# Kernel Optimization using Pairwise Constraints for Semi-Supervised Clustering

**Bojun Yan**
Department of Information and software Engineering
George Mason University
Fairfax, VA 22030
byan@gmu.edu

**Carlotta Domeniconi**
Department of Information and software Engineering
George Mason University
Fairfax, VA 22030
carlotta@ise.gmu.edu

## Abstract

A critical problem related to kernel-based methods is the selection of an *optimal* kernel for the problem at hand. The kernel function in use must conform with the learning target in order to obtain meaningful results. While solutions to estimate optimal kernel functions and their parameters have been proposed in a supervised setting, the problem presents open challenges when no labeled data are provided, and all we have available is a set of pairwise *must-link* and *cannot-link* constraints. In this paper we address the problem of optimizing the kernel function using pairwise constraints for semi-supervised clustering. To this end we derive a new optimization criterion to automatically estimate the optimal parameters of composite Gaussian kernels, directly from the data and the given constraints. We combine the optimal kernel function computed by our technique with a recently introduced semi-supervised kernel-based algorithm to demonstrate experimentally the effectivess of our approach. The results show that our method enables the practical utilization of powerful kernel-based semi-supervised clustering approaches by providing a mechanism to automatically set the involved critical parameters.

## 1 Introduction

Kernel-based methods enhance the modeling capability of learning algorithms by mapping data from the input space to a new feature space, usually of higher dimensionality. The key issue with kernel-based methods is the avoidance of an explicit knowledge of the mapping function. This is achieved by computing dot products in feature space via a kernel function. Kernel methods have been used with success to solve classification, clustering, and regression problems for a variety of applications.

Recently, a kernel method for semi-supervised clustering has been introduced [13]. In semi-supervised clustering, limited supervision is provided as input. Compared to traditional clustering algorithms, semi-supervised clustering employs both unlabeled and supervised data to obtain a partitioning that conforms more closely with the user's preferences. Several recent papers have discussed this problem [17, 8, 1, 19, 2, 13]. Typically, the supervision has the form of labeled data or pairwise constraints. A constraint on a pair of points specifies whether the two points belong to the same cluster (must-link), or not (cannot-link). Clearly, pairwise constraints can be induced by

labeled data. In many applications, such as information and image retrieval, qualitative measures of similarity between pairs of objects can be made available by the user. Thus, pairwise constraints can be readily available. The technique introduced in [13] extends semi-supervised clustering to a kernel space, thus enabling the discovery of clusters with non-linear boundaries in input space.

A critical issue related to kernel-based methods is the selection of an "optimal" kernel for the problem at hand. The performance of a kernel-based method depends critically on the selection of the kernel function, and on the setting of the involved parameters. The kernel function in use must conform with the learning target in order to obtain meaningful results. While solutions to estimate the optimal kernel function and its parameters have been proposed in a supervised setting, the problem presents open challenges when no labeled data are provided, and all we have available is a set of pairwise constraints.

For classification, the distribution of data in feature space should reflect the label distribution. The authors in [9] introduce the concept of *kernel alignment* to measure the correlation between groups of data in feature space and the distribution of labels to be learned. In [7] the authors consider the problem of automatically tuning multiple parameters for a support vector machine. This task is achieved by minimizing the estimated generalization error by means of a gradient descent approach over the set of parameters. In [18], a Fisher discriminant rule is used to estimate the optimal spread parameter of a Gaussian kernel. The authors in [10] propose a new criterion to address the selection of kernel's parameters within a kernel Fisher discriminant analysis framework for face recognition. A new formulation is derived to optimize the parameters of a Gaussian kernel based on a gradient descent algorithm.

The above mentioned research makes use of labeled data to address classification problems. In this work, instead, we are interested in kernel optimization using pairwise constraints; our aim is to solve clustering problems. Our objective is to learn a kernel function that maps pairs of points subject to must-link constraints close to each other in feature space, and maps points subject to cannot-link constraints far apart in feature space. To this end we derive a new optimization criterion to automatically estimate the optimal parameters of composite Gaussian kernels, directly from the data and the given constraints. Our approach integrates the constraints into an objective function that shapes the kernel mapping by bringing must-link points close to each other, and by taking cannot-link points far apart. As a result, our technique is able to automatically embed the optimal non-linear similarity within the feature space. This makes our adaptive technique capable of discovering clusters with non-linear boundaries in input space with high accuracy, as demonstrated in our experiments. Our proposed method enables the practical utilization of powerful kernel-based semi-supervised clustering approaches by providing a mechanism to automatically set the involved critical parameters.

