Sarcheshmehpour, Y.; Tian, Y.; Zhang, L.; Jung, A.

Flow-Based Clustering and Spectral Clustering

Published in:
55th Asilomar Conference on Signals, Systems and Computers, ACSSC 2021

DOI:
10.1109/IEEECONF53345.2021.9723162

Published: 01/01/2021

Document Version
Peer reviewed version

FLOW-BASED CLUSTERING AND SPECTRAL CLUSTERING: A COMPARISON

Y. SarcheshmehPour, Y. Tian, L. Zhang, A. Jung

Department of Computer Science, Aalto University, Finland; firstname.lastname(at)aalto.fi

ABSTRACT

We propose and study a novel graph clustering method for data with an intrinsic network structure. Similar to spectral clustering, we exploit an intrinsic network structure of data to construct Euclidean feature vectors. These feature vectors can then be fed into basic clustering methods such as k-means or Gaussian mixture model (GMM) based soft clustering. What sets our approach apart from spectral clustering is that we do not use the eigenvectors of a graph Laplacian to construct the feature vectors. Instead, we use the solutions of total variation minimization problems to construct feature vectors that reflect connectivity between data points. Our motivation is that the solutions of total variation minimization are piece-wise constant around a given set of seed nodes. These seed nodes can be obtained from domain knowledge or by simple heuristics that are based on the network structure of data. Our results indicate that our clustering methods can cope with certain graph structures that are challenging for spectral clustering methods.

Index Terms— machine learning; clustering; non-smooth optimization; community detection; complex networks

I. INTRODUCTION

The analysis of networked data is often facilitated by grouping or clustering the data points into coherent subsets of data points. Clustering methods aim at finding subsets (clusters) of data points that are more similar to each other than to the remaining data points [29]. Many basic clustering algorithms aim at clusters that are enclosed by a hypersphere or hyperellipsoid in a Euclidean feature space. These methods are most successful if datapoints are characterized by Euclidean feature vectors whose distance is small (large) for datapoints in the same (different) cluster(s).

Graph clustering methods can be applied to data with an intrinsic network structure that reflects a notion of similarity between different datapoints. Many important application domains, ranging from the Internet of Things to the management of pandemics, generate distributed collections of local datasets ("big data over network") [22], [23]. The network structure of these local datasets might be induced by spatio-temporal proximity ("contact networks"), statistical dependencies, or functional relations [1], [2].

II. PROBLEM FORMULATION

We represent networked data using an undirected "empirical" graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$ [9], [24]. Every node $i \in \mathcal{V} = \{1, \ldots, n\}$ of the empirical graph represents a data point. The dataset consists of n different data points. A data point might be a single sensor measurement, an entire time series, or even a whole collection of videos.

Our approach does not take the individual nature of data points into account. Rather, we only use their similarities as encoded in the weighted edges of $\mathcal{G}$. In particular, two nodes i, i' are connected by an edge $\{i, i'\}$ if the corresponding data points are similar. The amount of similarity is quantified by a positive edge weight $A_{i,i'}$.

We find it convenient to use an oriented (or directed) version of the empirical graph $\mathcal{G}$ by declaring, for each undirected edge $\{i, i'\}$, the node $\min\{i, i'\}$ as tail and the other node $\max\{i, i'\}$ as head in the corresponding directed edge. The resulting oriented empirical graph $\overrightarrow{\mathcal{G}} := \big(\mathcal{V}, \overrightarrow{\mathcal{E}}, \mathbf{A}\big)$ has the same nodes as $\mathcal{G}$ and contains the directed edge (i, i') from node i to node i' if and only if $i < i'$ and $\{i, i'\} \in \mathcal{E}$. This directed edge $(i, i') \in \overrightarrow{\mathcal{E}}$ has the same weight $A_{i,i'}$ as the corresponding undirected edge $\{i, i'\} \in \mathcal{E}$.

We will use several matrices that are naturally associated with an empirical graph $\mathcal{G}$. The weight matrix $\mathbf{A}$ contains the edge weight $A_{i,i'}$ in its ith row and i'th column. The diagonal degree matrix $\mathbf{D} := \mathrm{diag}\{d^{(1)}, \ldots, d^{(n)}\}$ collects the (weighted) node degrees $d^{(i)} = \sum_{i'} A_{i,i'}$ for each node i. The graph Laplacian matrix $\mathbf{L} := \mathbf{D} - \mathbf{A}$ is instrumental for spectral clustering methods (see Section III).

