International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol. 4, No. 3, September, 2011

# Feature Selection in Spectral Clustering

Soheila Ashkezari Toussi, Hadi Sadoghi Yazdi

*Computer Department, Ferdowsi University of Mashhad, Iran*
*Soheila.Ashkezari@stu-mail.um.ac.ir,h-sadoghi@um.ac.ir*

***Abstract***

*Spectral clustering is a powerful technique in clustering specially when the structure of data is not linear and classical clustering methods lead to fail. In this paper, we propose a spectral clustering algorithm with a feature selection schema based on extracted features of Kernel PCA. In the proposed algorithm, selecting appropriate vectors is dependent upon entropy of clusters on these vectors and weighting method is influenced by sum of the existence gap between clusters and entropy of the vectors. Tuning the parameters has a great effect on the results of spectral clustering techniques. In the ideal case, comparing our method with NJW and Kernel K-Means indicate the successful of the proposed algorithm.*

***Keywords:*** *spectral clustering, kernel PCA, feature selection, entropy*.

## 1. Introduction

Clustering can be considered the most important unsupervised learning problem in pattern recognition and machine learning. This problem deals with finding a structure in a collection of unlabeled data in such a way that objects belonging to a cluster are similar between them and dissimilar to the objects belonging to other clusters. There are many clustering algorithms developed in the literature which mainly contain hierarchical (Such as Single Link, Complete Link, etc.) and partitional (Such as K-means, Gaussian Mixture, Density Estimation ,etc.) clustering. When natural clusters in the input space are not correspond to convex region or when the databases become large, varied, and complex and of an unknown shape, especially in the presence of noise, conventional clustering methods yield to poor results. In such cases, subspace learning algorithms are proposed which aim to discover important statistical distribution on lower dimensions and obtain a new data representation. Subspace learning contains fields such as manifold learning [1],[2] spectral analysis [3],[4], kernel machine [5],[6] and so on.

The use of kernels allows mapping data into a high dimensional space called feature space. A linear partitioning in the feature space produces nonlinear separating hypersurfaces among clusters in the input space in order that reveals the structure of the original data. Kernel K-means, kernel FCM and kernel SOM are such clustering methods.

Spectral clustering techniques arise from the spectral graph theory and make use of the spectrum of the similarity matrix of the data to perform dimensionality reduction for clustering in fewer dimensions. Hence, the basic idea is to construct a weighted graph from the dataset in such a way that the vertices of the graph are data points and each weighted edge represents the degree of similarity between every correspond pair of vertices. Scott and Longute-Higgines algorithm [7], Perona and Freeman algorithm [8], Normalized cut [9] and NJW [10] are such spectral techniques. This kind of method has been successfully applied to parallel computations [11-12], load balancing[13] ,VLSI design [14], image segmentation [9], speech separation [15], bioinformatics [16-17], etc. The similarity and connection between kernel and spectral clustering methods has been discussed in [18].

Kernel principal component analysis [19] is an extension of principal component analysis (PCA) using techniques of kernel methods. Using a kernel, the originally linear operations of PCA are done in a reproducing kernel Hilbert space with a non-linear mapping. This method corresponds to an unsupervised learning technique which is useful for nonlinear feature extraction, de-noising and dimensionality reduction. The connection between spectral clustering and kernel PCA have been discussed in [20], [21] and [22].

Spectral clustering methods utilize the eigenvectors of the normalized affinity matrix derived from data to perform data partitioning. In most of these techniques the value of corresponding eigenvalues determines the priority of the eigenvectors. For example, for partitioning data on $K$ clusters, NJW always partitions data using the largest $K$ eigenvectors of the normalized affinity matrix of a dataset. However, this order does not guarantee to select the best features for clustering. For the first time, Xiang and Gong [23] proposed to use eigenvector selection to improve the performance of the spectral clustering method. In their approach after finding the largest $K_m > K$ eigenvectors of the normalized affinity matrix of the data, the relevance of each eigenvectors is estimated according to how well it can separate data into different clusters and utilizes these eigenvectors to perform data clustering. Result of this method in image segmentation is satisfied. However, the eigenvector combination consisting of all these relevant eigenvectors cannot always represent the data structure. Another approach for feature selection in eigenspace of spectral clustering is proposed by Feng Zhao et.al [24] which is based on entropy ranking. After finding the eigenvectors of the affinity matrix, all of these vectors will be ranked according to their importance on clustering and then a suitable subset of these eigenvectors is selected and combined to represent the structure of the original data. To evaluate the importance and obtain the ranking list of eigenvectors they proposed to remove each eigenvector in turn and compute the corresponding entropy of the rest of the eigenvectors set. If the removal of an eigenvector causes more disorder in the system than another, it shows more importance and higher rank of this eigenvector. Therefore, selection suitable eigenvectors is an important issue in spectral clustering because using uninformative eigenvectors could lead poor clustering results[23].

On the other hand one of the main drawbacks of spectral clustering is their limitation for test data. In [20], the extension of weighted Kernel PCA to spectral clustering for out-of-sample points has been discussed.

In this paper, we propose a new eigenvector selection method based on entropy of clusters. To achieve this goal, we use Kernel PCA as a powerful method to extract nonlinear features. Then, suitable eigenvectors which are able to reveal homogenous clusters are selected based on their entropy and weighted according to their eigenvalue and their ability for clustering by considering the existing gap in the projected data using them. Indeed, assigned weight to each vector shows its contribution in clustering. To express more precisely, clustering is done using some of top feature vectors (which are belonging to eigenvectors with largest eigenvalues) separately. Afterward, best vectors are selected and weighted. We use these weights for making weighted labels based and finally, the weighted average voting shows the final label of data points.

This paper is organized as follow: Sections 2 contains a review of existing spectral and kernel clustering techniques and a short review of Kernel PCA. In Section 3, we first discuss about the importance of eigenvector selections and then propose the selection and weighting schema for feature vector selection. Then, the proposed algorithm and the out-of-sample extension are presented. Section 4 contains the empirical results and related discussion, and in Section 5, we give conclusions.

## 2. Preliminaries

In this section, we review some spectral and kernel clustering methods. Then Kernel PCA and weighted Kernel PCA are reviewed shortly.