The rest of the paper is organized as follows. Section 2 provides the necessary background on kernel-based clustering and semi-supervised clustering. Section 3 discusses the details of our algorithm. Section 4 describes our experimental settings and results, and finally we provide conclusions and future research directions in Section 5.

## 2 Background and Related Work

This section introduces the necessary background on kernel-based clustering and semi-supervised clustering.

### 2.1 Kernel KMeans

Let $X = \{\mathbf{x}_i\}_{i=1}^{N} \subseteq \Re^D$ be a set of $N$ samples with $D$ dimensions. Let $\phi : \Re^D \rightarrow \Re^{D'}$ be a non-linear mapping function, which maps data from the $D$ dimensional input space to a $D'$ dimensional feature space, with $D' > D$. The Kernel KMeans algorithm generates a $k$-partitioning $\{\pi_c\}_{c=1}^{k}$ of $X$ (where $\pi_c$ represents the $c^{th}$ cluster) so that the objective function $\sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\phi(\mathbf{x}_i) - \mathbf{m}_c^{\phi}\|$ is minimized, where $\mathbf{m}_c^{\phi} = \frac{1}{|\pi_c|} \sum_{\mathbf{x}_i \in \pi_c} \phi(\mathbf{x}_i)$ is the centroid of cluster $\pi_c$ in feature space. The key issue of Kernel-KMeans is the computation of distances in feature space. The distance of a point $\mathbf{x}_i$ from $\mathbf{m}_c^{\phi}$ in feature space can be expressed as: $\|\phi(\mathbf{x}_i) - \mathbf{m}_c^{\phi}\| = A_{ii} + B_{cc} - D_{ic}$, where $A_{ii} = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i)$, $D_{ic} = \frac{2}{|\pi_c|} \sum_{\mathbf{x}_j \in \pi_c} \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$, and $B_{cc} = \frac{1}{|\pi_c|^2} \sum_{\mathbf{x}_j, \mathbf{x}_{j'} \in \pi_c} \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_{j'})$.

Following the standard SVM method, we can represent the dot product of points in kernel space using an appropriate Mercer kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ [16]. Since all computations involving data points are in the form of dot products, the components for distance computation can be re-written using the kernel trick: $A_{ii} = K(\mathbf{x}_i, \mathbf{x}_j)$, $D_{ic} = \frac{2}{|\pi_c|} \sum_{\mathbf{x}_j \in \pi_c} K(\mathbf{x}_i, \mathbf{x}_j)$, and $B_{cc} = \frac{1}{|\pi_c|^2} \sum_{\mathbf{x}_j, \mathbf{x}_{j'} \in \pi_c} K(\mathbf{x}_j, \mathbf{x}_{j'})$. We note that $A_{ii}$ is common to every cluster, thus we can avoid calculating it, while $B_{cc}$ must be calculated once during each iteration.

### 2.2 HMRF Model and Kernel-based Semi-supervised Clustering

In semi-supervised clustering, we are given two sets of pairwise constraints: a set of must-link constraints $ML = \{(\mathbf{x}_i, \mathbf{x}_j)\}$, and a set of cannot-link constraints $CL = \{(\mathbf{x}_i, \mathbf{x}_j)\}$. The goal is to partition the data into $k$ clusters so that a given measure of distorsion between each point and the corresponding cluster representative is minimized, and, at the same time, the smallest number of constraint violation is achieved. Basu et al. (2004) [2] proposed a framework for semi-supervised clustering based on Hidden Markov Random Fields (HMRFs). Considering the squared Euclidean distance as a measure of cluster distortion, and the generalized Potts potential as constraint violation potential, the semi-supervised clustering objective can be expressed as [2]:

$$J_{obj}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\mathbf{x}_i - \mathbf{m}_c\|^2 + \sum_{\mathbf{x}_i, \mathbf{x}_j \in ML, l_i \neq l_j} w_{ij} + \sum_{\mathbf{x}_i, \mathbf{x}_j \in CL, l_i = l_j} \overline{w}_{ij}$$

where $\mathbf{m}_c$ is the centroid of cluster $\pi_c$, $ML$ is the set of must-link constraints, $CL$ is the set of cannot-link constraints, $w_{ij}$ and $\overline{w}_{ij}$ are the penalty costs for violating a must-link and a cannot-link constraint respectively, and $l_i$ represents the cluster label of $\mathbf{x}_i$.