There are also vector spaces naturally associated with an empirical graph. The vector space consisting of maps $\mathbf{u}$ from nodes to real numbers is denoted $\mathbb{R}^n$. A vector $\mathbf{u} \in \mathbb{R}^n$ is a function or map that assigns each node $i \in \mathcal{V}$ a real-number u_i. Similarly, we define the vector space $\mathbb{R}^{\overrightarrow{\mathcal{E}}}$ that consists of vectors $\mathbf{f}$ that assign each edge $(i, i') \in \mathcal{E}$ a real-number $f_{(i,i')}$.

Many graph clustering methods, such as spectral clustering, rely on a clustering assumption. This clustering assumption applies to a dataset for which we know a useful empirical graph $\mathcal{G}$ [6], [7], [9]. A cluster is then constituted by datapoints whose nodes in $\mathcal{G}$ are more densely connected with each other than with nodes representing data points outside the cluster [15], [18]. This informal clustering assumption can be made precise by constraining the cut-size

of cluster [9]. By the maxflow/mincut duality, requiring a small cut is equivalent to requiring a minimum amount of network flow that can be routed from the nodes inside a cluster through its boundary [25].

Using the above clustering assumption, we construct feature vectors $\mathbf{x}^{(i)}$ for each node (or datapoint) $i \in \mathcal{V}$. The feature vectors will be constructed such that two nodes i, i' that belong to a well-connected subset of nodes (a cluster) have more feature vectors $\mathbf{x}^{(i)}, \mathbf{x}^{(i')}$ with a small Euclidean distance $\left\|\mathbf{x}^{(i)} - \mathbf{x}^{(i')}\right\|$. Loosely speaking, the feature construction maps similarity between datapoints in the empirical graph $\mathcal{G}$ into proximity of their Euclidean feature vectors. This allows us then to apply standard clustering methods, such as k-means of soft clustering [29], to the network structured data.

III. SPECTRAL CLUSTERING

Before we detail our construction of the feature vectors $\mathbf{x}^{(i)}$ let us briefly review the construction used by spectral graph clustering methods [9] . The most basic variant of spectral clustering constructs the node feature vectors $\mathbf{x}^{(i)}$ using the eigenvectors of the graph Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$ [9]. The matrix $\mathbf{L}$ is positive semi-definite (psd) and therefore we can find an orthonormal set of eigenvectors [11]

$$\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(n)} \text{ with eigenvalues } 0 \leq \lambda_1 \leq \ldots \leq \lambda_n. \quad (1)$$

For a given a number k of clusters, spectral clustering methods use the feature vectors

$$\mathbf{x}^{(i)} = \left(u_i^{(1)}, \ldots, u_i^{(k)}\right)^T \text{ for every node } i \in \mathcal{V}. \quad (2)$$

The Ideal Case. To develop some intuition for the usefulness of the construction (2), consider an empirical graph that contains k components $\mathcal{C}^{(1)}, \ldots, \mathcal{C}^{(k)} \subseteq \mathcal{V}$ that are not connected by any edge. In this case, the first k (possibly repeated) eigenvalues in (1) are equal to 0. The correspond eigenvectors $\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(k)}$ constitute an orthonormal basis for the subspace of $\mathbb{R}^n$ that is spanned by the indicator vectors $\mathbf{e}^{(c)}$ of the components $\mathcal{C}^{(c)}$, $e_i^{(c)} = 1$ for nodes $i \in \mathcal{C}^{(c)}$ and $e_i^{(\mathcal{C}^{(c)})} = 0$ for nodes $i \notin \mathcal{C}^{(c)}$. Thus, the feature vectors of nodes in the same cluster $\mathcal{C}^{(c)}$ are identical and orthogonal to the feature vectors of nodes in different clusters $\mathcal{C}^{(c')}$ with $c' \neq c$.

General Case. In general, the empirical graph will contain edges between different clusters. A widely used approach to analyzing spectral clustering methods is to consider the effect of these inter-cluster edges as perturbations of the Laplacian matrix $\mathbf{L}$. For sufficiently small perturbations, the subspace spanned by the first k eigenvectors of $\mathbf{L}$ will be close to the subspace spanned by the indicator vectors $\mathbf{e}^{(c)}$. This deviation can be made precise using tools from linear algebra and results in conditions on the empirical graph for the success of k-means when applied to the feature vectors (2) [6], [9].

