteristics compared to the first row where the seed node has agglomerated into different cluster due to the relaxed density parameter. The third row network has similar characteristics to the second row network which shows that relaxed density parameter could lead to less densely clustered community. Fourth row network has similar results to the first row which shows that if we allow relaxed density parameter, the network will continue to expand. The optimal result of community detection has been obtained with 0.1 in Orkut network while the optimal value is 4 in LiveJournal network. Therefore, the experiment result shows that the density parameter is closely related to density of the network where the denser network requires stricter density parameter. It means that when evaluating the optimal value of density parameter, the density of the network should be considered.

Figure 4 shows the results of an analysis of the agglomeration process of the proposed method with $\mu = 4$ and AH-KSC as the number of iterations increase. Figure 4(a) shows the early stage agglomeration result of middle community

FIGURE 3: Comparison by varying density parameters on Orkut network.

FIGURE 4: Analysis of agglomeration process. (a, b, c) AH-KSC method. (d, e, f) Proposed method.

(at the 17th iteration). The early stage of AH-KSC was successfully clustering colored in red but in the halfway stage (at 26th iteration), the middle community was agglomerated together with some nodes that were included in the left side community, even though there were no direct connections to the nodes. In the late stage of agglomeration process (at the 30th iteration), the middle community was clustered with the right side community even though the right side community was the satisfied community, as shown in Figure 4(c). Figure 4(d) is the early stage of the proposed method. Like AH-KSC, the agglomeration process of the middle community is well clustered (at the 23rd iteration). In the halfway stage of agglomeration (at the 26th iteration), the middle community was clustered successfully because only directly connected nodes are agglomerated according to (5), where the number of edges between the nodes is added to the similarity function. At the late stage (at the 39th iteration), the right side community is clustered accurately because the ratio of the inner and outer edge connection is applied to (7) so that the smallest community on the right side has stopped agglomerating. Figure 4(f) shows the final result of the algorithm.

This study compared the accuracy of detection of the proposed method with AH-KSC and KSC using the ground-truth LiveJournal network. To show the comparison conveniently, only four parts of the network are used because the network is too large, that is, more than 4 million nodes. As shown in Figure 5, there are four subnetworks with different structures. The 1st network has 292 nodes and 1858 edges, and ground-truth community, to which the seed node belongs, has 24 nodes. The second network has 356 nodes and 33616 edges with a ground-truth community of 52 nodes. The third network has 652 nodes and 63044 edges, and the ground-truth community has 22 nodes. The last network has 119 nodes and 866 edges with a ground-truth community of 15 nodes. The 2nd and 3rd networks are so complex that it is difficult to detect communities while the 1st and 4th networks are well structured, that is, average in difficulty. In Figure 5, the light yellow colored node groups in the first column

TABLE 1: Overall accuracy of community detection for the proposed method, AH-KSC, and KSC.

	Proposed method	AH-KSC	KSC
Precision	0.64	0.57	0.55
Recall	0.95	0.70	0.82
<i>F</i> -score	0.75	0.61	0.62

are the ground-truth community, the green colored node groups in the second column are the detected community from the proposed method, and the red colored node groups are the result of detected community from AH-KSC and KSC, respectively. From the observation of the experiment, AH-KSC agglomerates the neighbor nodes successfully in the early stages of agglomeration, as mentioned above, but it failed to terminate the agglomeration, as shown in the first and last networks in Figure 4 due to the lack of termination criteria. In addition, when the networks are too complex, such as the 2nd and 3rd networks, clustering is not done efficiently. KSC also produces similar results to AH-KSC but it clusters better than AH-KSC for the case in which the network is well organized, as with the 4th network. AH-KSC provides a better result than KSC when the network is very complex, such as the 2nd and 3rd network.

Table 1 shows overall accuracy of community detection using LiveJournal ground-truth network with respect to the precision, recall, and *F*-score for the proposed method, AH-KSC, and KSC. For AH-KSC, the average precision, recall, and *F*-score were 0.57, 0.7, and 0.61, respectively. For KSC, the average precision, recall, and *F*-score were 0.55, 0.82, and 0.62, respectively. For the proposed method, the average precision, recall, and *F*-score were 0.64, 0.95, and 0.75, respectively. The overall precision of AH-KSC and KSC were similar with a 2 percent difference but KSC has higher performance in the overall recall with more than 12 percent, which means the KSC detected more true positive nodes than AH-KSC from the network. The average *F*-score for AH-KSC and KSC is similar with only a 1% difference. The proposed

FIGURE 5: Example of detected community comparison.

method outperformed AH-KSC and KSC in all evaluation metrics. In average precision, the proposed method improved 7 to 9% compared to AH-KSC and KSC. In the average recall, the proposed method had the highest improvement with 25 to 13%. The average F -score of the proposed method is improved by 14%.