Kulis et al. (2005) [13] extended this framework to a kernel-based semi-supervised clustering. Instead of adding a penalty term for a must-link violation, a reward is given for the satisfaction of the constraint. This is achieved by subtracting the corresponding penalty term from the objective:

$$J_{obj}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\phi(\mathbf{x})_i - \mathbf{m}_c^{\phi}\|^2 - \sum_{\mathbf{x}_i, \mathbf{x}_j \in ML, l_i = l_j} w_{ij} + \sum_{\mathbf{x}_i, \mathbf{x}_j \in CL, l_i = l_j} \overline{w}_{ij} \quad (1)$$

The algorithm derived in [13] (called SS-Kernel-KMeans), when combined with the Gaussian kernel, is shown to outperform the HMRF-KMeans approach [2], and SS-Kernel-KMeans combined with a linear kernel. However, the setting of the kernel's parameter is left to manual tuning, and the chosen value can largely affect the quality of the results. Thus, the selection of kernel's parameters remains a critical and open problem when only limited supervision is available. Our approach, discussed in the next Section, tackles this problem.

## 3 Kernel Optimization using Pairwise Constraints

### 3.1 Construction of the Kernel

Suppose $K_1$ and $K_2$ are known kernel functions. By using the closure properties of kernel functions [11], we can construct a new kernel $K$ as a linear combination of $K_1$ and $K_2$: $K = \alpha_1 K_1 + \alpha_2 K_2$, where $\alpha_1, \alpha_2 \in \Re^+$. By generalizing to $m \geq 2$, we consider a linear combination of $m$ Gaussian kernels $K_1$, $K_2$, $\cdots$, $K_m$, with unknown spread parameters $\sigma_1$, $\sigma_2$, $\cdots$, $\sigma_m$, respectively:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l=1}^{m} \alpha_l \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_l^2}\right) \quad (2)$$

where $\alpha_1, \alpha_2, \cdots, \alpha_m \in \Re^+$ and $\sum_{l=1}^{m} \alpha_l = 1$. For notational convenience, we group the parameters $\alpha_1, \cdots, \alpha_m, \sigma_1, \cdots, \sigma_m$ into a vector $\boldsymbol{\theta} = (\alpha_1, \cdots, \alpha_{m-1}, \sigma_1, \cdots, \sigma_m)$. We note that $\alpha_m = 1 - \sum_{l=1}^{m-1} \alpha_l$. Our goal is to optimize the kernel $K$ with respect to the parameters $\boldsymbol{\theta}$. We formulate $K$ in terms of Gaussian kernels since they are widely used in the literature, and have shown very good learning properties in a variety of applications. In practice, $m$ may be set to a small positive integer to allow accurate parameter estimations when the number of available pairwise constraints is limited. In our experiments, we set $m = 3$.

### 3.2 Definition of the Objective Function

We assume we are given a set of data $X = \{\mathbf{x}_i\}_{i=1}^{N} \subseteq \Re^D$, a set $ML$ of must-link constraints, and a set $CL$ of cannot-link constraints on $X$. We want to utilize this information to guide the clustering procedure to discover a partition of the data in $X$ which conforms with the given constraints. To this end, we want to compute optimal parameter values for the kernel function given in equation (2), using the given $ML$ and $CL$ constraints. The idea is then to learn a kernel that maps pairs of points subject to a must-link constraint close to each other in feature space, and maps points subject to a cannot-link constraint far apart in feature space. This goal is achieved by the following objective function:

$$F_{kernel} = \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in CL} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in ML} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 \tag{3}$$

The maximization of the above function implicitly defines a feature space in which similarities are measured according to the information provided by the given constraints. Thus, meaningful clusterings can readily be achieved in such feature space. By expanding the distance computation $\|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2$, and using the kernel trick $K(\mathbf{x}_i,\mathbf{x}_j) = \phi(\mathbf{x}_i)\cdot\phi(\mathbf{x}_j)$, we obtain the following:

$$\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in CL} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 = 2N_{CL} - 2\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in CL} K(\mathbf{x}_i,\mathbf{x}_j) \tag{4}$$

$$\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in ML} \|\phi(\mathbf{x}_i) - \phi(\mathbf{x}_j)\|^2 = 2N_{ML} - 2\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in ML} K(\mathbf{x}_i,\mathbf{x}_j) \tag{5}$$

where $N_{CL}$ is the number of cannnot-link constraints and $N_{ML}$ is the number of must-link constraints. By substituting equations (4) and (5) into (3), we can rewrite the objective function $F_{kernel}$ as follows:

$$F_{kernel} = 2N_{CL} - 2N_{ML} + 2\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in ML} K(\mathbf{x}_i,\mathbf{x}_j) - 2\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in CL} K(\mathbf{x}_i,\mathbf{x}_j) \tag{6}$$