There are certain types of empirical graphs that are challenging for spectral clustering methods that use the features (2) [9], [27], [28]. For example, spectral methods tend to fail for datasets that consist of clusters with significantly varying sizes (see Section VI-A). Another example of a challenging network structure for spectral clustering is a chain graph [26].

Consider a chain graph that consists of two clusters connected via a single edge. The weights of all intra-cluster edges are equal to 1 while the weight A_o of the single boundary edge is slightly smaller. As the chain graph is connected, $\lambda_1 = 0$ with corresponding eigenvector $\mathbf{u}^{(1)}$ having identical entries. The smallest non-zero eigenvalue is λ_2 has a corresponding eigenvector $\mathbf{u}^{(2)}$ whose entries are depicted in Figure 1. The resulting feature vectors (see (2) with $k = 2$) do not reflect well the cluster structure of the chain graph.

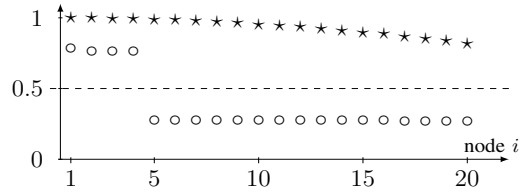

Fig. 1: Solution of TV minimization ("$\circ$") for the chain graph obtained from Algorithm 1 using $R = 1000$ iterations. Entries ("$\star$") of the eigenvector $\mathbf{u}^{(2)}$ corresponding to the smallest non-zero eigenvalue λ_2 (see (1)).

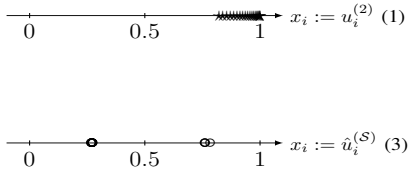

Fig. 2: Scatterplot of (scalar) node features x_i constructed by spectral and flow-based clustering for the chain graph in Figure 1. Spectral clustering uses the entries ("$\star$") of the eigenvector $\mathbf{u}^{(2)}$ (see (1)). Flow-based clustering uses the solution of TV minimization ("$\circ$") (see (3)).

We next describe our novel construction of feature vectors. The idea is to replace the eigenvectors of the Laplacian in (2) with solutions of total variation (TV) minimization problems. It turns out that these feature vectors better reflect the cluster geometry of $\mathcal{G}$ in terms of bottlenecks for network flows between different clusters. These flow bottlenecks are the boundaries between the clusters that are obtained by applying clustering methods to these feature vectors.

IV. TOTAL VARIATION MINIMIZATION

The construction of feature vectors (2) in spectral clustering methods is global. Indeed, the eigenvectors (1) of the graph Laplacian will typically depend on every edge in the empirical graph. Instead, we construct features in

a more local fashion by considering local clusters around seed nodes. These seed nodes might be obtained by domain knowledge (telling us which nodes must belong to the same cluster), based on basic connectivity properties (nodes with high degree), or chosen randomly.

Given a set of seed nodes $\mathcal{S}$ we solve the TV minimization problem

$$\hat{\mathbf{u}}^{(\mathcal{S})} = \operatorname*{argmin}_{\mathbf{u}\in\mathbb{R}^{\mathcal{V}}} \sum_{i\in\mathcal{S}}(1-u_i)^2/2+\sum_{i\notin\mathcal{S}}(\alpha/2)u_i^2+\lambda\|\mathbf{u}\|_{\rm TV}. \quad (3)$$

Here, we used the TV of a vector $\mathbf{u}\in\mathbb{R}^n$,

$$\|\mathbf{u}\|_{\rm TV} = \sum_{\{i,i'\}\in\mathcal{E}} A_{i,i'}|u_i-u_{i'}|. \quad (4)$$

Note that the TV minimization (3) is parametrized by the set $\mathcal{S}\subseteq\mathcal{V}$ of seed nodes. Our clustering methods solve multiple instances of (3), each time using another choice for the seed nodes $\mathcal{S}$. Section V discusses approaches for choosing a useful set of seed nodes.

