Wavelets and Linear Algebra 10(2) (2023) 29- 38

Vali-e-Asr University
of Rafsanjan

Wavelets and Linear Algebra

http://wala.vru.ac.ir

Spectral clustering by considering stationary distribution vector and transition matrix

Elaheh Vaziri[a], Mina Jamshidi[a,*], Hassan Motallebi[b]

[a] *Department of Applied Mathematics, Graduate University of Advanced Technology, Kerman, Iran.*
[b] *Faculty of Electrical and Computer Engineering, Graduate University of Advanced Technology, Kerman, Iran.*

Article Info

Article history:
Received 13 February 2023
Accepted 29 April 2023
Available online 22 November 2023
Communicated by Ali Armandnejad

Keywords:
Transition matrix, Random walk, Spectral clustering, Anchor nodes.

2010 MSC:
68P05, 15A99, 05C50.

Abstract

One of the popular methods of data clustering is spectral clustering. The main step of this method is constructing a graph representation of the data set and its similarity matrix. The similarity matrices which are constructed based on some important points not all data points, are among the main approaches. In this paper, the stationary distribution for a random walk on a weighted graph G is considered to find anchor points of the data set. Then we build the similarity matrix based on the anchor nodes and the weighted random walk transition matrix. After that, spectral clustering is applied on the gained similarity matrix. We propose the theoretical discussions and then we evaluate our method on benchmarks.

[*] Corresponding author
Email addresses: el68vazery@gmail.com (Elaheh Vaziri), m.jamshidi@kgut.ac.ir (Mina Jamshidi), h.motallebi@kgut.ac.ir (Hassan Motallebi)

http://doi.org/10.22072/wala.2023.1989619.1413

1. Introduction

Matrices and graphs have an essential role in representing data sets. A data set could be represented as its corresponding similarity graph. A similarity graph is a graph whose set of nodes represents the samples and the set of edges represents the similarities among them. Social networks and biological processes such as the dynamics of epidemics[16, 13, 14] are all actively studied by considering their correspondence graphs or matrices. Among different methods of analyzing a data matrix representation, spectral clustering has been one the most popular methods. In this method, first, a graph representation of the data set is constructed as a similarity matrix and then using the eigenvectors of its laplacian matrix an appropriate clustering of data set is obtained. Hence, one step in conventional spectral clustering methods is constructing the similarity graph. Some common approaches for constructing the graph representation of the data set are fully connected graph, k-nearest neighborhood graph and ε-neighborhood graph. Let $X = \{x_1, \ldots, x_n\}$ be a set of data samples. We denote the set of k-nearest neighbor of x by $N(x, k)$. In the k-nearest neighbor graph two points x_i and x_j are connected if $x_i \in N(x_j, k)$ or $x_j \in N(x_i, k)$. In the ε-neighborhood graphs, on the other hand, x_i and x_j are connected if and only if their distance is less than ε, where the appropriate value for ε is chosen according to the data set. After finding the adjacent nodes, the next step is assigning similarity or weights to the edges. Once the graph is constructed, we could use its similarity matrix.

There have been introduced several methods to construct representation matrices and graphs [2, 3, 4, 8, 9, 10, 19, 20]. In some recent methods, instead of considering all points actively in the construction, the main points are considered. For example, in [2], the authors have proposed the statistical methods to drop out some points. Also, some methods are introduced based on anchor nodes. Anchor based methods have been applied in learning approaches like dimension reduction, clustering and semisupervised learning [1, 7]. In these methods, a subset of data points are considered as essential points which can represent the collective behavior of the entire data set. Finding these set of points provides a better understanding of the data set.