### 3.3 Optimization Algorithm

To maximize the objective function $F_{kernel}$ given in equation (6), we define a gradient ascent algorithm. We consider the definition of $K$ given in (2), and compute the partial derivatives of $F_{kernel}$ with respect to the components $\theta_i$ of the vector of parameters $\boldsymbol{\theta}$:

$$\frac{\partial F_{kernel}}{\partial \theta_i} = 2\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in ML} \frac{\partial K(\mathbf{x}_i,\mathbf{x}_j)}{\partial \theta_i} - 2\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in CL} \frac{\partial K(\mathbf{x}_i,\mathbf{x}_j)}{\partial \theta_i}$$

Computing the partial derivatives of $K$ with respect to the parameters $\sigma_l$, for $l = 1, 2, \cdots, m-1$, gives:

$$\frac{\partial K(\mathbf{x}_i,\mathbf{x}_j)}{\partial \sigma_l} = \alpha_l \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_l^2}\right)\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma_l^3} \tag{7}$$

The partial derivative with respect to $\sigma_m$ is:

$$\frac{\partial K(\mathbf{x}_i,\mathbf{x}_j)}{\partial \sigma_m} = \left(1 - \sum_{l=1}^{m-1}\alpha_l\right)\exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_m^2}\right)\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma_m^3} \tag{8}$$

1. Initialize the vector of parameters $\boldsymbol{\theta}$ to an initial vector value $\boldsymbol{\theta}_0$.
2. Compute the gradients $\frac{\partial F_{kernel}}{\partial \theta_i}$ according to equations (7)–(9).
3. Update the components $\theta_i$ of the parameter vector $\boldsymbol{\theta}$ according to the rule:
$$\theta_i^{(new)} = \theta_i^{(old)} + \rho \frac{\partial F_{kernel}}{\partial \theta_i}$$
4. Go to step (2) until the maximal value of $F_{kernel}$ is achieved.
5. Output $\boldsymbol{\theta}$.

Figure 1: Optimization Algorithm

The partial derivatives of $K$ with respect to the parameters $\alpha_l$, for $l = 1, 2, \cdots, m-1$, are as follows:

$$\frac{\partial K(\mathbf{x}_i, \mathbf{x}_j)}{\partial \alpha_l} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_l^2}\right) - \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma_m^2}\right) \tag{9}$$

We maximize the objective function $F_{kernel}$ by performing gradient ascent in parameter space $\boldsymbol{\theta}$ using equations (7)–(9):

$$\theta_i^{(new)} = \theta_i^{(old)} + \rho \frac{\partial F_{kernel}}{\partial \theta_i}$$

where $\rho$ is a scalar parameter that determines the length of the step at each iteration; $\rho$ is optimized via a line-search method. The resulting gradient ascent algorithm is summarized in Figure 1. The algorithm provides in output the optimal parameters $\boldsymbol{\theta}$, which combined with equation (2) give the optimal kernel. Such optimal kernel can then be utilized in conjunction with a kernel-based clustering procedure to determine a partition of the data.

## 4 Experimental Evaluation

### 4.1 Datasets

We performed experiments on one simulated dataset and four real datasets. (1) The simulated dataset contains two clusters in two dimensions distributed as concentric circles (See Figure 2(a)). Each cluster contains 200 points. (2) **Digits**: This dataset is the pendigits handwritten character recognition dataset from the UCI repository [5]. 10% of the data was chosen randomly from the four classes $\{3, 6, 8, 9\}$. This gives in 423 points and 16 dimensions. (3) **Ionosphere**: This dataset is also from the UCI repository [5]. It was collected by a radar system in Goose Bay, Labrador. The targets were free electrons in the ionosphere. "Good" radar returns are those showing evidence of some type of structure in the ionosphere. "Bad" returns are those that do not. The dataset has 34 features. (4) **Vowel**: This dataset concerns the recognition of eleven steady state vowels of British English, using a specified training set of lpc derived log area ratios[1]. Three class corresponding to the vowels "i", "I", and "E" were chosen, for a total of 126 points and 10 dimensions; (5) **Wine**: This dataset is from UCI repository [5]. It results from a chemical analysis of wines grown in the same region in Italy, but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

