Advanced Engineering Forum *Online: 2012-09-26*
ISSN: 2234-991X, Vols. 6-7, pp 243-249
doi:10.4028/www.scientific.net/AEF.6-7.243

Unsupervised Kernel Learning Vector Quantization

Kuo-Lung Wu

Department of Information Management, Kun Shan University, Yung-Kang, Tainan 71023, Taiwan
klwu@mail.ksu.edu.tw

Keywords: Learning vector quantization, kernel method, robustness.

Abstract. In this paper, we propose an unsupervised kernel learning vector quantization (UKLVQ) algorithm that combines the concepts of the kernel method and traditional unsupervised learning vector quantization (ULVQ). We first use the definition of the shadow kernel to give a general representation of the UKLVQ method and then easily implement the UKLVQ algorithm with a well-defined objective function in which traditional unsupervised learning vector quantization (ULVQ) becomes a special case of UKLVQ. We also analyze the robustness of our proposed learning algorithm by means of a sensitivity curve. In our simulations, the UKLVQ with Gaussian kernel has a bounded sensitivity curve and is thus robust to noise. The robustness and accuracy of the proposed UKLVQ algorithm is also demonstrated via numerical examples.

Introduction

The objective of vector quantization is to find a set of reference vectors $W_1,\cdots,W_c$ that best represent a data set. The k-means clustering algorithm is the best known batch-type vector quantization method that minimizes the mean squared error cost function [1,2]. The problem with these batch-version quantization algorithms is that the process may not converge to an optimal input data set configuration [3]. However, by adapting the competitive learning neural network to vector quantization, the convergence rate and quantization quality can be improved.

Self-organizing map (SOM) [4,5] is a two-layer feed-forward competitive learning neural network that can discover the topological structure hidden in data and display it in one or two dimensional space. It is also widely used in the vector quantization problem [6-8]. Suppose that W_k is the specified k^{th} reference vector and the feature vector $X(t)$ is inputted at time *t*, self-organization is then made using a two-step learning rule. First, it determines a winner neuron using the nearest neighbor condition:

$$\left\|X(t)-W_k(t-1)\right\| = \min_i \left\|X(t)-W_i(t-1)\right\| \tag{1}$$

This means that the weight of neuron *k* matches best with $X(t)$. Second, it updates all neurons using the learning formula

$$W_i(t) = W_i(t-1) + \alpha(t)h_{i,k}(t)(X(t)-W_i(t-1)). \tag{2}$$

where $\alpha(t)$ is the learning rate factor confined to decrease monotonically with time *t*. The selection of $\alpha(t)$ for a $W_i(t)$ being an asymptotically unbiased estimate can be found in Ref. [9]. The most often used is $\alpha(t)=\alpha/t$ with a constant α. The neighborhood function $h_{i,k}(t)$ denotes the lateral neural interaction phenomenon and the neuron excitation degree. Neuron excitation decreases in accomplishing the winner-take-all principle ($h_{i,k}(t)$ is 1 when $i=k$ and 0 otherwise) as time t increases. The winner neuron and its neighborhood will update toward $X(t)$ with a step size $\alpha(t)h_{i,k}(t)$. The vector quantization quality using competitive learning depends heavily on the annealing schedule $\alpha(t)$ and $h_{i,k}(t)$.

For the annealing schedule $h_{i,k}(t)$, Kohonen [4] first used the crisp competitive learning function to approximate the lateral neural interaction phenomenon and the most frequently used $h_{i,k}(t)$ is the Gaussian function [10,11]. Unsupervised learning vector quantization (ULVQ) adopts the winner-take-all neighborhood function to find a set of reference vectors W_i so that the expected error value

$$E(\sum_{i=1}^{c}\|X-W_i\|^2)=\sum_{i=1}^{c}\int_{C_i}\|X-W_i\|^2 f(x)dx \tag{3}$$

is minimized, where $f(x)$ denotes the probability density function of X and $C_1,\cdots,C_c$ represent c partitions. Since $\nabla_{W_i}(\|X-W_i\|^2)=-2(X-W_i)$, ULVQ tries to optimize Equation (3) using gradient descent update equation, i.e., Equation (2). The learning rate $\alpha(t)$ controls the descent speed.