### 2.1. Spectral Clustering

Spectral clustering algorithms[25] have a strong connection with spectral graph theory [3]. It usually refers to the graphical partitioning based on the eigenvalues and eigenvectors of the adjacency (or affinity) matrix of a graph. Given a set of points in d-dimensional space $\mathbf{X}=\{\mathbf{x_1},\mathbf{x_2},...,\mathbf{x_n}\}$, we can build a complete, weighted undirected graph ***G (V, A)*** whose nodes $\mathbf{V}=\{v_1,v_2,...,v_\mathrm{n}\}$ corresponds to the *n* patterns and edges defined through the $n\times n$ adjacency matrix ***A*** encode the similarity between each pairs of sample points**.** Adjacency between two data points can be defined as follow:

$$a_{ij}=\begin{cases} h(x_i,x_j) & i\neq j \\ 0 & otherwise. \end{cases} \tag{1}$$

The function h measures the similarity between patterns by a typical Gaussian function

$$h(x_i,x_j)=\exp(-d^2(x_i,x_j)/\sigma^2) \tag{2}$$

where, *d* measure the dissimilarity or distance between patterns and the scaling parameter $\sigma$ controls how rapidly the affinity *h* falls off with the distance between $x_i$ and $x_j$.

The degree matrix ***D*** is a diagonal matrix whose element $D_{ii}=\sum_{j=1}^{N}A_{ij}$ is the degree of the point $x_i$.

**2.1.1. Shi and Malik algorithm (Normalized cut):** The proposed algorithm by Shi and Malik [9], applies the concept of graph partitioning to image segmentation using the *normalized cut* criterion. This criterion measures both the total dissimilarity between the different groups as well as the total similarity within the group. The algorithms composed of the following steps:

1.Set up a weighted graph ***G= (V, A)*** starting from the dataset ***X*** calculating the adjacency between patterns using

$$a_{ij}=\exp(-\frac{\left\|\mathbf{f}_i-\mathbf{f}_j\right\|^2}{2\sigma_1^2})\times\begin{cases} \exp(-\frac{\left\|\mathbf{x}_i-\mathbf{x}_j\right\|^2}{2\sigma_2^2}) & \text{if } \left\|\mathbf{x}_i-\mathbf{x}_j\right\|<\mathrm{R} \\ 0 & otherwise. \end{cases} \tag{3}$$

where $x_i$ is the position of the *i*th pixel and $\mathbf{f}_i$ is the feature vector of the pixel attributes, and R controls the sparsity of the adjacency matrix by effecting on the number of neighboring pixels.

2.Compute the degree matrix D.

3.Construct the Laplacian matrix $L_N=D^{-1/2}AD^{-1/2}$.

4.Compute the eigenvector **e₂** associated to the second smallest eigenvalue $\lambda_2$.

5.Use $\mathbf{D}^{-1/2}\mathbf{e}_2$ to segment the graph **G**.

**2.1.2. Ng-Jordan-Weiss (NJW) method:** The idea of NJW algorithm [10] is to find a new representation of patterns on the first *K* eigenvectors of the Laplacian matrix, where *K* is the number of clusters. The algorithm composed of the following steps:

1.Form the affinity matrix ***A*** by Equation 2.

2.Compute the degree matrix ***D*** and normalized affinity matrix $\mathbf{L_N} = \mathbf{D^{-1/2}AD^{-1/2}}$.

3.Compute the first *K* eigenvectors $v^1, v^2, ..., v^k$ corresponding to the first *K* largest eigenvalues $\lambda^1, \lambda^2, ..., \lambda^k$ of $\boldsymbol{L_N}$ and form the column wise matrix $\mathbf{V} = [\mathbf{v^1}, \mathbf{v^2}, ..., \mathbf{v^k}] \in \mathbf{R}^{n \times k}$.

4.Renormalize the matrix ***V*** and form the matrix ***Y*** such that all rows have unit length $\mathbf{Y}_{ij} = \mathbf{V}_{ij} / \sqrt{\sum_j \mathbf{V}_{ij}^2}$.

5.Cluster represented data matrix ***Y*** into *K* clusters via K-means.

The difference between NJW and N-cut is the renormalization of the ***V*** matrix (step four) and the main difference between NJW and Kernel PCA with a Gaussian kernel is the normalization of ***A*** (to form ***L***) and ***V*** in steps two and four, respectively.

## 2.2. Kernel Clustering Methods

Many of clustering methods have been modified incorporating kernel to produce nonlinear separating hypersurfaces among clusters. Kernel K-means, Kernel FCM and Kernel SOM are such techniques which allow mapping implicitly data into a high dimensional space in which clusters are more tight and clear. There is a close relation between spectral and kernel-based clustering methods.

**2.1.1. Kernel K-means:** The Kernel K-means algorithm [19] is a generalization of the standard K-means algorithm. Mapping points to a higher dimensional space by Mercer's theorem allows discovering clusters that are nonlinearly separable in the input space. In [26] it is showed that a general weighted version of Kernel K-means is mathematically equivalent to a weighted graph partitioning objective, so it may be used to directly optimize the graph portioning objectives and vice versa.

The Kernel K-means can be written as a minimization of the following objective function

$$\sum_{c=1}^{k}\sum_{x_i \in C_c} \left\| \Phi(x_i) - m_c \right\|^2 \text{, where } m_c = \sum_{x_i \in C_c} \Phi(x_i) \big/ \left| C_c \right| . \qquad (4)$$

$\Phi(x_i) : R^d \to R^{d_h}$ is the mapping function to the high dimensional nonlinear feature space, $C_c$ denote the *c*th cluster and $m_c$ is the mean of mapped data in the feature space. Expanding the distance $\left\| \Phi(x_i) - m_c \right\|^2$ and using Mercer's theorem we can compute distances between points and centroids without knowing explicit representations of $\Phi(x_i)$.

The equivalence of Kernel K-means to Kernel PCA prior to the conventional K-means is discussed in [27]. It is showed that the distance between samples in the Kernel PCA space is equivalent to that in the Hilbert nonlinear space.

### 2.2. Kernel PCA and Weighted Kernel PCA

A kernel based extension of principal component analysis is described in [19] by Schölkopf and et.al. Using a kernel, the linear PCA is performed in the feature space which is nonlinearly related to the original input space. Given a set of centered observations $\{x_i\}_{i=1}^{N}, x_i \in R^d$ , $\sum_{j=1}^{N} \varphi(x_i) = 0$ kernel PCA finds directions in which the projected variables $w^t \Phi(x_i), i = 1,...,N$ have maximal variance. The covariance matrix in the feature space is equal to $\mathbf{C} = 1/\mathrm{N} \sum_{j=1}^{N} \varphi(x_j) \varphi(x_j)^T$ . Using the theorem presented in [19], by defining a positive kernel it is possible to solve the equation $\mathbf{C}\mathbf{w} = \lambda \mathbf{w}$ in the feature space. Therefore, solving the eigen-decomposition of the kernel matrix $\mathbf{\Omega} \in \mathbf{R}^{N \times N}$ , $N\lambda\boldsymbol{\alpha} = \tilde{\mathbf{\Omega}}\boldsymbol{\alpha}$ , leads to find eigenvectors of the kernel matrix. Centering data in the feature space is equivalent to center the kernel matrix according to $\tilde{\mathbf{\Omega}}_{ij} = \mathbf{\Omega} - \mathbf{1}_N \mathbf{\Omega} - \mathbf{\Omega} \mathbf{1}_N + \mathbf{1}_N \mathbf{\Omega} \mathbf{1}_N$ , $\mathbf{1}_N = \mathbf{1}/N$ for all *i* and *j*. Hence, the projected variables become

$$\mathbf{Y} = w^t \Phi(x_i) = \sum_{j=1}^{N} \alpha^l \Phi(x_i) \Phi(x_j) = \alpha^l \Omega(x_i, x_j) . \qquad (5)$$