### 4.2 Evaluation Criterion

To evaluate the clustering results, we use the Rand Statistic index [15, 19, 17]. The Rand Statistic is an external cluster validity measure that estimates the quality of the clustering results with respect to the underlying classes of the data. Let $P_1$ be the partition of the data $X$ after applying a clustering algorithm, and $P_2$ be the underlying class structure of the data. We refer to a pair of points $(\mathbf{x}_u, \mathbf{x}_v) \in X \times X$ from the data using the following terms:

[1] http://www-stat-class.stanford.edu/~tibs/ElemStatLearn/

- $SS$: if both points belong to the same cluster of $P_1$ and to the same group of the underlying class structure $P_2$.
- $SD$: if the two points belong to the same cluster of $P_1$ and to different groups of $P_2$.
- $DS$: if the two points belong to different clusters of $P_1$ and to the same group of $P_2$.
- $DD$: if both points belong to different clusters of $P_1$ and to different groups of $P_2$.

Assume now that $N_{SS}, N_{SD}, N_{DS}$ and $N_{DD}$ are the number of $SS, SD, DS$ and $DD$ pairs respectively, then $N_{SS} + N_{SD} + N_{DS} + N_{DD} = N_{Pair}$ which is the maximum number of all pairs in the data set[2]. The Rand Statistic index measures the degree of similarity between $P_1$ and $P_2$ as follows:

$$RandStatistic = (N_{SS} + N_{DD})/N_{Pair} \tag{10}$$

### 4.3 Results and Discussion

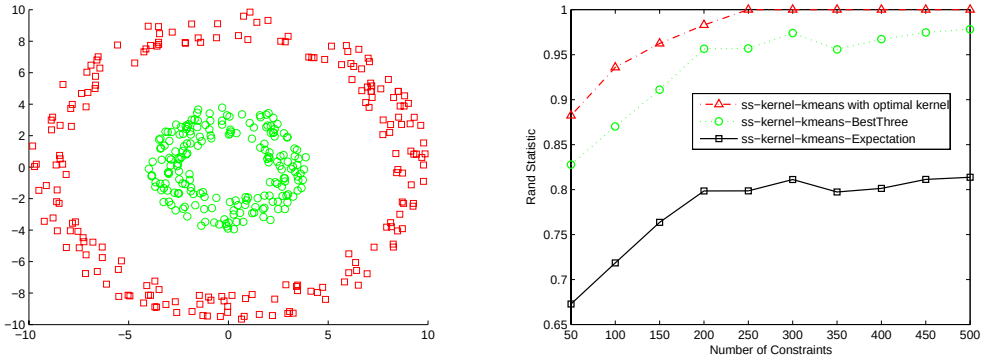
To evaluate the effectiveness of our proposed optimal kernel computation, we combine it with the SS-Kernel-KMeans technique introduced in [13], and described in Section 2.2. We call the combined approach SS-Optimal-Kernel-KMeans. In the experiments, we consider the linear combination of three Gaussian kernels, i.e. we set $m = 3$ in equation (2). We found that three kernel functions provide enough flexibility and modeling capability, while keeping the number of parameters to be estimated low. We initialize the weights $\alpha_i$ to equal values, i.e. $\alpha_i = \frac{1}{3}$ for $i = 1, 2, 3$. The spread parameters $\sigma_i$ are initialized to the values $r_i S$, where $r_i$ is a random number between 0 and 1, and $S$ is the average of the standard deviations of the data computed over each dimension.