In this paper, we combine the concepts of the kernel method and traditional ULVQ to develop the unsupervised kernel learning vector quantization (UKLVQ) algorithm. In Section 2, we first give an overview of the kernel method and the shadow kernel then introduce our UKLVQ algorithm at the end of the section. We then illustrate the robustness of the algorithm by means of numerical examples in Section 3. Finally, we present our conclusions in Section 4.

Unsupervised Kernel Learning Vector Quantization

The advantages of using the kernel methods are that a robust non-Euclidean distance measure is induced and the robustness of the algorithm to noise and outliers is enhanced. In this section, we give an overview of these kernel methods and then introduce our proposed unsupervised kernel learning vector quantization (UKLVQ) algorithm.

Kernel Methods. Kernel methods have been widely studied and applied in pattern recognition and function approximation, and they are one of the most important subjects in machine learning [12-14]. The common concept underlying the kernel method is the transformation of the data space into a high-dimensional feature space F with $\Phi: X \to F$, where the inner products in F can be represented by a Mercer kernel function defined on the data space. A kernel $K(x,y)$ in the feature space can then be denoted $K(x,y)=<\Phi(x)\Phi(y)>$, which is the inner product of $\Phi(x)$ and $\Phi(y)$. We then have

$$\begin{aligned}\|\Phi(x)-\Phi(y)\|^2 &= (\Phi(x)-\Phi(y))^T(\Phi(x)-\Phi(y)) = \Phi(x)^T\Phi(x)-\Phi(x)^T\Phi(y)\\ -\Phi(y)^T\Phi(x)+\Phi(y)^T\Phi(y) &= K(x,x)-K(x,y)-K(y,x)+K(y,y).\end{aligned} \tag{4}$$

If we choose the kernel functions that satisfy the conditions $K(x,x)=1$ and $K(x,y)=K(y,x)$, the above equation becomes

$$\|\Phi(x)-\Phi(y)\|^2 = 2(1-K(x,y)). \tag{5}$$

The most commonly used kernel is the Gaussian kernel

$$K(x,y)=\exp\{-(1/\sigma)\|x-y\|^2\}. \tag{6}$$

The kernel method is implemented as the k-means algorithm by transforming the data space into a high-dimensional feature space and restricting the cluster center to being the sample/weighted means of the data points in their feature space [15-20]. That is, the distance between the data point and the cluster center can be computed from

$$\|\Phi(x)-W_i^{\Phi}\|^2 = \left\|\Phi(x)-\frac{1}{|N_i|}\sum_{x_j\in C_i}\Phi(x_j)\right\|^2 = K(x,x)-\frac{2}{|N_i|}\sum_{x_j\in C_i}K(x,x_j)+\frac{1}{|N_i|^2}\sum_{x_j\in C_i}\sum_{x_k\in C_i}K(x_j,x_k) \tag{7}$$

where N_i denotes the number of x_j that belong to cluster C_i. Since we do not partition an online data set during the learning process, this implementation method is not available in an online sequential learning algorithm. Moreover, Zhang and Chen [12] have stated that these kernel clustering methods with cluster centers laid in the feature space may lack clear and intuitive descriptions.

The Proposed Kernel Learning Algorithm. The second way to implement the kernel method is to transform both the data points and the cluster centers into the feature space—resulting in

$$\left\|\Phi(x)-\Phi(W_i)\right\|^2 = 2(1-K(x,W_i)). \tag{8}$$

The objective of the unsupervised kernel learning vector quantization (UKLVQ) algorithm is to minimize the kernel-based expected error value

$$E(\sum_{i=1}^{c}\left\|\Phi(x)-\Phi(W_i)\right\|^2) = \sum_{i=1}^{c}\int_{C_i}\left\|\Phi(x)-\Phi(W_i)\right\|^2 f(x)dx = \sum_{i=1}^{c}\int_{C_i}2(1-K(x,W_i))f(x)dx \tag{9}$$