5. Conclusion

This paper introduced an agglomerative spectral clustering with conductance and edge weight for detecting communities. The proposed method projects the original points into eigenvector feature space in the first stage. In the second

stage, the eigenvector space and the number of edges between nodes are used to evaluate the similarity between nodes. Each node finds candidate for the most similar nodes. The third stage finds the conductance between the node and its candidate. If only the conductance improves, the nodes are agglomerated. The three-stage process is iterated until the network requires no further agglomeration. The time complexity of the proposed method is increased compared to AH-KSC because we check the conductance of each agglomerated node but it is necessary for more accurate detection. From the analysis of the experiment, the proposed method outperformed the AH-KSC and KSC using a real life network, LiveJournal.

The two contributions of this method can be summarized as follows. One is the improvement accuracy compared to related works. The other is that the proposed method is feasible for practical situations because the performance of the method is well suited to real life social networks. On the other hand, the eigenvector space is calculated in every iteration so that the computation time is slower than that of KSC. Our future work will focus on improving the time complexity with a method such as parallel computing.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2016R1D1A1B03932447).

References

- [1] S. Fortunato, "Community detection in graphs," *Physics Reports. A Review Section of Physics Letters*, vol. 486, no. 3-5, pp. 75–174, 2010.
- [2] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proceedings of the 19th International World Wide Web Conference (WWW '10)*, pp. 631–640, ACM, New York, NY, USA, April 2010.
- [3] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002.
- [4] M. E. Newman, "Community detection and graph partitioning," *EPL (Europhysics Letters)*, vol. 103, no. 2, Article ID 28003, 2013.
- [5] M. E. Newman, "Spectral methods for community detection and graph partitioning," *Physical Review E*, vol. 88, no. 4, 2013.
- [6] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.
- [7] Y. Zhao and G. Karypis, "Hierarchical clustering algorithms for document datasets," *Data Mining and Knowledge Discovery*, vol. 10, no. 2, pp. 141–168, 2005.
- [8] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.
- [9] Y. Ng, Andrew, M. I. Jordan, and Weiss Y., "On spectral clustering analysis and an algorithm," in *Proceedings of the Advances in Neural Information Processing Systems*, British Columbia, Canada, 2001.
- [10] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," in *Proceedings of the 12th IEEE International Conference on Data Mining, ICDM 2012*, pp. 745–754, bel, December 2012.
- [11] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.
- [12] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *The Bell System Technical Journal*, vol. 49, no. 1, pp. 291–307, 1970.
- [13] H. N. Djidjev, "A scalable multilevel algorithm for graph clustering and community structure detection," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4936, pp. 117–128, 2006.
- [14] A. Clauset, C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, no. 7191, pp. 98–101, 2008.
- [15] M. C. Ramos, "Divisive and hierarchical clustering techniques to analyse variability of rainfall distribution patterns in a Mediterranean region," *Atmospheric Research*, vol. 57, no. 2, pp. 123–138, 2001.
- [16] L. Lin, T. Luo, J. Fu, Z. Ji, and D. Xiao, "A new community detection based on agglomeration mechanism," in *Proceedings of the IEEE 2nd International Conference on Computing, Control and Industrial Engineering, CCIE 2011*, pp. 352–355, chn, August 2011.
- [17] M. Leng, J. Wang, P. Wang, and X. Chen, "Hierarchical Agglomeration Community Detection Algorithm via Community Similarity Measures," *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol. 10, no. 6, pp. 1510–1518, 2012.
- [18] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means, spectral clustering and normalized cuts," in *Proceedings of the KDD-2004 - Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 551–556, Seattle, Wash, USA, August 2004.
- [19] C. Alzate and J. A. K. Suykens, "Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 335–347, 2010.
- [20] R. Langone, R. Mall, C. Alzate, and J. A. K. Suykens, "Kernel spectral clustering and applications," *Unsupervised Learning Algorithms*, pp. 135–161, 2016.
- [21] R. Langone, C. Alzate, and J. A. K. Suykens, "Kernel spectral clustering for community detection in complex networks," in *Proceedings of the 2012 Annual International Joint Conference on Neural Networks, IJCNN 2012, Part of the 2012 IEEE World Congress on Computational Intelligence, WCCI 2012*, Queensland, Australia, June 2012.
- [22] R. Mall, R. Langone, and J. A. K. Suykens, "Agglomerative hierarchical kernel spectral data clustering," in *Proceedings of the 5th IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2014*, pp. 9–16, usa, December 2014.
- [23] B. Hendrickson and T. G. Kolda, "Graph partitioning models for parallel computing," *Parallel Computing*, vol. 26, no. 12, pp. 1519–1534, 2000.
- [24] C. F. Olson, "Parallel algorithms for hierarchical clustering," *Parallel Computing*, vol. 21, no. 8, pp. 1313–1325, 1995.
- [25] A. R. Brodtkorb, T. R. Hagen, and M. L. Sætra, "Graphics processing unit (GPU) programming strategies and trends in GPU computing," *Journal of Parallel and Distributed Computing*, vol. 73, no. 1, pp. 4–13, 2013.