We compare our method with SS-Kernel-KMeans itself. As shown in [13], SS-Kernel-KMeans requires in input a predefined value for the Gaussian kernel parameter $\sigma$. In absence of labeled data, parameters cannot be cross-validated; thus, we estimate the expected accuracy of SS-Kernel-KMeans by averaging the resulting clustering quality over multiple runs for different values of $\sigma$. Specifically, in our experiments, we test the SS-Kernel-KMeans algorithm with the values of $\sigma^2$: 0.1, 1, 10, 100, 1000, 10000. We report the average Rand Statistic achieved over the six $\sigma$ values, as well as the average over the best three performances achieved, in order to show the advantage of our technique also in this latter case. The violation costs $w_{ij}$ and $\overline{w}_{ij}$ in equation (1) are set to $\frac{N}{kC}$, as indicated in [13], where $N$ is the number of data points, $k$ is the number of clusters, and $C$ is the total number of constraints. The value of $k$ is set to the actual number of classes in the data.

Figures 2-4 show the learning curves using 20 runs of 2-fold cross-validation for each data set (30% for training and 70% for testing). These plots show the improvement in clustering quality on the test set as a function of an increasing amount of pairwise constraints. To study the effect of constraints in clustering, 30% of the data was randomly drawn as the training set at any particular fold, and the constraints are generated only using the training set. The clustering algorithm was run on the whole data set, but we calculated the Rand Statistic only on the test set. Each point on the learning curve is an average of results over 20 runs.

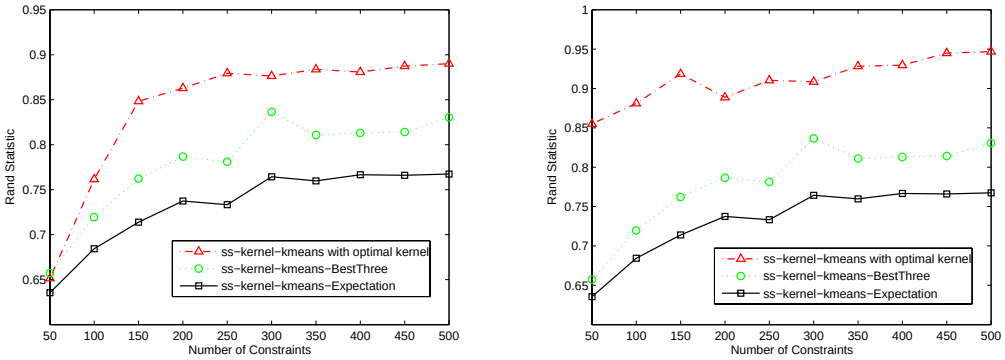
The results shown in Figures 2-4 clearly demonstrate the effectiveness of the SS-Optimal-Kernel-KMeans approach. For all five datasets, the clustering quality achieved by the SS-Optimal-Kernel-KMeans significantly outperforms the results provided by SS-Kernel-KMeans, averaged over the $\sigma$ values tested. In addition, the SS-Kernel-KMeans with the optimal kernel also outperforms the average top three performances of SS-Kernel-KMeans. These results show that our technique is capable of estimating the optimal kernel parameter values from the given constraints. In particular, for the TwoConcentric data (see Figure 2(b)), the SS-Kernel-KMeans with the optimal kernel effectively uses the increased amount of constraints to learn a perfect separation of the two clusters. For the Digits, Ionosphere, and Wine data, the SS-Kernel-KMeans with the optimal kernel provides a clustering quality that is significantly higher than the one given by SS-Kernel-KMeans, even when a small amount of constraints is available. This behavior is very desirable since in practice only a limited amount of supervision might be available.

[2] $N_{Pair} = N(N-1)/2$, where $N$ is the total number of points in the data set.



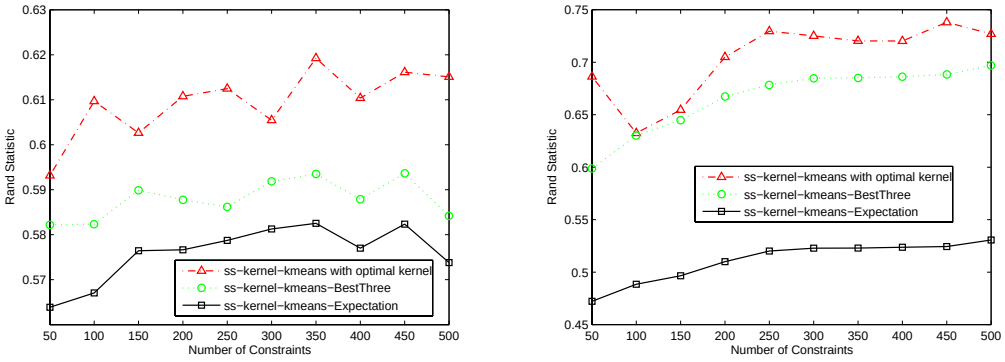

(a) (b)