For a more general representation of the UKLVQ algorithm, we now define the shadow kernels. Let K be a kernel function with $K(x,W_i)=k(\left\|x-W_i\right\|^2)$. A kernel K is called a shadow of kernel H if $k'(r)=vh(r)$, where v is a constant. We then get

$$\nabla_{W_i}(\left\|\Phi(x)-\Phi(W_i)\right\|^2) = \nabla_{W_i}K(x,W_i) = \frac{\partial}{\partial W_i}k(\left\|x-W_i\right\|^2) = (x-W_i)h(\left\|x-W_i\right\|^2) = (x-W_i)H(x,W_i) \tag{10}$$

and the gradient descent update equation for UKLVQ given as

$$W_i(t) = W_i(t-1)+\alpha(t)h_{i,k}(t)H(X(t),W_i(t-1))(X(t)-W_i(t-1)) \tag{11}$$

where $\alpha(t)=\alpha/t$ and the competitive learning function $h_{i,k}(t)$ uses the winner-take-all principle. According to the definition of the shadow kernel, update equation (11) using kernel H will try to optimize the corresponding expected error objective (9) with shadow kernel K. (Note that one can adopt any suitable kernel H satisfying the conditions $H(x,x)=1$ and $H(x,y)=H(y,x)$ to create a UKLVQ algorithm with update equation (11).) However, these UKLVQ algorithms with an unknown shadow kernel will not optimize expected error objective (9).

If we choose the Gaussian kernel $K(x,W_i)=k(\left\|x-W_i\right\|^2)=\exp\{-(1/\sigma)\left\|x-W_i\right\|^2\}$ which is a shadow kernel of the kernel $H(x,W_i)=\exp\{-(1/\sigma)\left\|x-W_i\right\|^2\}$, we will then have

$$\nabla_{W_i}(\left\|\Phi(x)-\Phi(W_i)\right\|^2) = (x-W_i)\exp\{-(1/\sigma)\left\|x-W_i\right\|^2\} \tag{12}$$

and our proposed UKLVQ algorithm will try to optimize the kernel-based expected error value

$$\begin{aligned} E(\sum_{i=1}^{c}\left\|\Phi(x)-\Phi(W_i)\right\|^2) &= \sum_{i=1}^{c}\int_{C_i}2(1-K(x,W_i))f(x)dx \\ &= \sum_{i=1}^{c}\int_{C_i}2(1-\exp\{-(1/\sigma)\left\|x-W_i\right\|^2\})f(x)dx \end{aligned} \tag{13}$$

using the following gradient descent update equation:

$$W_i(t) = W_i(t-1)+\alpha(t)h_{i,k}(t)\exp\{-(1/\sigma)\left\|X(t)-W_i\right\|^2\}(X(t)-W_i(t-1)). \tag{14}$$

In this paper, we adopt the Gaussian kernel function in our UKLVQ algorithm and the above update equation is used in all our simulations. The following is another example of UKLVQ.

If we choose the kernel $K(x,W_i)=k(\|x-W_i\|^2)=1-\|x-W_i\|^2$ which is a shadow of the flat kernel $H(x,W_i)=h(\|x-W_i\|^2)=1$, we will have $\nabla_{W_i}(\left\|\Phi(x)-\Phi(W_i)\right\|^2)=(x-W_i)$, and our proposed UKLVQ algorithm will try to optimize the kernel-based expected error value

$$E(\sum_{i=1}^{c}\|\Phi(x)-\Phi(W_i)\|^2)=\sum_{i=1}^{c}\int_{C_i}2(1-K(x,W_i))f(x)dx=2\sum_{i=1}^{c}\int_{C_i}(\|x-W_i\|^2)f(x)dx$$

using the following gradient descent update equation:

$$W_i(t)=W_i(t-1)+\alpha(t)h_{i,k}(t)(X(t)-W_i(t-1)).$$

This indicates that the traditional ULVQ is a special case of the UKLVQ with flat kernel.

Robust Analysis and Numerical Examples

Although a good clustering algorithm should have the ability to tolerate noise and outliers, only a small section of the existing literature discuss the robustness of an unsupervised learning algorithm. Many criteria, such as sensitivity curves, breakdown points, local-shift sensitivity, gross error sensitivity, and influence functions, can be used to measure robustness. We will now use the sensitivity curve (SC) to discuss the robustness of our UKLVQ online learning algorithm. The sensitivity curve (SC) of the estimate $\hat{\mu}$ for the sample $x_1,\cdots,x_n$ is defined by

$$SC(x_0)=\hat{\mu}(x_1,\cdots,x_n,x_0)-\hat{\mu}(x_1,\cdots,x_n) \quad (15)$$

as a function of location x_0 of the outlier. The SC statistic denotes the influence of an outlier point on the estimate. If there are a set of locational parameters $\mu_1,\cdots,\mu_c$ that need to be estimated, we define the statistic SC^* to measure the influence of an outlier point on these estimators with

$$SC^*(x_0)=\sum_{i=1}^{c}[\hat{\mu}_i(x_1,\cdots,x_n,x_0)-\hat{\mu}_i(x_1,\cdots,x_n)]^2. \quad (16)$$

However, if the sensitivity curve is unbounded, an extreme outlier may cause problems. We will now give a simple example to simulate the SC^* statistic of the UKLVQ and ULVQ.

