Clustering by Maximizing Sum-of-Squared Separation Distance

Yixin Chen*† Jinbo Bi‡

Abstract

Maximizing the separating margin is crucial for the good generalization performance of Support Vector Machines (SVMs). Analogous to the definition of separation distance or separating margin in SVMs, we propose a definition on separation distance in clustering tasks when a hyperplane is used to separate clusters. For given training data and a given metric distance, by maximizing the proposed separation distance, our clustering algorithm constructs an "optimal" hyperplane that can be applied to unseen data in the future. The resulting hyperplane corresponds to a nonlinear decision boundary in the input feature space through an appropriate distance feature mapping. A graph-theoretic perspective of the proposed method is discussed. In particular, we show that, under certain conditions, the proposed clustering algorithm is equivalent to a spectral relaxed graph cut. Extensive experimental results are provided to validate the method.

Keywords: biclustering, optimization, graph partitioning, spectral relaxation, spectral clustering.

1 Introduction and Overview.

As an important branch in unsupervised learning, cluster analysis aims at partitioning a collection of objects into groups or "clusters" so that members within each cluster are more closely related to one another than objects assigned to different clusters [12]. In a variety of areas including bio-informatics, computer vision, information retrieval, data mining, and VLSI design, clustering algorithms provide automated tools to help identify a structure from an unlabeled data set. There is a rich resource of prior work on this subject. The works reviewed below are most related to ours, which by no means represent a comprehensive list.

1.1 Related Work. Depending on the underlying model assumption, clustering algorithms roughly fall into two categories: *generative approach* and *discriminative approach*.

A generative clustering algorithm supposes that the data are independent and identically distributed samples generated from an unknown probability density function. This density function is usually parameterized by a mixture model: weighted sum of a collection of component density functions, each of which characterizes one of the clusters. Consequently, clustering turns into a density estimation problem, which is commonly tackled by Expectation Maximization (EM) algorithm [6]. However, in practice there is usually no *a priori* knowledge about the parametric forms of component density functions. In many applications the Gaussian assumption does not lead to satisfactory performance. Moreover, estimation techniques, such as EM algorithm, only guarantee local optimality. Nonetheless, generative clustering techniques have several advantages, such as the scalability to large data sets [4] and the ability to handle examples outside the training set.

A discriminative clustering algorithm works directly on the training data without explicitly assuming an underlying probability model. Each training sample is assigned to one and only one of the clusters. A "loss" function is defined over all possible assignments. It measures the degree to which the clustering goal is met. Optimal cluster assignments for all training samples are achieved by optimizing the loss function. Since this optimization problem is essentially combinatorial, discriminative clustering algorithms are also called *combinatorial clustering algorithms* [14]. Combinatorial optimization is straightforward in principle: searching all possible assignments. Unfortunately, this is feasible only for very small data sets since the number of distinct assignments is $O(k^n)$, where k is the number of clusters and n is the sample size. Therefore, practical discriminative clustering algorithms typically seek for a trade-off between optimality and computational complexity. For example, the k-means [13] and k-medoids algorithms [16] use an iterative greedy descent strategy to search for a suboptimal partition. Agglomerative (divisive) clustering methods [16] generate a hierarchy of clusters via recursively merging (splitting) clusters according to certain greedy heuristics. Spectral clustering [20, 8, 17] formulates clustering as a graph partitioning problem. The

*Department of Computer Science, University of New Orleans, New Orleans, LA 70148. Email: yixin@cs.uno.edu

†The Research Institute for Children, 200 Henry Clay Avenue, Research & Education, New Orleans, LA 70118

‡Computer-Aided Diagnosis & Therapy Group, Siemens Medical Solutions, Inc., 51 Valley Stream Parkway, Malvern, PA 19355. Email: jinbo.bi@siemens.com

optimal partition is approximated by eigenvectors of a properly normalized affinity matrix of the graph. The relationships between spectral partitioning methods and kernel k-means are discussed in [7].

Interesting connections between generative and discriminative approaches have been discussed in [2]. The equivalence between a class of generative and discriminative clustering algorithms were established based on Bregman Divergence loss function [2]. Unlike generative approaches, which can predict on examples outside the training set, many discriminative clustering algorithms, including those mentioned above, cannot do so without rerunning the algorithm. In a recent work by Bengio *et al.*, a family of discriminative clustering methods were extended to deal with "out-of-sample" examples [3].