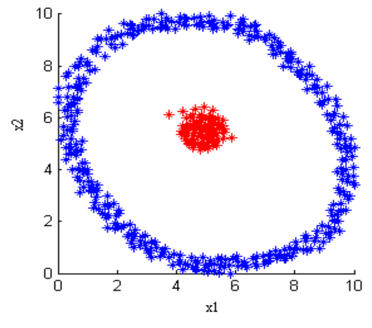
An LS-SVM approach to Kernel PCA is introduced in [28]. This approach showed that Kernel PCA is the solution to a primal optimization problem formulated in a kernel-induced feature space. Using this approach, in [20] and its earlier versions a weighted kernel PCA formulation is discussed in which introducing a symmetric positive definite diagonal weighting matrix V, leads to the dual problem $N / \lambda \boldsymbol{\alpha} = \mathbf{V}\tilde{\mathbf{\Omega}}\boldsymbol{\alpha}$ , where $\boldsymbol{\alpha}$ is a eigenvector and $\lambda$ is its corresponding eigenvalue. In addition the relation between weighted Kernel PCA and spectral clustering methods such as NCut, NJW and etc. is discussed. In this way by choosing the weighting matrix $\mathbf{V} = \mathbf{D}^{-1}$ and second largest eigenvector of Weighted Kernel PCA, the solution is equivalent to NCut method, also selecting the $\mathbf{D}^{1/2}\boldsymbol{\alpha}^2$ relaxed solution lead to the NJW method answer.

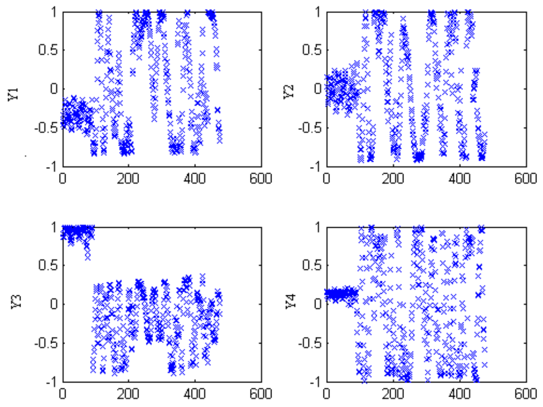
## 3. Proposed Method

In this section, we first discuss about the importance of feature selection in eigenspace and then present our proposed algorithm with its out-of-sample extension.

### 3.1. The Importance of Eigenvector Selection

In the previous sections we discussed about mapping the original input space into the nonlinear feature space using kernel or spectral methods. In the eigenspace, all the eigenvectors are not equally informative. Generally, in most of techniques the order of the eigenvalues determines the importance of eigenvectors, but as we will discuss in this section and it is discussed in recent researches such as [23-24] this order is not always suitable to describe the data and also all the eigenvectors are not informative. Therefore, to achieve a more obvious representation of data selecting the best eigenvectors is necessary. In Figure 1 a toy datasets and its first four projected data vectors (based on the first four eigenvectors) are illustrated. As it is indicated in Figure 1 (b) the third projected vector is more suitable to describe the dataset in Figure 1 (a) and it is preferred to use it instead of any others. Therefore, it can be inferred all eigenvectors are not equally informative. As discussed in[23], in principal, a clustering algorithm is expected to perform better when there are more information about data but in practice the incorporation of uninformative or noisy eigenvectors may lead to poor clustering results. Therefore it is obviously necessary to choose the best ones instead of all.





(a) (b)

**Figure 1. (a): A synthetic dataset (b): the first four projected data in the feature space using Kernel PCA, in $Y^3$ clusters are separated completely. σ=4**

## 3.2. Eigenvector Selection and Weighting Schema

Let a dataset $X=\{x_1,x_2,...,x_N\}$ consists of $K$ groups. Using Kernel PCA to map data into the nonlinear feature space and through eigen-decomposition of the kernel matrix, we can obtain its eigenvalues and eigenvectors. After projecting data into the feature space using equation (5), we are interested in selecting feature vectors which reveal structure of data as much as possible. Each eigenvector of Kernel PCA include information about all clusters, but indeed, only some of them are really informative and lead to enhance the clustering results. Therefore, we aim to find those vectors $Y^i, i=1,...,N$ which cause to obtain densest $K$ clusters. To fulfill this task, first eigenvectors $\alpha^i, i=1,...,N$ are sorted based on the descending order of their eigenvectors and data matrix is projected to the feature space using (5). Then on each vector $Y^i$, *k-1* thresholds will be set based on classical clustering techniques such as K-means or Linkage. At the end of this stage, there are $N$ data vectors each of them consists of $K$ clusters. Whereas using all eigenvectors (or projected data vectors $Y^i$) is not necessary, in practice just $K_S << N$ vectors are selected. To measure density of each cluster, we define the proposition 1.

**Proposition1.** Let $X_i^j=\{x_1^j,x_2^j,...,x_{N_j}^j\}, j=1,...,K$ is the set of *j*th cluster members on $Y^i$. We define *cluster density* as the proportion of sum of pairwise distance of $X_i^j$ to the sum of pairwise distance of all projected data points on $Y^i, i=1,...,N$ as follow,

$$density(X_i^j)=\sum_{i,k=1}^{N_j} d(x_i^j,x_k^j)\Big/\sum_{i,k=1}^{N} d(x_i,x_k). \quad (6)$$

Whatever data points distributed more uniformly among clusters, this value is closer to *1/k* and clusters are more homogeneous. Using next proposition we define a measure to evaluate ability of the vector $Y^i$ to describe the data structure based on the *clusters density*.