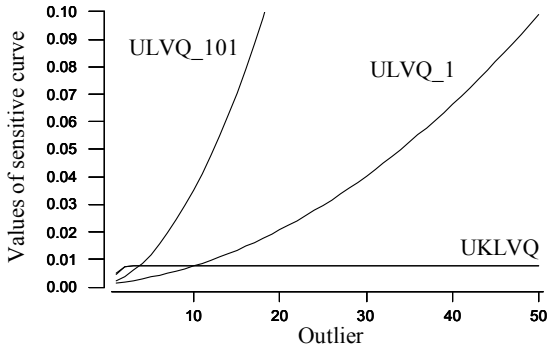

Fig. 1. The sensitivity curves of the ULVQ and UKLVQ.

Table 1. The details of the sensitive curves

outlier coordinate	ULVQ		UKLVQ	
	order1	order101	order1	order101
1	0.00125	0.00204	0.00487	0.00435
2	0.00169	0.00371	0.00722	0.00717
3	0.00219	0.00588	0.00744	0.00744
4	0.00275	0.00854	0.00744	0.00744
5	0.00338	0.01170	0.00744	0.00744
6	0.00408	0.01536	0.00744	0.00744
7	0.00484	0.01952	0.00744	0.00744
8	0.00567	0.02417	0.00744	0.00744
9	0.00656	0.02932	0.00744	0.00744
10	0.00752	0.03496	0.00744	0.00744
11	0.00854	0.04110	0.00744	0.00744
12	0.00962	0.04774	0.00744	0.00744
13	0.01077	0.05488	0.00744	0.00744
14	0.01199	0.06251	0.00744	0.00744
15	0.01327	0.07064	0.00744	0.00744
16	0.01462	0.07927	0.00744	0.00744
17	0.01603	0.08839	0.00744	0.00744
18	0.01750	0.09801	0.00744	0.00744
19	0.01904	0.10812	0.00744	0.00744
20	0.02065	0.11874	0.00744	0.00744

We first generate 100 random samples from standard normal distribution $N(0,1)$ and then add an outlier point x_0 into the data set. The coordinate of the outlier point is {1, 2, 3, …, 50}, respectively. Since the input order of the outlier will influence the learning performance, we consider two ways of inputting the outlier. The first is to input the outlier in order 1, which results in the inputted data order being $\{x_0,x_1,\cdots,x_{100}\}$. The second is to input the outlier in order 101, which results in the inputted data order being $\{x_1,\cdots,x_{100},x_0\}$. The sensitivity curves of ULVQ for these two kinds of learning orders are denoted ULVQ_1 and ULVQ_101, respectively. Figure 1 shows that the sensitivity curve of ULVQ is dependent on the learning order and is a monotonically increasing function of the outlier

coordinate. The resulting sensitivity curves of the UKLVQ for these two kinds of learning order are virtually identical, and give bounded outlier coordinates, as shown in Fig. 1. The details of these simulation results with the outlier coordinates {1, 2, 3, …, 20} are listed in Table 1.