Finding the set of anchor nodes is a challenging issue since there is no specific method. One may consider statistical, global or social behavior and other methods to find anchor nodes. The most important step in constructing the similarity matrix in anchor based methods is finding the anchor nodes. In this area of research, two most commonly approaches for generating the anchors are random selection and k-means generation. The performance of random strategy cannot be guaranteed. Although the k-means generation achieves good performance, it has high computational cost [1]. In [18], the authors proposed an anchor generation method called Balanced K-means based Hierarchical k-means (BKHK) which generates representative anchors with low computational cost. In this paper, we propose a new method for finding anchor points based on the stationary distribution of weighted random walks on a graph representation of data set. Then, we consider the transition matrix of the random walk to define weights of edges and the similarity matrix. Subsequently, we apply spectral clustering on the obtained matrix. We first briefly introduce the traditional spectral clustering method. Then, we explain the proposed method for finding anchor points and constructing the similarity matrix based on those points. We evaluate our method on some benchmarks and compare it with the previous methods.

2. Traditional Spectral Clustering Methods

Due to its ability in distinguishing clusters with different shapes, spectral clustering method, as a popular graph-based method, has attracted the attention of researchers in recent years. Also, unlike many clustering methods, which are sensitive to initial random points, the graph theory-based spectral clustering method does not have this problem. Also, unlike almost all methods in which the clustering process is applied directly on the original data, in spectral clustering method clustering is done through the graph Laplacian eigenvectors. In a famous method introduced by Ng et al. an embedding space is constructed by using top Laplacian eigenvectors [10]. Their method can be summarized by the following steps:

1) Constructing a representation graph of data: The affinity matrix is constructed by considering the similarity graph. The most popular method for constructing the representation graph is k-nearest neighbor graph. Denoting the set of k-nearest neighbor of x by $N(x, k)$, in the k-nearest neighbor graph two points x_i and x_j are connected if $x_i \in N(x_j, k)$ or $x_j \in N(x_i, k)$. This graph could be a weighted graph and often the Gaussian kernel function is considered as the similarity (i.e. weights of the edges) between connected nodes:

$$s(x_i, x_j) = exp(-\|x_i - x_j\|^2/(2\sigma^2))$$

2) Finding top eigenvectors of the Laplacian matrix: The second step is extracting top k eigenvectors of the Laplacian matrix $L = I - D^{-1}S$. In situations where the number of clusters is not determined, the eigengap method can be used to find the number of clusters.

3) Applying k-means: First, a matrix in which the columns are the top k-eigenvectors of L is constructed and then the k-means method is applied on the rows of the matrix to obtain the result.

3. Finding anchors based on weighted random walks

We start this section by giving a definition of a walk in a graph.

Definition 3.1. Let $G = (V, E)$ be a graph. A walk in G is a sequence of vertices and edges in which every vertex is incident to both the edges that come before and after it in the sequence. The total number of edges covered in a walk is called as length of the walk.

A random walk on a graph $G = (V, E)$ is known as a process that describes the position of a walk with random steps. On the other words, it is a process that begins at some vertex, and at each step moves to another vertex, randomly. It can be denoted by $\{\xi_t, t = 0, 1, 2, \ldots\}$, where ξ_t is a variable that describes the position of random walk after t steps. It could be considered also as a special case of Markov chain. In the initial state, ξ_0 is considered fixed or it could be gained from an initial distribution P_0. Then we may obtain the distribution of position after t steps by $P_t(i) = Pr(\xi_t = i)$. If G is unweighted, the vertex that the random walk selects for the next step is chosen uniformly at random among the adjacent vertices of the current vertex. When G is weighted, it moves to a vertex with probability proportional to the weight of the corresponding edge. Random walks on weighted graphs are described in [12].

Consider a weighted graph G with n nodes. Let Ω be an $n \times n$ adjacent matrix of G which is a symmetric matrix of weights where Ω_{ij} is the weight of the edge that connects v_i and v_j, and $\Omega_{ii} = 0$.

Definition 3.2. Let G be a weighted graph with the adjacent matrix Ω. The strength of the node v_i is defined as $S_i = \sum_{j=1}^{n} \Omega_{ij}$.

Definition 3.3. If a walker is at node i after t steps, the single step transition probability is defined as the probability of moving random walk to the node j in the next step. It is represented as π_{ij}:

$$\pi_{ij} = Pr(\xi_{t+1} = j|\xi_t = i).$$

The transition probability matrix $\Pi = (\pi_{ij})$ for weighted graph is calculated by $\pi_{ij} = \frac{\Omega_{ij}}{S_i}$, or we may write simply

$$\Pi = S^{-1}\Omega, \tag{3.1}$$

where S is a diagonal matrix in which the ith diagonal entry is the strength s_i of the ith node.