We have recently explored the duality between TV minimization (3) and network flow optimization [10]. This duality allows to characterize the solution $\hat{\mathbf{u}}$ of (3) in terms of network flows. A network flow is a vector $\mathbf{f}\in\mathbb{R}^{\vec{\mathcal{E}}}$ whose entries $f_{(i,i')}$ represent a flow from node i to i'.

Theorem 1. *A vector $\widehat{\mathbf{u}}$ solves TV minimization (3) if and only if there is a flow vector $\mathbf{f}\in\mathbb{R}^{\vec{\mathcal{E}}}$ such that*

$$-\sum_{(i,i')\in\vec{\mathcal{E}}} f_{(i,i')} + \sum_{(i',i)\in\vec{\mathcal{E}}} f_{(i',i)} = \hat{u}_i - 1 \text{ for } i\in\mathcal{S},$$

$$-\sum_{(i,i')\in\vec{\mathcal{E}}} f_{(i,i')} + \sum_{(i',i)\in\vec{\mathcal{E}}} f_{(i',i)} = \alpha\hat{u}_i \text{ for } i\notin\mathcal{S},$$

$$|f_{(i,i')}| \le \lambda A_{i,i'} \text{ for all } (i,i')\in\vec{\mathcal{E}},$$

$$\hat{u}_i-\hat{u}_{i'}=0 \text{ for all } (i,i')\in\vec{\mathcal{E}} \text{ with } |f_{(i,i')}|<\lambda A_{i,i'}. \quad (5)$$

Proof. The result can be obtained by applying Fenchel duality [8, Ch. 31] to TV minimization (3) and a dual minimum cost flow problem (see [10, Sec. 3]). □

Let us illustrate Theorem 1 for the empirical graph in Figure 1. This empirical graph is a chain graph and partitioned into two clusters $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$ which are connected by a boundary edge b with weight A_o.

Ideal Case. Assume we solve TV minimization (3) for the chain graph in Figure 1 with a single seed node $\mathcal{S}=\{i^{(1)}\}$ with $i^{(1)}\in\mathcal{C}^{(1)}$. Moreover, we choose λ and α in (3) such that

$$\lambda A_o < 1, \text{ and } |\mathcal{C}^{(1)}|(\alpha/\lambda)+A_o<1, |\mathcal{C}^{(2)}|\alpha\ge\lambda A_o. \quad (6)$$

A direct inspection of the optimality condition (5) then yields the TV minimization solution

$$\hat{u}_i = \begin{cases} (1-\lambda A_o)/\big(1+\alpha(|\mathcal{C}^{(1)}|-1)\big) & \text{for } i\in\mathcal{C}^{(1)} \\ \lambda A_o/\big(\alpha|\mathcal{C}^{(2)}|\big) & \text{for } i\in\mathcal{C}^{(2)} \end{cases}. \quad (7)$$

Note that the vector (7) is piece-wise constant over the clusters $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$. Thus, if we would use (7) as (single) feature $x_i=\hat{u}_i$, basic clustering methods would successfully recover the clusters $\mathcal{C}^{(1)}$ and $\mathcal{C}^{(2)}$.

General Case. Note that (7) characterizes the solution of TV minimization (3) only when λ and α are chosen such that (6) is valid. However, condition (6) is not useful in practice as it involves the size of the clusters which we would like to determine. A practical approach to choosing α and λ can be based on probabilistic models for the empirical graph such as stochastic block models [21]. Alternatively, the choice for α and λ can be guided by the size of the piece $\{i\in\mathcal{V}:\hat{u}_i=\hat{u}_{i^{(1)}}\}$ around the seed node $i^{(1)}$. This piece grows for increasing λ and becomes the entire node set $\mathcal{V}$ whenever $\lambda A_{i,i'}>1$ for all edges $\{i,i'\}\in\mathcal{E}$.

The duality between TV minimization (3) and network flow optimization is also instrumental for developing iterative methods to solve (3). Algorithm 1 summarizes the application of a generic primal-dual method to solve (3) [10].