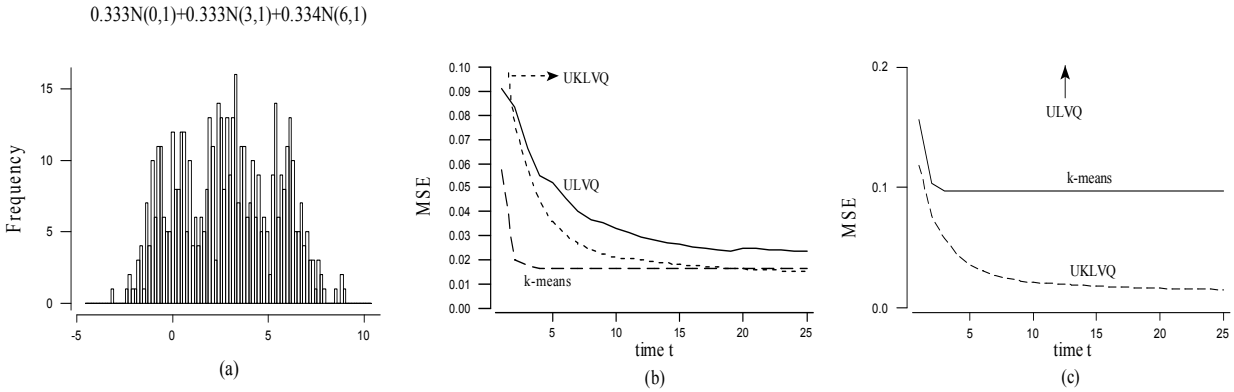

Fig. 2. (a) Histogram of 500 data points that form a three-class normal mixture. (b) The MSE values obtained by ULVQ, UKLVQ, and k-means versus time t without outliers. (c) The MSE values obtained by ULVQ, UKLVQ, and k-means versus time t with an outlier.

The above example shows the robustness of our proposed UKLVQ algorithm. We also tested its accuracy in parameter estimations. Figure 2(a) shows a histogram of 500 data points that form a three-class normal mixture. The MSE values obtained by ULVQ, UKLVQ, and k-means versus time t are shown in Fig. 2(b). We added an outlier point with coordinate 50 to the mixture data that resulted in the MSE values shown in Fig. 2(c). Since the MSE values of ULVQ with outlier are always greater than 10, we did not include it in Fig. 2(c). Moreover, the k-means is a batch version algorithm and the MSE values will not change after algorithm convergence. This example demonstrates the robustness and accuracy of the UKLVQ algorithm in parameter estimation.

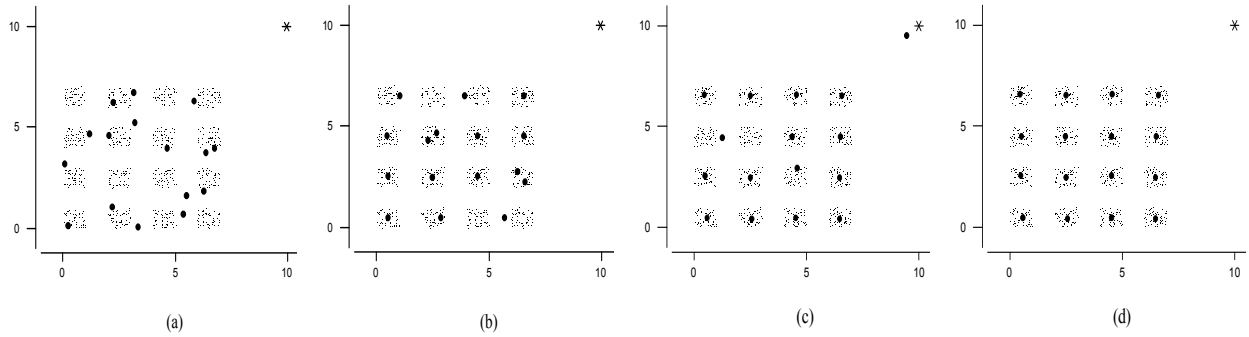

Fig. 3. (a) The 16-group data set. The noisy point is denoted by an asterisk. The dots and solid circles represent the data points and initial values. (b), (c), and (d) are the results of k-means, ULVQ, and UKLVQ, respectively.

Figure 3(a) shows a two-dimensional 16-group data set with sample size n = 800 consisting of 50 points in each of 16 clusters. The data points in each group are generated uniformly from each rectangle. We also added a noisy point with coordinate (10, 10), denoted by an asterisk. The dots and solid circles represent the data points and initial values. The vector quantization results (solid circles) obtained by k-means, ULVQ, and UKLVQ are illustrated in Figs. 3(b), 3(c), and 3(d), respectively. In this example, the k-means results cannot properly represent the data structure. However, it works well if the initial values are properly specified. This is a common problem of batch-version algorithms—that is, the quantization process may not converge to an optimal configuration of the input data set with a poor initialization. Although the ULVQ is less sensitive to initializations than k-means, it is influenced by the outlier. However, by adopting the kernel method of the ULVQ, the UKLVQ becomes robust to initials and noise with good quantization results.