Definition 3.4. The t steps transition probability, represented by $P_{ij}(t)$, is defined as the probability for a walker to start from node i in $t = 0$ and reach node j in tth step:

$$P_{ij}(t) = Pr(\xi_t = j|\xi_0 = i).$$

It is common that for the weighted graph the t steps transition probability be calculated $P_{ij}(t+1) = \sum_{m=1}^{n} P_{im}(t)\pi_{mj}$ and consequently $P(t+1) = (\Pi^T)^t P(0)$. Note that the transition matrix $\Pi = (\pi_{ij})$ is not always symmetric. However, we have the following lemma:

Lemma 3.5. *Let S, P and Π be defined as above. We have:*

i) $S_i\pi_{ij} = S_j\pi_{ji}$.

ii) $S_iP_{ij}(t) = S_jP_{ji}(t)$.

Definition 3.6. A stationary distribution for a random walk is a distribution, π, for which we have $\pi P = \pi$.

One of the consequences of the above lemma is finding the stationary probability distribution $P_j^{\infty} = lim_{t\to\infty}P_{ij}(t)$, that gives the probability to find the random walker in the node j when $t \to \infty$. It has been proved that this limit is independent of i [**?**]. We have the following lemma.

Lemma 3.7. *Let S and P be defined as before. Then $P_j^{\infty} = \frac{S_j}{\sum_{l=1}^{n} S_l}$. Consequently, the stationary distribution for a random walk on a weighted graph G is*

$$P^{\infty} = (\frac{S_1}{\sum_{l=1}^{n} S_l}, \ldots, \frac{S_n}{\sum_{l=1}^{n} S_l})^T. \tag{3.2}$$

Algorithm 1 Spectral clustering by considering stationary distribution vector and transition matrix

```
Require: Data set $X$, $l$(number of anchor points), $k$, $\varepsilon$
Ensure: Clusters
Construct $\varepsilon$-neighborhood graph of data
for $i = 1, \dots, n$ do
    Compute strengths of $x_i$ according to definition 3.2
end for
Compute the transition matrix via Eq. 3.1
Compute the stationary distribution $P^\infty$ via Eq. 3.2
Find the set of $l$ anchors, $\mathcal{A}$
for $i = 1, \dots, n$ do
    Compute the $k$-nearest anchor points to $x_i$, i.e. $\mathcal{N}_l(\mathcal{A}, x_i)$
end for
Construct the similarity graph by Eq.4.1
Apply spectral clustering method
```

4. Spectral Clustering based on anchor nodes

In this section, we first introduce anchor nodes of weighted graph by considering stationary distribution for a random walk on a weighted graph. Then, we construct a similarity graph based on anchor nodes. Finally, we apply the spectral clustering method on the obtained similarity graph.

Let $X = \{x_1, \dots, x_n\}$ be a data set. First we construct the ϵ-neighborhood graph for this data, according to the following similarity matrix:

$$a_{ij} = \begin{cases} exp(-\|x_i - x_j\|^2/(2\sigma^2)) & \|x_i - x_j\| < \epsilon \\ 0 & else \end{cases}$$

Now, having the above similarity matrix we may find the stationary distribution for a random walk on a weighted graph G by lemma 3.7 which is $P^\infty = (\frac{S_1}{\sum_{l=1}^{n} S_l}, \dots, \frac{S_n}{\sum_{l=1}^{n} S_l})^T$. We expect that anchor points have more probability for meeting a random walker. Hence, to choose k anchor nodes, we consider the set $\mathcal{A} = \{v_{i_1}, \dots, v_{i_k}\}$, where $p_{i_1}^\infty, \dots, p_{i_k}^\infty$ are the k largest amounts among all P^∞ entries.