**Proposition2.** Let $density(X_i^j)$ is the *j*th cluster density computed by (6). Entropy of this cluster is defined as $entropy(X_i^j) = -density(X_i^j) \times \log(density(X_i^j))$, and total entropy of the projected data vector $Y^i$ is $entropy(Y^i) = \sum_{j=1}^{K} entropy(X_i^j)$. When clusters are homogeneous, they have almost equal entropy. For more clarifying, we define *cluster validity* which is $cluster\ validity(X_i^j) = entropy(X_i^j) / entropy(Y^i) \times K$. This value changes between 0 and $K$ and indicates how well data are partitioned among *j*th cluster on $Y^i$. If it is close to $K$, means all data points are collected into cluster $X^j$ and if this value is close to zero means the cluster is almost empty. These two states are not desired for our purpose. Therefore, we define *vector fitness* as (7) and then normalize it to have unit length,

$$fitness(Y^i) = \prod_{j=1}^{K} cluster\ validity(X_i^j) \tag{7}$$

This value is between 0 and 1. Whatever it is closer to 1, clusters are well defined and more homogeneous on $Y^i$ and this vector is a good candidate to represent data.

After computing fitness, best projected data vectors $Y^i$ are determined. Then, we should assign a proper weight to each one to determine their proportion in clustering. As discussed, vectors which are composed of tight clusters are preferred. Therefore, we define vector's weight based on the existing gap between clusters and the fitness of related eigenvectors which is used to compute projected data vector $Y^i$.

**Proposition3.** Let $G^i = \{g_1^i, g_2^i, ..., g_{k-1}^i\}$ is distance vector between $K$ clusters on $Y^i$ which $g_j^i, j = 1,...,K-1$ indicate the existing distance between clusters *j* and *j+1*. Then, we define $Gap^i$ to show the total distance between clusters on $Y^i$ as

$$Gap^i = \sum_{j=1}^{K-1} g_j^i. \tag{8}$$

**Proposition4.** Using proposition 2 and proposition 3, the weight of $Y^i$ is computed by

$$weight(Y^i) = fitness(Y^i) \times Gap^i = fitness(Y^i) \times \sum_{j=1}^{K-1} g_j. \tag{9}$$

As it is inferred, whatever clusters are far from each other, the summation of gap between them is larger and in consequence, the assigned weight is larger, so, the contribution of the vector in final clustering is higher.

### 3.3. Proposed Algorithm

Let $X = \{x_1, x_2, ..., x_N\}$ is a set of $N$ data points in $\boldsymbol{R^d}$ that we want to cluster into $K$ subsets. Steps of the proposed method for spectral clustering based on Kernel PCA extracted features are performed by following algorithm that we call Spectral Clustering based on projected data into Kernel PCA feature space (SCKPCA).

***SCKPCA algorithm***

**Input**: the original dataset $X = \{x_1, x_2, ..., x_N\}$; number of clusters $K$

**Output**: labeled data vector *IDX*

1. Form the kernel matrix $\mathbf{\Omega}$ in the feature space using Kernel PCA.
2. Compute eigenvectors $\boldsymbol{\alpha} = \{\alpha^i\}$ and eigenvalues $\lambda_i, i = 1,...,N$ through eigen-decomposition of $\mathbf{\Omega}$.

3. Sort eigenvectors based on descending order of eigenvalues in order that $\boldsymbol{\alpha} = [\alpha^1, \alpha^2, ..., \alpha^N]$ and $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$.

4. For each eigenvectors $\alpha^i, i = 1,...,K_S; K < K_S << N$ do

4.1. Project data onto $\alpha^i$ using (5) and form projected data vector $Y^i$.

4.2. Cluster data points on $Y^i$ using K-means, Linkage or etc.

4.3. Form data label vector $L^i$.

4.4. Set *K-1* thresholds on $Y^i$ according to the clusters center,

$$Thr_l^i = \frac{c_l^i + c_{l+1}^i}{2}, l = 1,...,K-1$$

where, $c_l^i$ is the center of cluster *l* on $Y^i$.

4.5. Compute fitness of $Y^i$ using (7).

4.6. Compute weight of $Y^i$ using (9).

5.Select those vectors $Y^i, i = 1,...,N$ where their fitness is higher than a predefined threshold and make a set of *best vectors* index.

6.Assign final labels of data points based on the clustering results on *best vectors* considering related weights.

$$IDX = \frac{\sum_{i=1}^{len_{bestvectors}} L^i \otimes weights(Y^i)}{\sum_{i=1}^{len_{bestvectors}} weights(Y^i)}, \tag{10}$$

where $\otimes$ is the element-by-element product of the labels array $L^i$ and weights vector $weights(Y^i)$.

---

When *K* is not very large (e.g. *K<10*), $K_S = 10$ is enough to determine suitable vectors for spectral clustering.

## 3.4. Parameter Selection

We used radial basis function (RBF) kernel $K(x_i, x_j) = \exp(-\|x_i - x_j\|_2^2 / \sigma^2)$ for Kernel PCA. The selection of tuning parameter $\sigma^2$ greatly affects on spectral clustering results [29], and results under different values of $\sigma^2$ are very different from another. The tuning method proposed in [29] introduces local scaling by selecting a $\sigma_i$ for each data point $x_i$ instead of the fixed scale parameter $\sigma$. The selection is done using the distance of the point $x_i$ to some nearest neighbor. The result of this approach depends on the choice of nearest neighbor. In [20] a criterion called balance line fit (BLF) is proposed for selection the scale parameter $\sigma$. This criterion is based on the eigenstructure of piecewise constant eigenvectors. In this paper, we propose a criterion based on two factors, ratio of within clusters variance to total variance of data in the input space and sum of gap between clusters in the feature space. The variance ratio is computed by $VarRatio = \sum_{c=1}^{K} \text{var}(\mathbf{x}^c) / \text{var}(\mathbf{x})$, where $\mathbf{x}$ is the set of all data points in the input space and $\mathbf{x}^c$ is the set of data points assigned to the cluster *c*. Gap value is equal to $TotalGap = \sum_{i \in bestvectors} Gap^i$, where $Gap^i$ is computed using (8). So, we define the *Gap-Variance Index* (GVI) as

$$GVI = \frac{VarRatio}{TotalGap} \tag{11}$$

Minimal value of the GVI determines the best choice for $\sigma$.

### 3.5. Out-of-sample Extension

The main drawback of spectral clustering methods is their limitation to a given dataset without a clear extension to test data points. It is possible to extend the proposed method in this paper to out-of-samples data. To achieve this goal, first a test data point should be projected into the feature space.

Then clustering is done by considering the position of this point on the '*best vectors*' set and adjusted thresholds (step4-4) on each vector of this set. The final cluster label of the test point is obtained using (9).