Conclusions

We combined the concepts of kernel method and learning vector quantization to propose an unsupervised kernel learning vector quantization (UKLVQ) algorithm. In order to have a general representation of the UKLVQ method, we used the definition of shadow kernel to connect the relationship of the learning rule and kernel-based expected error objective. According to this general representation of the UKLVQ, we could easily implement a UKLVQ algorithm with a well-defined objective function—resulting in the traditional unsupervised learning vector quantization (ULVQ) with flat kernel becoming a special case of UKLVQ. We also analyzed the robustness of our proposed learning algorithm using sensitivity curves. In our simulations, the UKLVQ with Gaussian kernel had a bounded sensitivity curve, which indicates that it is robust to noise. The results of numerical examples also indicate that our proposed UKLVQ algorithm is robust and accurate.

Acknowledgments

This work was supported in part by the National Science Council of Taiwan, under Kuo-Lung Wu's Grant: NSC-101-2118-M-168-001

References

[1] S.P. Lloyd, Least squares quantization in pcm, IEEE Trans. Inf. Theory 28 (2) (1982) 129-137.

[2] D. Pollard, Quantization and the method of k-means, IEEE Trans. Inf. Theory 28 (2) (1982) 199-205.

[3] C. Chinrungrueng, C.H. Séquin, Optimal adaptive k -means algorithm with dynamic adjustment of learning rate, IEEE Trans. Neural Network 6 (1995) 157-169.

[4] T. Kohonen, Self-organized formation of topologically correct feature maps, Biol. Cybern. 43 (1982) 59-69.

[5] T. Kohonen, The self-organizing map, Proc. IEEE 78 (1990) 1464–1480.

[6] E. de Bodt, M. Cottrell, P. Letremy, M. Verleysen, On the use of self-organizing maps to accelerate vector quantization, Neurocomputing 56 (2004) 187-203.

[7] M.M. Campos, G.A. Carpenter, S-TREE: self-organizing trees for data clustering and online vector quantization, Neural Networks 14 (2001) 505-525.

[8] J.A. Lee, M. Verleysen, Self-organizing maps with recursive neighborhood adaptation, Neural Networks 15 (2002) 993-1003.

[9] E. Yair, K. Zeger, A. Gersho, Competitive learning and soft competition for vector quantizer design, IEEE Trans. Signal Process 40 (1992) 294-309.

[10] H. Ritter, K. Schulten, On the stationary state of Kohonen's self-organizing sensory mapping, Biol. Cybern. 54 (1986) 99-106.

[11] Z.P. Lo, B. Bavarian, On the rate of convergence in topology preserving neural networks, Biol. Cybern. 65 (1991) 55-63.

[12] D.Q. Zhang, S.C. Chen, A novel kernelized fuzzy c-means algorithm with application in medical image segmentation, Artificial Intelligence in Medicine. 32 (2004) 37-50.

[13] D.W. Kim, K.Y. Lee, D. Lee and K.H. Lee, A kernel-based subtractive clustering method, Pattern Recognition Letters 26 (2005) p. 879-891.

[14] E.A. Zanaty, S. Aljahdali and N. Debnath, A kernelized fuzzy c-means algorithm for automatic magnetic resonance image segmentation, Journal of Computational Methods in Science and Engineering 9 Supplement 2 (2009) 123-136.

[15] M. Girolami, Mercer kernel based clustering in feature space, IEEE Trans. Neural Networks 13 (2002) 780-784.

[16] A. Likas, N. Vlassis and J.J. Verbeek, The global k-means clustering algorithm, Pattern Recognition 36 (2003) 451-461.

[17] B. Scholkopf, A. Smola and K.-R. Muller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation, Neural Computation 10 (1998) p. 1299-1319.

[18] I.S. Dhillon, Y. Guan and B. Kulis, Weighted graph cuts without eigenvectors: a multilevel approach, IEEE Trans. Pattern Analysis and Machine Intelligence 29 (2007) 1944-1957.

[19] D.W. Kim, K.Y Lee, D. Lee and K.H. Lee, Evaluation of performance of clustering algorithms in kernel-induced feature space, Pattern Recognition 38 (2005) 607-611.

[20] J. Shi and J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Analysis and Machine Intelligence 22 (2002) 888-905.