Having anchor nodes, some ways of construction of similarity graphs based on these points are well-known. One has been introduced in [7]. Here instead of considering conventional similarities like Gaussian kernel similarity to find K_{nn} anchor neighbors for each sample, we use random walk property. Let $\mathcal{A}$ be the set of generated anchors. For a sample $x_i \in X$, we consider $\mathcal{N}_l(\mathcal{A}, x_i)$ to be the set of l anchor nodes that have more probability to reach them starting from x_i. In other words, $\mathcal{N}_l(\mathcal{A}, x_i)$ is the set of l anchors x_j with the higher amount of π_{ij}. In the graph construction we just put edges between x_i and the l anchors in $\mathcal{N}_l(\mathcal{A}, x_i)$. For the connections between anchor nodes, if there is any, we ignore the directions, i.e. two anchors x_i and x_j are connected if $x_i \in \mathcal{N}_l(\mathcal{A}, x_j)$ or $x_j \in \mathcal{N}_l(\mathcal{A}, x_i)$; We also use the weight of the edge between an anchor point x_j and a sample x_i ,π_{ij}, to define similarity between nodes, however to have symmetric matrix we consider $\frac{\pi_{ij}+\pi_{ji}}{2}$. Hence

Table 1: The details of data sets used in our experiment

Dataset	Samples	Features	Classes
Ecoli	336	343	8
Control	600	60	6
Iris	150	4	3
Glass	214	9	6

we have a similarity graph based on anchor nodes and random walk with the following similarity matrix:

$$a_{ij} = \begin{cases} \frac{\pi_{ij}+\pi_{ji}}{2} & x_i, x_j \in \mathcal{A}, x_i \in \mathcal{N}_l(\mathcal{A}, x_j) \text{ or } x_j \in \mathcal{N}_l(\mathcal{A}, x_i) \\ \frac{\pi_{ij}+\pi_{ji}}{2} & x_i \in X - \mathcal{A}, x_j \in \mathcal{N}_l(\mathcal{A}, x_i) \\ 0 & else \end{cases} \tag{4.1}$$

Now we can apply traditional spectral clustering on this graph. We summarize our method in Algorithm 1.

5. Experiments

In this section, we have chosen four benchmarks to compare our method we some other famous methods. The details of benchmarks are listed in the table ??.

5.1. Comparison Methods

Two classic clustering methods (traditional spectral clustering(SC)[10] and k-means [21]) and four state-of-the-art clustering methods (Sparse Subspace Clustering (SSC)[11, 4], Low-Rank Representation (LRR)[19, 20], Least Squares Regression (LSR)[5], and Subspace Segmentation via Quadratic Programming [17]) are considered for comparison. The details of the comparison methods are listed below:

- *K*-means is one of the classical and popular distance based clustering algorithms. In this iterative algorithm, some points are chosen as the center points of clusters and then other points are assigned to each cluster by considering their distances to the centers.

- SC is introduced in section 2.

- SSC is a kind of subspace clustering methods in which points are represented in low-dimensional subspaces. The algorithm of this approach looks for a sparse coefficient matrix and the sparsity is imposed by $||.||_1$ on coefficient matrix.

- LRR is a kind of subspaces clustering approaches in which clusters are determined by using the low-rank self-expression of data.

- LSR uses the data global construction as well as relationships among the data points. In this method the block diagonal structure of the similarity matrix is constructed to find clusters.

- SSQP is a subspace segmentation method via quadratic programming achieve multiple clusters from multiple subspace as subspace partitioning of the quadratic programming.

5.2. Experimental setting

In the experiment, we employed three evaluation metrics (such as clustering ACCuracy (ACC), Normalized Mutual Information (NMI) and Adjusted Rand Index (ARI)[15]) to evaluate the clustering performance of the methods. The codes of evaluation metric is downloaded from Kang's GitHub [6]. We set the range of parameters (i.e., σ and k) in our method as $\sigma \in \{1, 10, 100\}$ and $k \in \{5, 10, 15, 20\}$, Also we consider number of anchors as about half of data points and the ε is chosen about 1/2 or 3/4 of the maximum distances between points.