## 4.Experimental Results

In this section, some experimental results are presented to illustrate the proposed approach. Simulation on some toy data is presented. In addition, a number of datasets from UCI Machine Learning Repository are selected as benchmark comparisons. NJW and Kernel K-Means methods are performed for a baseline comparison. We use ratio of the number of true clustered data to the number of all data as accuracy measure.

### 4.1.Toy Problems

In toy problems, we use Linkage clustering with 'single' method to measure the distance between clusters for clustering data vectors ***Y***.

*Example 1: Nonlinear dataset*; the first toy problem consists of two-circle, Figure 2 (a). The total number of data points is 350. First four projected data vectors are illustrated in Figure 2 (b). Two clusters data points which are achieved using the proposed algorithm, are shown with two colors. Threshold is plotted with a yellow line on each vector. The RBF kernel parameter σ is set to 1.4 using (11).

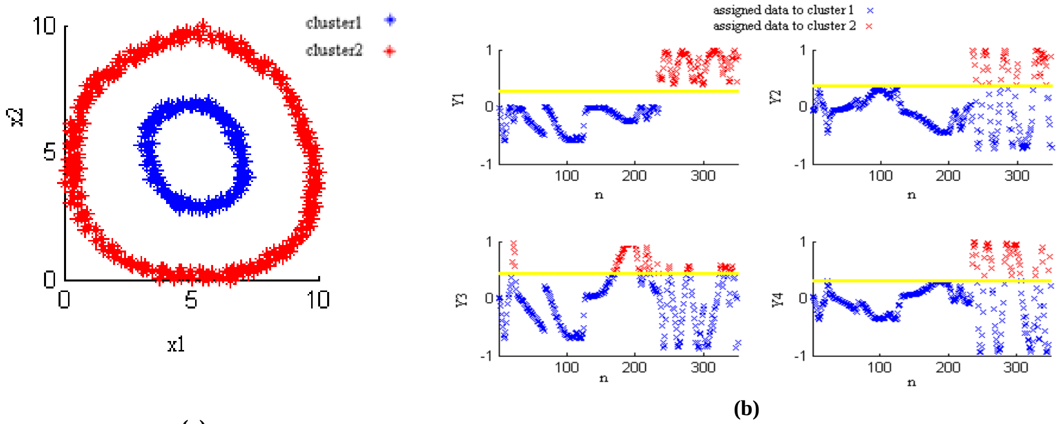


**Figure 2. (a): A synthetic dataset (b): the first four projected data in the feature space using Kernel PCA, in $Y^1$ clusters are separated completely. σ=1.4**

The GVI index is illustrated in Figure 3(a). Figure 3(b) shows the fitness of first ten projected data vector (right-top) and their weight (right-down). Threshold applied on vectors fitness (Eq. (8)) is set to 0.9, so for the first example just first vector is selected to represent data in the feature space, i.e. $best\ vectors = \{Y^1\}$.

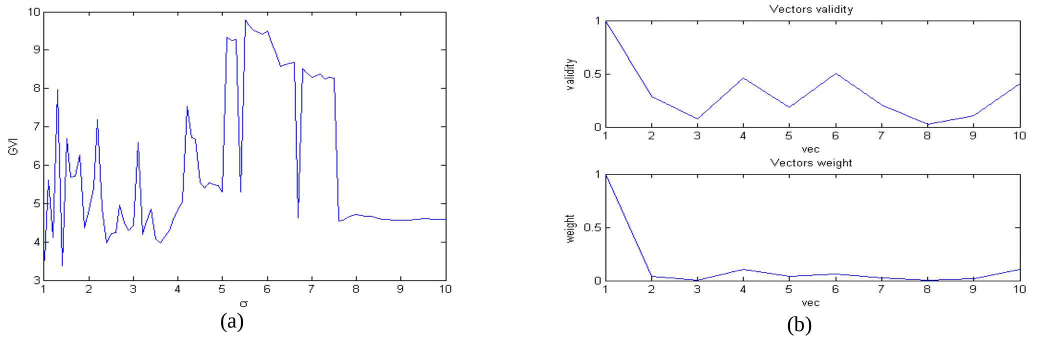


**Figure 3. Example 1 parameter setting: (a) Selecting optimal σ according to GVI index. (b) Fitness and weights of the first ten projected data vectors**

*Example 2: small dataset*; the second toy problem shows the influence of increasing number of clusters $K$ with small training set sizes. This dataset consists of 5 classes with 15, 5, 10, 11 and 5 samples and illustrated in Figure 4 (a). Figure 4 (b) shows the first four projected data vectors $\boldsymbol{Y}$. The optimal value for σ is 4.7, and GVI index is plotted in Figure 4 (c).

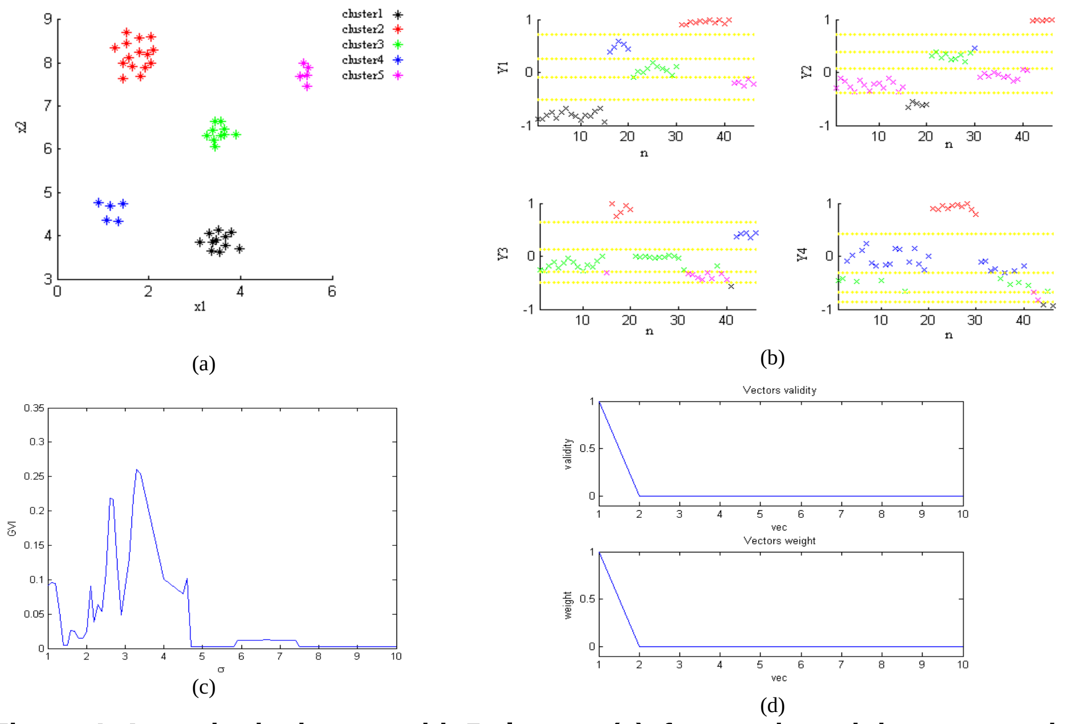
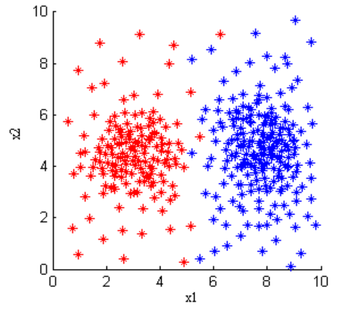