Algorithm 1 Primal-Dual Method For TV Minimization (3)

Input: $\mathcal{G}=\big(\mathcal{V},\mathcal{E},\mathbf{A}\big),\mathcal{S},\lambda,\alpha,R$
Initialize: $\hat{\mathbf{u}}^{(0)}:=\mathbf{0}$; $\hat{\mathbf{u}}^{(-1)}:=\mathbf{0}$;$\hat{\mathbf{f}}^{(0)}:=\mathbf{0}$;
Output: approximation $\widehat{\mathbf{u}}$ of solution $\widehat{\mathbf{u}}^{(\mathcal{S})}$ to (3)

for $r=0,\ldots,R-1$ **do**
  for $i\in\mathcal{V}$ **do** : $\tilde{u}_i:=2\hat{u}_i^{(r)}-\hat{u}_i^{(r-1)}$
  for $e=(i,i')\in\vec{\mathcal{E}}$ **do**
    $\hat{f}_e^{(r+1)}:=\hat{f}_e^{(r)}+(1/2)(\tilde{u}_i-\tilde{u}_{i'})$
    $\hat{f}_e^{(r+1)}:=\hat{f}_e^{(r+1)}/\max\{1,|\hat{f}_e^{(r+1)}|/(\lambda A_e)\}$
  end for
  for $i\in\mathcal{V}$ **do**
    $\hat{u}_i^{(r+1)}:=\hat{u}_i^{(r)}-\gamma_i\Big[\sum_{(i,i')}\hat{f}_{(i,i')}^{(r+1)}-\sum_{(i',i)}\hat{f}_{(i',i)}^{(r+1)}\Big]$
    if $i\in\mathcal{S}$ **then**
      $\hat{u}_i^{(r+1)}:=\big(\gamma_i+\hat{u}_i^{(r+1)}\big)/(\gamma_i+1)$
    else
      $\hat{u}_i^{(r+1)}:=\hat{u}_i^{(r+1)}/(\alpha\gamma_i+1)$
    end if
  end for
end for
$\widehat{\mathbf{u}}:=\hat{\mathbf{u}}^{(R)}$

The output $\widehat{\mathbf{u}}\in\mathbb{R}^n$ of Algorithm 1 is an approximation to the solution of (3). We assume that the number of iterations R used for Algorithm 1 is sufficiently large such that the output of Algorithm 1 can be considered a (numeric) solution to (3). The number R of iterations can be guided by probabilistic models for the underlying empirical graph

combined with the convergence rates guaranteed by primal-dual methods [4]. Alternatively, we can tune the number of iterations based on the final clustering result obtained by using Algorithm 1 as a sub-routine within our clustering method (see Algorithm 2).

It is instructive to interpret Algorithm 1 as a message passing method for iteratively optimizing the network flow $\hat{f}^{(r)}_{(i,i')}$. Step 5 enforces the capacity constraint $|\hat{f}^{(r)}_{(i,i')}| \leq \lambda A_{i,i'}$. Step 8 adjusts the value $\hat{u}^{(r)}_i$ based on the net flow into the node i. In step 10, flow is injected into seed nodes $i \in \mathcal{S}$ while in step (12) flow is leaked out of remaining nodes $i \notin \mathcal{S}$.

V. FLOW-BASED GRAPH CLUSTERING

We are now in the position to formulate our flow-based graph clustering method as Algorithm 2. This method constructs feature vectors $\mathbf{x}$ using the solutions of TV minimization (3) for different choices of seed nodes $\mathcal{S}$. Instead of using the values of eigenvectors for the graph Laplacian (as used by spectral clustering), we use entries of the vector solving TV minimization (3) to construct feature vectors for each node $i \in \mathcal{V}$.

Algorithm 2 Flow-Based Graph Clustering

Input: empirical graph $\mathcal{G}$, TV min. parameters λ, α, number k of cluster; number s of seeds.
Output: cluster assignments $\hat{c}_1, \ldots, \hat{c}_n \in \{1, \ldots, k\}$

1: **for** $r = 1, \ldots, s$ **do**
2: select new seed nodes $\mathcal{S}$ with Algorithm 3
3: run Algorithm 1 with $\mathcal{G}, \mathcal{S}, \lambda, \alpha$
4: store resulting vector in $\widehat{\mathbf{u}}^{(r)}$
5: **end for**
6: construct node features

$$\mathbf{x}^{(i)} = \left(\hat{u}^{(1)}_i, \ldots, \hat{u}^{(s)}_i\right)^T \text{ for every node } i \in \mathcal{V}. \quad (8)$$

7: compute cluster assignments $\hat{c}_i$ by applying k-means to feature vectors (8)

A key challenge for the successful application of Algorithm 2 is a suitable section of seed nodes in step 2. One simple approach is to choose the set $\mathcal{S}$ by randomly selecting a single node $i \in \mathcal{V}$. In general it is preferable to use more than one seed node. However, we must ensure that a particular selection of seed nodes $\mathcal{S}$ contains only nodes from the same cluster.