In the following we briefly introduce ACC, NMI and ARI.

- ACC computes the percentage of samples in correct clusters, and defined as:

$$ACC = \frac{N_{cor}}{N},$$

where N is the number of all samples and N_{cor} is the number of samples in correct clusters.

- NMF is used for measuring the correlation between the two variables

$$NMI = \frac{2M(X_i, X_j)}{E(X_i) + E(X_j)}$$

where M denotes mutual information between two variables, and E represents the entropy of a variable.

- ARI computes the similarity between the prediction and the standard results. It is the adjusted form of RI and defined as:

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]}$$

5.3. Experimental analysis of the results

We have brought the clustering results of all the methods on the real data sets in Table **??**. We have used the results of comparison methods in [3] for previous methods. The following observations are noticeable. Our proposed method has often achieved the best performance, among the other methods, i.e. SC, k-means, LRR, LSR, SSC, and SSQP.

The accuracy of our method improved on average by 10 percent compared to the all comparison methods on all four data sets. This range of improvement has also accurd in NMI for three data sets Iris, Glass and Ecoli; For Control the NMI is not the highest amount in our method, however it measure is acceptable. ARI measure improvement is like NMI. It has been improved well compared with all methods in three data sets Iris, Glass and Ecoli; Also it is has a good improvement for Control compared with all methods except SC.

Table 2: Performance (ACC, NMI and ARI) on 4 data sets

data/method		(K-Means)	SC	LRR	LSR	SSC	SSQP	Ours
	ACC	72.53	90.67	80.00	80.00	80.00	85.33	**95.69**
Iris	NMI	64.19	78.38	64.59	64.59	63.08	71.18	**86.33**
	ARI	57.74	75.83	56.35	56.35	57.62	65.79	**88.58**
	ACC	44.39	64.49	50.93	50.89	50.47	51.68	**66.36**
Glass	NMI	33.60	55.85	32.94	38.94	32.67	33.24	**59.49**
	ARI	20.08	42.61	21.20	21.20	22.69	23.01	**45.87**
	ACC	49.46	53.44	56.55	57.04	62.82	55.03	**65.62**
Ecoli	NMI	46.96	50.83	47.12	46.92	46.14	46.28	**53.12**
	ARI	37.67	39.71	38.36	38.58	37.07	35.83	**43.94**
	ACC	56.83	66.20	25.17	57.17	70.50	60.17	**74.32**
Control	NMI	70.11	**76.13**	20.20	72.21	66.53	71.55	71.71
	ARI	51.94	**65.39**	60.99	60.81	52.95	59.75	61.24

5.4. *Parameters sensitivity*

Two main parameters need to be adjusted in our proposed algorithm, number of anchor nodes, l and number of chosen nearest anchor nodes to each node, k. Two other parameter are sensitive however the sensitivity of σ in Gussian kernel function has been discussed in past articles as well as ε for constructing ε-neighbor hood graphs. In this section, we changed the values of l and k by setting the range of k as $k \in \{5, 10, 15, 20\}$ and setting l to be half or quarter of the samples. We investigate the variations of the clustering accuracy of our method by changing these parameters. We have shown the results on Ecoil and Control data sets in Figures 1 and 2. The results showed that our method is sensitive to parameters setting. Moreover, the parameter l is more sensitive than the parameter k in our method.

References

[1] D. Cai and X. Chen, Large scale spectral clustering via landmark-based sparse representation, *IEEET Cybernetics*, **45**(8) (2015), 1669–1680.

[2] Y. Chen, Ch.-G. Li and Ch. You, Stochastic Sparse Subspace Clustering, *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA*, (2020), 4154–4163.

[3] T. Du, G. Wen, Zh. Cai, W. Zheng, M. Tan and Y. Li, Spectral clustering algorithm combining local covariance matrix with normalization, *Neural Computing and Applications*, **32** (2018), 6611–6618.

[4] R. Hu, X. Zhu, D. Cheng, W. He, Y. Yan, J. Song and Sh. Zhang, Graph self-representation method for unsupervised feature selection, *Neuro comput.*, **220** (2017), 130–137.