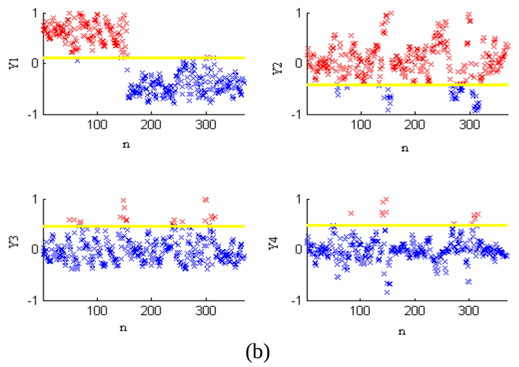
**Figure 4. A synthetic dataset with 5 clusters (a), four projected data vectors in the feature space (b), GVI index for selecting the best σ (c), fitness value and weight of first ten vectors (d).**

Fitness and weights of ten projected data vector are illustrated in Figure 4(d). As it is induced from these plots, the first vector is enough to represent dataset well.

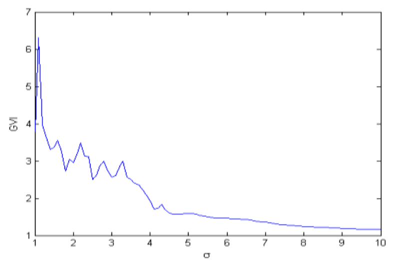
*Example 3: Noisy dataset*; the third toy example in Figure 5 (a), shows the influence of presence of noisy data in a dataset. The first four projected data vectors are illustrated in Figure 5(b). The GVI index is plotted in Figure 5(c), and best choice for σ is 10. Figure 5(d) shows the fitness and weights of first ten projected data vector.



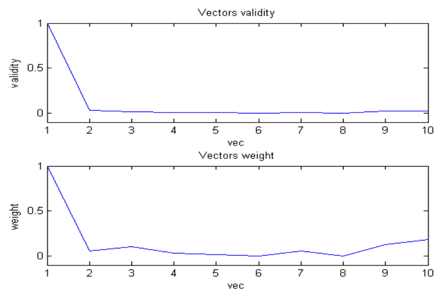
(a)



(b)



(c)



(d)

**Fig. 5. A synthetic dataset with noisy data (a), projected data to feature space (b-e), fitness value of first ten vectors (f), weight of first ten vectors (g).**

The first vector is again enough to represent the dataset and the accuracy of the proposed method is 0.9912.

## 4.2.Experiments on UCI benchmark datasets

In this section we examine the proposed method for spectral clustering on seven UCI benchmark datasets and compare its result with NJW and Kernel K-means. A summary of datasets is in Table 1.

**Table 1. UCI Benchmark Datasets Properties**

| **Datasets** | Instants | Attributes | Classes |
|---|---|---|---|
| Iris | 150 | 4 | 3 |
| Wine | 178 | 13 | 3 |
| Haberman | 306 | 3 | 2 |
| Breast | 569 | 30 | 2 |
| Soybean | 47 | 35 | 4 |

| | | | |
|---|---|---|---|
| Ionosphere | 351 | 34 | 2 |
| Pima | 768 | 8 | 2 |

We use 'Median' Linkage method for clustering projected data matrix ***Y***. In the first case (*I*), we examine the proposed method on all data. We used the proposed method in [29] as Zelnik method and also GVI (11) for selecting the best value for σ. The result of these methods is in Table 2. In addition, we search for the best accuracy of the clustering methods trough exhaustive search. Although this process is time consuming, but we are interested in because of a better comparison of different methods performance. To achieve this goal, we run three clustering methods for σ value in domain *[1:1: N]*, where *N* is the number of data points in a dataset. The result of this experiment is given in Table 3.

**Table 2. Performance comparison on UCI benchmark datasets (based on GVI and Zelnik criterion)**

| **Datasets** | NJW | | Kernel K-Means | | Proposed | |
|---|---|---|---|---|---|---|
| | GVI | Zelnik | GVI | Zelnik | GVI | Zelnik |
| Iris | 0.6733 | **0.8933** | 0.9067 | 0.7267 | **0.9467** | 0.7267 |
| Wine | 0.3616 | 0.4372 | **0.4686** | 0.4854 | 0.3401 | **0.6511** |
| Haberman | **0.5123** | **0.6511** | 0.4916 | 0.5568 | 0.4906 | 0.5474 |
| Breast | 0.5024 | 0.5048 | **0.8514** | **0.5682** | 0.5028 | 0.3334 |
| Soybean | 0.7750 | 0.7750 | 0.8015 | **0.8015** | **0.8162** | **0.8015** |
| Ionosphere | 0.7113 | 0.7517 | 0.7179 | **0.7602** | **0.7532** | 0.6387 |
| Pima | 0.5019 | 0.4856 | **0.5245** | 0.5321 | 0.5020 | **0.5820** |
| Best results | 1/7 | 2/7 | 3/7 | 3/7 | 3/7 | 3/7 |

Using GVI for selecting σ, in the best case, results of the proposed method over Iris, Haberman, Soybean and Pima is better than NJW and Kernel K-means. Using Zelnik for selecting σ, the proposed method has better results over Wine, Soybean and Pima.