Our numerical experiments (see Section VI) use a simple heuristic method for selecting seed nodes $\mathcal{S}$ in step (2) of Algorithm 2. This heuristic method is summarized in Algorithm 3. Algorithm 3 constructs a new set of seed nodes by first randomly choosing a node i' with large degree $d^{(i)}$ and then adding all neighbours of i' that have a sufficiently large number of common neighbours with i'.

Algorithm 3 Select Seed Nodes $\mathcal{S}$

Input: empirical graph $\mathcal{G}$, minimum number of common neighbours η, minimum degree d
Output: seed nodes $\mathcal{S} \subseteq \mathcal{V}$

1: initialize $\mathcal{S} := \emptyset$
2: determine $i' := randomly\ select\ from\ \{i : d^{(i)} \geq d\}$
3: add node, $\mathcal{S} := \{i'\}$
4: **for** each i'' with $\{i', i''\} \in \mathcal{E}$ **do**
5: $\mathcal{D} = \{i''' : \{i''', i'\}, \{i'', i'''\} \in \mathcal{E}\}$
6: **if** $|\mathcal{D}| \geq \eta$ **then**
7: $\mathcal{S} := \mathcal{S} \bigcup \{i''\}$
8: **end if**
9: **end for**

VI. NUMERICAL EXPERIMENTS

For datasets without intrinsic graphs, we construct the empirical graph by using an Euclidean distance based equation to define the similarity between data points: $A_{i,j} := \exp\left(-\left\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\right\|_2^2 / \left(2\sigma^2\right)\right)$. Here, σ is a tuning parameter that chosen via cross-validation or using a probabilistic model for the data points to keep numerical stability.

VI-A. Experiments with Synthetic data

A dataset [28] with two clusters, as depicted in Figure 3, the first cluster is a set of 2D data points drawn from Gaussian density centered at $(2, 0.2)$ with diagonal covariance matrix $0.01 * I$, the second cluster is a set of 2D data points denoting uniform density in a rectangular region: $\{(x_1, x_2) \mid 0 < x_1 < 8, -0.05 < x_2 < 0\}$. As shown in Figure 3, our method significantly outperforms spectral clustering. Parameters used for Algorithm 2 are $\alpha = 0.005$, $\lambda = 0.01$, $\eta = 50$, and $d = 12$.

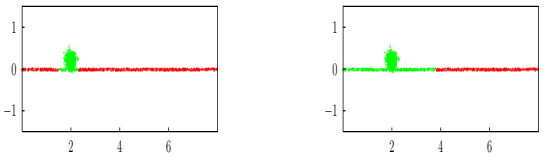

Fig. 3: The left plot is the clustering result of Algorithm 2, the right one is the clustering result of spectral clustering [6].

VI-B. Image Segmentation / Pixel Clustering

The performance of Algorithm 2 for image segmentation/pixel clustering is tested on some RGB images. We construct an empirical graph as mentioned in VI based on pixel values. Each pixel is connected to other pixels within up to three hops. empirical graph is forced to be sparse by removing some edges associated with small weights. Pixels in a rectangular are set to be seed nodes. After 1000 iterations, a local cluster which segment out the object of

Fig. 4: Image Segmentation: The left plot is the original image, the middle one is the segmentation result from Algorithm 2, the right one is the result from spectral clustering.

interest from the background is determined. As depicted in fig 4, the algorithm can accurately detect which pixels belong to the object. To compare and contrast, we also use spectral clustering to execute this task, the result in fig 4 shows that our algorithm out-performs spectral clustering.
The source code for the above experiments can be found at https://github.com/YuTian8328/.

VII. REFERENCES

[1] D. Koller and N. Friedman. "Probabilistic Graphical Models: Principles and Techniques. Adaptive computation and machine learning," in *MIT Press*, 2009.

[2] S. Banerjee, B.P. Carlin, and A.E. Gelfand. "Hierarchical Modeling and Analysis for Spatial Data," in *Chapman and Hall/CRC*, 2015.

[3] H.-J. Li and J.J. Daniels. "Social significance of community structure: Statistical view," in *IEEE Phys. Rev*, 91(1):012801, Jan. 2015.