[5] P. Jain, Sh.M. Kakade, R. Kidambi, P. Netrapalli and A. Sidford, Parallelizing stochastic gradient descent for least squares regression: mini-batching, averaging, and model misspecification, *J. Mach. Learn Res.*, **18**(223) (2018), 1–42.

[6] Zh. Kang, X. Zhao, Ch. Peng, H. Zhu, J.T. Zhou, X. Peng, W. Chen and Z. Xu, Partition level multiview subspace clustering, *Neural Networks*, **122** (2020), 279–288.

[7] W. Liu, J.Sh.-Fu Chang, Large graph construction for scalable semi-supervised learning, *Proceedings of the 27th International Conference on Machine Learning*, (2010), 679–686.

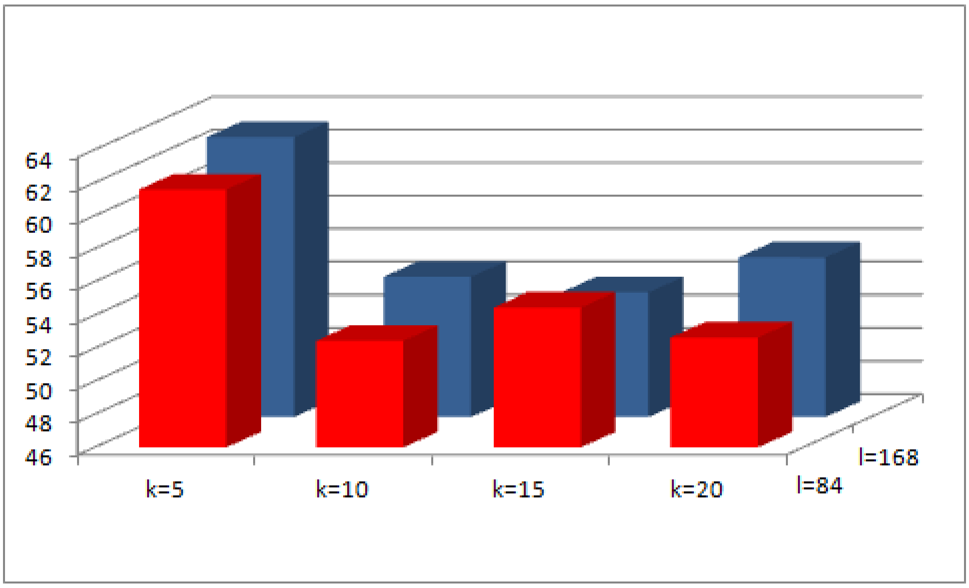

Figure 1: Acc w.r.t. *k* and *l* on Ecoli data

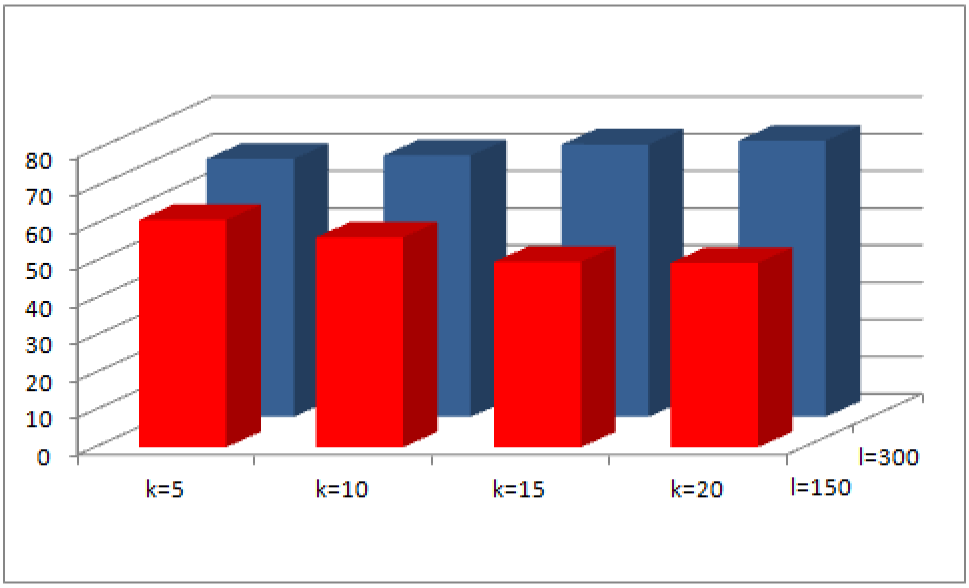

Figure 2: Acc w.r.t. *k* and *l* on Control data

[8] H. Motallebi and M. Jamshidi, A distance scaling method to improve spectral clustering of data with different densities, *International Journal of Data Science*, **6**(4) (2022), 328–347.

[9] H. Motallebi, R. Nasihatkon and M. Jamshidi, A local mean-based distance measure for spectral clustering, *Pattern Analysis and Applications*, **25** (2022), 351–359.

[10] A.Y. Ng, M.I. Jordan and Y. Weiss, On spectral clustering: Analysis and an algorithm, *Advances in Neural Information Processing Systems*, **14** (2001), 849–856.

[11] X. Peng, J. Feng, J.T. Zhou, Y. Lei and Sh. Yan, Deep sparse subspace clustering, *Computer vision and pattern recognition*, (2017), arXiv:1709.08374 [cs.CV].

[12] A.P. Riascos and J.L. Mateos, Random walks on weighted networks: Exploring local and non-local navigation strategies, *arXiv: Statistical Mechanics*, (2019), 1–21.