**Table 3. Best performances comparison on UCI benchmark datasets**

| **Datasets** | NJW | Kernel K-Means | Proposed |
|---|---|---|---|
| Iris | 0.8000 | 0.9067 | **0.9467** |
| Wine | 0.5123 | **0.7170** | 0.6865 |
| Haberman | 0.4960 | 0.4891 | **0.6309** |
| Breast | **0.9075** | 0.8500 | 0.8955 |
| Soybean | 0.7868 | 0.6750 | **0.8162** |
| Ionosphere | 0.7517 | **0.7752** | 0.7532 |
| Pima | 0.6346 | 0.5894 | **0.7112** |
| Best results | 0/7 | 1/7 | 4/7 |

In the second case (*II*), for Iris, Haberman, Soybean and Ionosphere datasets, five split are considered where 80% of data are in the training set and test set consists of 20% of data. The accuracy measure of the proposed method over train and test datasets are given in Table 4. In this case, we use GVI for selecting the best value for σ. Furthermore, for each split the set of *'best vectors'* is determined.

**Table 4. Performance comparison on UCI benchmark datasets for train and test splits**

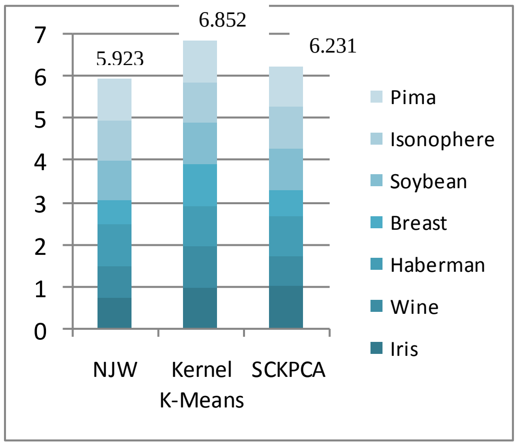
| Dataset | Proposed– train (5-fold) | | | Proposed– test (5-fold) | |
|---|---|---|---|---|---|
| | Accuracy | Average Accuracy | Best vectors | Accuracy | Average Accuracy |
| Iris | 0.9500<br>0.9333<br>0.9333<br>0.9167<br>0.9333 | 0.9333 | {1}<br>{1}<br>{1}<br>{1,9}<br>{1} | 0.7000<br>0.7000<br>0.7667<br>0.7333<br>0.7667 | 0.7333 |
| Haberman | 0.4823<br>0.5076<br>0.5248<br>0.5034<br>0.5158 | 0.5068 | {1,3,7}<br>{8}<br>{9}<br>{1}<br>{5} | 0.5729<br>0.5000<br>0.5111<br>0.5000<br>0.5000 | 0.5168 |
| Soybean | 0.8036<br>0.5625<br>0.8750<br>0.6696<br>0.5938 | 0.7009 | {2}<br>{2,3}<br>{2}<br>{2,5}<br>{2} | 0.5000<br>0.2500<br>0.8333<br>0.5000<br>0.2500 | 0.4667 |
| Ionosphere | 0.6867<br>0.6722<br>0.6733<br>0.5137<br>0.6916 | 0.6475 | {2}<br>{10}<br>{2,3}<br>{5,9}<br>{1,2} | 0.7077<br>0.5667<br>0.6000<br>0.7067<br>0.6556 | 0.6473 |

## 4.3.Robustness Analysis

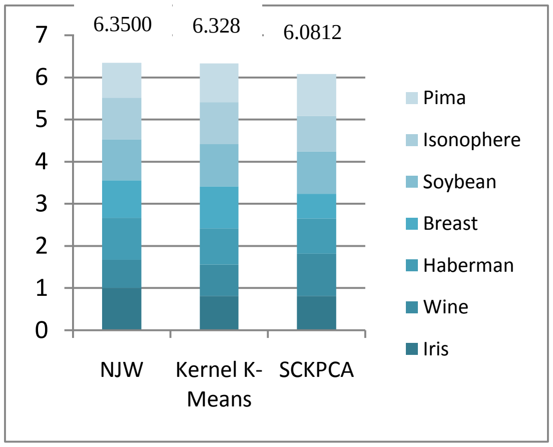
In this section we use the robustness analysis method proposed by Geng et al. [30] to evaluate the robustness of our method. The relative performance of the algorithm *a* on a dataset is represented by the ratio $R_a$ of its clustering recognition rate to the highest clustering accuracy among all the compared methods

$$R_a = \frac{Acc_a}{\max_j Acc_j}. \tag{12}$$

So the best method $a^*$ on that dataset has $R_{a^*} = 1$. The larger the value of $R_a$, the better the performance of the method *a* is on that dataset. Thus the sum of $R_a$ over all datasets offer a good measure of the robustness of the method *a*. A large value of the sum indicates good robustness. Figure 6 and 7 present the distribution of $R_a$ of three methods including NJW, Kernel K-Means and SCKPCA over seven datasets in Section 4.2. Total comparison among best results of three experiments (i.e. using Zelnik, GVI and exhaustive search) in Figure 7 (b), shows the priority of the proposed algorithm for spectral clustering. This Figure ure also indicate the importance of a good choice for σ.

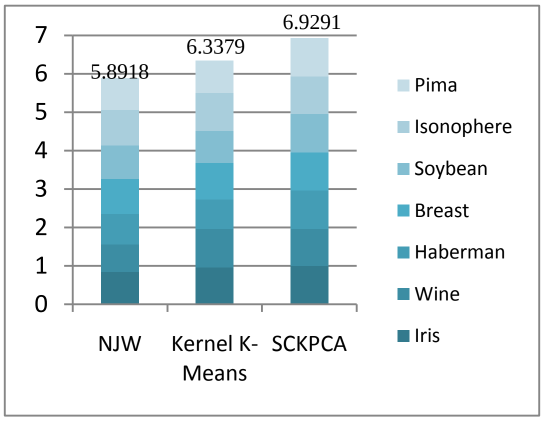


(a)

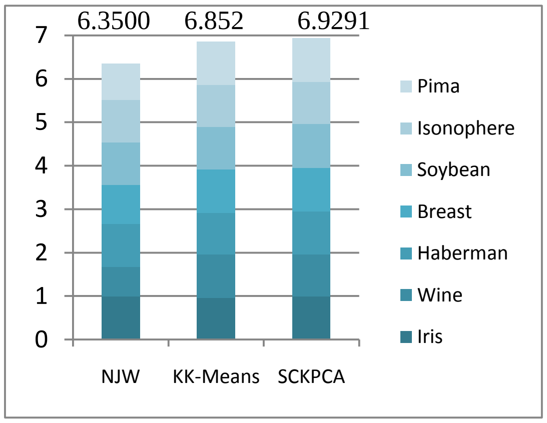


(b)

**Figure 6 . Robustness analysis based on the GVI (a), and Zelnik (b)**



(a)



(b)

**Figure 7. Robustness analysis based on best recognition rates of GVI index (a), Best results among all experiments for NJW with Zelnik criterian, Kernel K-means with GVI and SCKPCA with exhaustive search (b)**

## 5. Conclusion

In this paper, a feature selection method based on clusters entropy for spectral clustering is proposed. We use Kernel PCA with Gaussian RBF kernel to map data into a nonlinear feature space. In the present article, we used the entropy of clusters to define each vector's fitness and best vectors are selected based on this criterion. We defined each vector weight utilizing the sum of gap between clusters and considering the vector's fitness. Clustering is done based on each vector separately, and then considering vectors weight, final labels are determined for each data point. The proposed method can be trained and tested. This method is compared with NJW and Kernel K-Means. As it is inducted from experiments, different values for σ leads to completely different results. In this paper we proposed GVI index based on the variance and existing gap between clusters. Examining different methods for selecting best value of σ is one of our feature research directions. Also, we plan to compare our method with other spectral and kernel clustering methods. In addition, investigating other techniques for selecting best vectors is another purpose of future researches.