[4] T. Pock and A. Chambolle. "Diagonal preconditioning for first order primal-dual algorithms in convex optimization," in *IEEE ICCV*, 21, Nov. 2011.

[5] A. Jung. "On the duality between network flows and network lasso," *IEEE Sig. Proc. Lett.*, 27:940 – 944, 2020.

[6] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Adv. Neur. Inf. Proc. Syst*, 2001.

[7] D. Spielman, "Spectral graph theory," in *U. Naumann and O. Schenk, editors, Combinatorial Scientific Computing. Chapman and Hall/CRC*, 2012.

[8] R. T. Rockafellar, "Convex Analysis," in *Princeton Univ. Press*, 1970.

[9] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Dec. 2007.

[10] A. Jung and Y. SarcheshmehPour, "Local Graph Clustering With Network Lasso," IEEE Signal Processing Letters (Volume: 28), 2020, pp. 106–110.

[11] G. H. Golub and C. F. Van Loan, "Matrix Computations, 3rd." Johns Hopkins University Press, 1996.

[12] D. Hallac, J. Leskovec, and S. Boyd, "Network lasso: Clustering and optimization in large graphs," in *Proc. SIGKDD*, 2015, pp. 387–396.

[13] S. Boyd and L. Vandenberghe, "Convex Optimization," Cambridge Univ. Press, Cambridge, UK, 2004.

[14] A. Jung, A O. Hero, A. Mara, S. Jahromi, A. Heimowitz, and Y.C. Eldar. "Semi-supervised learning in network-structured data via total variation minimization," *IEEE Trans. Signal Processing*, 67(24), Dec. 2019.

[15] A. BertrandMarc and M. Moonen. "Seeing the bigger picture: How nodes can learn their place within a complex ad hoc network topology," *IEEE Signal Processing Magazine*, 30(3):71-82, May 2013.

[16] M.C.V. Nascimento and A.C. De Carvalho. "Spectral methods for graph clustering–a survey," *European Journal of Operational Research*, 211(2):221–231, 2011.

[17] A. Jung and N. Tran. "Localized linear regression in networked data," *IEEE Signal Processing Letters*, vol.26, no. 7, pp. 1090–1094, 2019.

[18] B. Saha, A. Mandal, S.B. Tripathy , D. Mukherjee1 "Complex Networks, Communities and Clustering: A survey," *CoRR*, vol. abs/1503.06277, 2015.

[19] B. He, Y. You, and X. Yuan. "On the convergence of primal-dual hybrid gradient algorithm," *SIAM J. Imaging Sci.*, 7(4):2526–2537, 2014.

[20] C. Lee and D.J. Wilkinson . "A review of stochastic block models and extensions for graph clustering," *Applied Network Science.*, volume 4, Article number: 122, 2019

[21] A. Jung. "Clustering in partially labeled stochasticblock models via total variation minimization," *in-Proc.54th Asilomar Conf.*, Signals, Systems, Computers, Pa-cific Grove, CA, Nov. 2020.

[22] S. Cui and A. Hero and Z.-Q. Luo and J.M.F. Moura. "Big Data over Networks," *cup.*, Cambridge, UK. 2016.

[23] A. Jung. "Federated Learning over Networks for Pandemics," *Manning.*, 2021.

[24] A . Chapelle and B. Schölkopf and A. Zien. "Semi-Supervised Learning," *The MIT Press.*, Cambridge, Massachusetts. 2006.

[25] Y. Boykov and V. Kolmogorov. "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, 2004.

[26] A. Jung and M. Hulsebos. "The Network Nullspace Property for Compressed Sensing of Big Data over Networks," *Front. Appl. Math. Stat.*, 2018.

[27] B. Nadler and N. Srebro and X. Zhou. "Statistical Analysis of Semi-Supervised Learning: The Limit of Infinite Unlabelled Data," *Advances in Neural Information Processing Systems 22.*, pages. 1330–1338, 2009.

[28] Nadler, Boaz and Galun, Meirav. "Fundamental Limitations of Spectral Clustering," *Advances in Neural Information Processing Systems.*, B. Schölkopf and J. Platt and T. Hoffman, 2007, vol. 19, MIT Press.

[29] A. Jung. "Machine Learning: The Basics," *ArXiv e-prints https://arxiv.org/abs/1805.05052.*, 2020.