[13] F. Saberi-Movahed, M. Mohammadifard, A. Mehrpooya, M. Rezaei-Ravari, K. Berahmand, M. Rostami, S. Karami, M. Najafzadeh, D. Hajinezhad, M. Jamshidi, F. Abedi, M. Mohammadifard, E. Farbod, F. Safavi, M. Dorvash, N. Mottaghi-Dastjerdi, Sh. Vahedi, M. Eftekhari, F. Saberi-Movahed, H. Alinejad-Rokny, Sh. S. Band and Iman Tavassoly, Decoding clinical biomarker space of covid-19: exploring matrix factorization-based feature selection methods, *Computers in Biology and Medicine*, **146** (2022), 105426.

[14] F. Saberi-Movahed, M. Rostami, K. Berahmand, S. Karami, P. Tiwari, M. Oussalah and Sh.S. Band, Dual Regularized Unsupervised Feature Selection Based on Matrix Factorization and Minimum Redundancy with application in gene selection, *Knowledge-Based Systems*, **256**(28) (2022), 109884.

[15] J.M. Santos and M. Embrechts, On the use of the adjusted Rand index as a metric for evaluating supervised classification, *Artificial Neural Networks-ICANN*, (2009), 175–184.

[16] P. Satorras, C. Castellano, P.V. Mieghem and A. Vespignani, Epidemic processes in complex networks, *Reviews of Modern Physics*, **87** (2015), 925–979.

[17] Sh. Wang, X. Yuan, T. Yao, Sh. Yan and J. Shen, Efficient subspace segmentation via quadratic programming, *AAAI conference on artificial intelligence*, (2011), 519–524.

[18] W. Zhu, F. Nie and X. Li, Fast spectral clustering with efficient large graph construction, *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (2017), 2492–2496.

[19] X. Zhu, Sh. Zhang, R. Hu, Y. Zhu and J. Song, Local and global structure preservation for robust unsupervised spectral feature selection, *IEEE Trans Knowl Data Eng.*, **30**(3) (2018), 517–529.

[20] X. Zhu, Sh. Zhang, Y. Li, J. Zhang and L. Yang, Low-rank sparse subspace for spectral clustering, *IEEE Transactions on Knowledge and Data Engineering*, **31**(8) (2019), 1532–1543.

[21] J. Zhu and H. Wang, An improved K-means clustering algorithm, *IEEE International Conference on Information Management and Engineering*, **9**(1) (2010), 44–46.