Figure 2: (a) TwoConcentric data (b) Clustering result on TwoConcentric data




(a) (b)

Figure 3: (a) Clustering result on Vowel data (b) Clustering result on Digits data




(a) (b)

Figure 4: (a) Clustering result on Ionosphere data (b) Clustering result on Wine data

## 5 Conclusions and Future Work

We proposed a new methodology to optimize the parameters of a composite Gaussian kernel for semi-supervised kernel-based clustering algorithms. Our optimization strategy can be combined with an arbitrary kernel-based clustering approach. In this work, we have tested the optimal kernel function computed by our technique in combination with a semi-supervised kernel-Kmeans algorithm recently introduced. The experimental results demonstrate that our technique enables the practical utilization of powerful kernel-based semi-supervised clustering approaches by providing a mechanism to automatically and effectively set the involved critical parameters. In our future work we will test the optimal kernel in combination with other kernel-based clustering techniques, and study the effect of varying the number of components in the composite kernel as a function of the number of contraints available. An additional avenue to explore is the use of active learning as a methodology to generate constraints which are most informative.

## References


[1] Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning distance functions using equivalence relations. International Conference on Machine Learning, 2003.

[2] Basu, S., Bilenko, M., Mooney, R. J.: A probabilistic framework for semi-supervised clustering. International Conference on Knowledge Discovery and Data Mining, 2004.

[3] Besag, J.: On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society, Series B (Methodological), 1986.

[4] Bilenko, M., Basu, S., Mooney, R. J.: Integrating constraints and Metric Learning in semi-supervised clustering. International Conference on Machine Learning, 2004.

[5] Blake, C. L., Merz, C. J.: UCI repository of machine learning databases. http://www.ics.uci.edu/ mlearn/MLRepository.html, 1998.

[6] Boykov, Y., Veksler, O., Zabih, R.: Markov Random fields with efficient approximations. IEEE Computer Vision and pattern Recognition Conference, 1998.

[7] Chapelle, O., Vapnik, V.: Choosing Mutiple Parameters for Support Vector Machines. Machine Learning Vol.46, No. 1. pp.131-159, 2002.

[8] Cohn, D., Caruana, R., McCallum, A.: Semi-supervised clustering with user feedback. TR2003-1892, Cornell University, 2003.

[9] Cristianini, N., Shawe-Taylor, J., Elisseeff, A.: On Kernel-Target Alignment, Neural Information Processing Systems (NIPS), 2001.

[10] Huang, J., Yuen, P. C., Chen, W. S., Lai, J. H.: Kernel Subspace LDA with optimized Kernel Parameters on Face Rcognition. The sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004.

[11] Shawe-Tayor, J., Cristianini, N.: Kernel methods for pattern analysis. Cambrige University, 2004.

[12] Kleinberg, J., Tardos, E.: Approximation algorithms for classification problems with pairwise relationships: metric labeling and Markov random fields. The 40th IEEE Symposium on Foundation of Computer Science, 1999.

[13] Kulis, B., Basu, S., Dhillon, I., Moony, R.: Semi-supervised graph clustering: a kernel approach. International Conference on Machine Learning, 2005.

[14] Segal, E., Wang, H., Koller, D.: Discovering molecular pathways from protein interaction and gene expression data. Bioinformatics, 2003.

[15] Theodoridis, S., Koutroubas, K.: Pattern Recognition. Academic Press, 1999.

[16] Vapnik., V.: The Nature of Statistical Learning Theory, Wiley, New York, USA, 1995.

[17] Wagstaff, K., Cardie, C., Rogers, S., Schroedl, S.: Constrained K-Means clustering with background knowledge. International Conference on Machine Learning, 2001.

[18] Wang, W., Xu, Z., Lu W., Zhang, X.: Determination of the spread parameter in the Gaussian Kernel for classification and regression. Neurocomputing, Vol. 55, No. 3, 645, 2002.

[19] Xing, E. P., Ng, A. Y., Jordan, M. I., Russell, S.: Distance metric learning, with application to clustering with side-information. Advances in Neural Information Processing Systems 15, 2003.

[20] Zhang, Y., Brady, M., Smith, S.: Hidden Markov random field model and segmentation of brain MR images. IEEE Transactions on Medical Imaging, 2001.