## Acknowledgment

This work has been partially supported by Iran Telecommunication Research Center (ITRC), Tehran, Iran (Contract number: 2-500/18496). This support is gratefully acknowledged.

## References

[1] S.T.Roweis and L.K.Saul, "Nonlinear dimensionality reduction by locally linear embedding," Science, vol. 290, pp. 2323–2326, 2000.

[2] J.B.Tenenbaum, et al., "A global geometric framework for nonlinear dimensionality reduction," Science, vol. 290, pp. 2319–2323, 2000.

[3] T.Zhang, et al., "A unifying framework for spectral analysis based dimensionality reduction " in IEEE World Congress on Computational Intelligence, 2008, pp. 1670–1677.

[5] K. R. Müller, et al., "An introduction to kernel-based learning algorithms," Neural Networks, IEEE Transaction on, vol. 12, pp. 181–202, 2001.

[6] B. Schölkopf and A. J. Smola, Learning with Kernels ,Support Vector Machines, Regularization, Optimization, and Beyond. Cambridge, MA, USA: MIT Press, 2002.

[7] G. L. Scott and H. C. Longuet-Higgins, "Feature grouping by relocalisation of eigenvectors of the proxmity matrix," presented at the British Machine Vision Conference, 1990.

[8] P. Perona and W. T. Freeman, "A factorization approach to grouping," in ECCV, 1998, pp. 655-670.

[9] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 22, pp. 888-905, Aug. 2000.

[10] A. Y. Ng, et al., "On spectral clustering: analysis and an algorithm," Advances in Neural Information Processing Systems, vol. 14, 2002.

[11] B.Hendrickson and R.Leland, "An improved spectral graph partitioning algorithm for mapping parallel computations," SIAM Journal of Scientific Computing, vol. 16, pp. 452–469, 1995.

[12] A. Pothen, ".Graph partitioning algorithms with applications to scientific computing, Parallel Numerical Algorithms," Kluwer, 1997.

[13] R. V. Driessche and D. Roose, "An improved spectral bisection algorithm and its application to dynamic load balancing," Parallel Computing, vol. 21, 1995.

[14] L. Hagen, "New spectral methods for ratio cut partitioninbg and clustering," Computer Aided Design, IEEE Transaction on, vol. 11, pp. 1074–1085, 1992.

[15] F.R.Bach and M.I.Jordan, "Learning spectral clustering, with application to speech separation," Journal of Machine Learning, vol. 7, pp. 1963–2001, 2006.

[16] D. Tritchler, et al., "A spectral clustering method for microarray data," Computational Statistics & Data Analysis, vol. 49, pp. 63-76, 2005.

[17] D. J. Higham, et al., "Spectral clustering and its use in bioinformatics," Journal of Computational and Applied Mathematics, vol. 204, pp. 25-37, 2007.

[18] M. Filippone, et al., "A survey of kernel and spectral methods for clustering," Pattern Recognition, vol. 41, pp. 176-190, 2008.

[19] B. Schölkopf, et al., "Nonlinear Component Analysis as a Kernel Eigenvalue Problem," Neural computation, vol. 10, pp. 1299-1319, 1996.

[20] C. Alzate and J. A. K. Suykens, "Multiway Spectral Clustering with Out-of-Sample Extensions through Weighted Kernel PCA," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 32, pp. 335-347, 2010.

[21] Y. Bengio, et al., "Spectral clustering and kernel PCA are learning eigenfunctions," CIRANO 1239, 2003.

[22] J. Ham, et al., "A Kernel View of the Dimensionality Reduction of Manifolds," in 21st Int'l Conf. Machine Learning, 2004, pp. 369-376.

[23] T. Xiang and S. Gong, "Spectral clustering with eigenvector selection," Pattern Recognition, vol. 41, pp. 1012-1029, 2008.

[24] F. Zhao, et al., "Spectral clustering with eigenvector selection based on entropy ranking," Neurocomputing, vol. 73, pp. 1704-1717, 2010.

[25] N. Cristianini, et al., "Spectral kernel methods for clustering," in Neural Information Processing Systems, NIPS'01, 2001, pp. 649–655.

[26] I. S. Dhillon, et al., "Kernel k-means: spectral clustering and normalized cuts," in tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004.

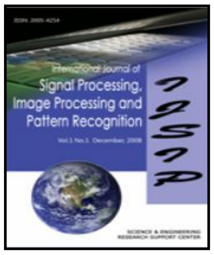
[27] J. Li, et al., "KPCA for semantic object extraction in images," Pattern Recognition, vol. 41, pp. 3244-3250, 2008.

[28] J. A. K. Suykens, et al., "A Support Vector Machine Formulation to PCA Analysis and Its Kernel Version," Neural Networks, IEEE Transaction on, vol. 14, pp. 447-450, 2003.

[29] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," Advances in Neural Information Processing Systems, vol. 17, pp. 1601-1608, 2004.

[30] X.Geng, et al., "Supervised nonlinear dimensionality reduction for visualization and classification " Systems,Man,and Cybernetics, IEEE Transactions on, vol. Part B 35, pp. 1098–1107, 2005.

## Authors



**Soheila Ashkezari Toussi** is currently M.S. student in Artificial Intelligence at Ferdowsi University of Mashhad, Iran. Her research interests include pattern recognition and intelligent systems.

**Hadi Sadoghi Yazdi** received the B.S. degree in electrical engineering from

Ferdowsi Mashad University of Iran in 1994, and then he received to the M.S. and Ph.D. degrees in electrical engineering from Tarbiat Modarres University of Iran, Tehran, in 1996 and 2005, respectively. He works in Department of Computer Engineering as Associate Professor at Ferdowsi University of Mashhad. His research interests include pattern recognition, adaptive filtering, image and video processing, and optimization in signal processing