The work proposed in this paper is a new discriminative clustering algorithm based on a loss function motivated by the separating margin of Support Vector Machines (SVM) [21]. An overview is provided below.

1.2 Overview of Our Approach. In the past decade, SVM has become an effective and robust tool for solving supervised classification problems. Loosely speaking, SVM finds an "optimal" hyperplane, in a kernel-induced feature space, which separates two classes of samples with the maximal margin. The characteristics of SVM can be summarized as follows:
1. maximizing a separation distance, i.e., the so-called separating margin, and
2. applying appropriate feature mappings, i.e., the kernel mapping.
By a kernel mapping, SVM is able to construct nonlinear models using a linear learning mechanism. The margin concept provides a theoretical basis for SVM since maximizing margin is related to minimizing upper bounds on the generalization error [21]. This paper proposes a novel clustering algorithm aiming to make analogous uses of the above two well-established characteristics of SVM in unsupervised learning, in particular, cluster analysis.

From a basic rationale of clustering, members of different clusters should be as dissimilar as possible. In terms of a linear separating boundary, intuitively, a good bipartition should divide the samples into two groups, and put them away from the separating hyperplane as far as possible. In SVM, the separating margin of two classes is defined as the shortest distance from the vectors in either of the two classes to the separating hyperplane [1]. Figure 1 illustrates the definition. Let L

[1] The margin defined here refers to the "hard margin" when separation is perfect. Margin can also be defined when perfect separation is impossible or undesirable [21]. It is then called the "soft margin."

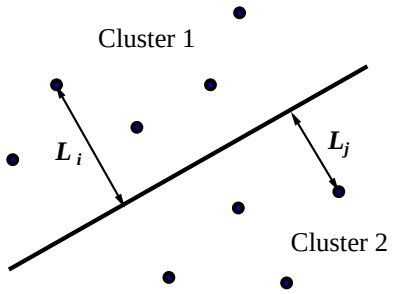

Figure 1: The separation distance definition.

denote the geometric distance from an example to the hyperplane. Assume indices i and j run through examples from cluster 1 and cluster 2, respectively. The margin can be written as:

$$\text{Margin} = \min_i L_i + \min_j L_j \ .$$

The use of margin was derived from VC theory for supervised learning tasks. Unfortunately, maximizing the margin on all examples with unknown labels in unsupervised learning is an NP-complete problem. The evaluation of margin varies for different possible label assignments. Hence we propose a different separation distance, which we call the sum-of-squared (SS) separation distance and is defined:

$$\text{SS} = \sum_i L_i^2 + \sum_j L_j^2 = \sum_t L_t^2$$

where t runs through all training examples. We will show that the sum-of-squared separation distance (SS) is much easier to evaluate and maximize, and maximizing SS produces neat properties similar to those obtained for spectral clustering.

For a nontrivial training set, there usually exist cluster assignments under which the samples are not linearly separable in the input space. Therefore it is possible that some bipartitions, which may correspond to good clustering, could not be realized by hyperplanes in the input space. This is certainly undesirable since the model itself may have already eliminated potentially good bipartitions. To avoid such an intrinsic deficiency, nonlinear feature mapping is adopted so that a hyperplane constructed in the feature space corresponds to a nonlinear model in the original input space, such as the kernel mapping used in SVMs. Notice that when perfect separation is impossible in supervised learning, a "soft margin" can be defined in terms of the given class labels. With unknown labels in unsupervised learning, no way exists to define a soft margin. Bearing this in mind, we propose a simple feature mapping, based on the given data set and the distance (or dissimilarity)

measure, which maps input vectors to a new feature space where samples are always linearly separable.

The proposed algorithm, Maximal Separation Clustering (MSC), possesses several properties:

- MSC only requires the knowledge of a *metric distance* function measuring the dissimilarity between samples with an assumption that prior knowledge can be properly incorporated in the chosen metric distance. For applications where a similarity function is specified, a transformation is needed to map the similarity into a metric distance. The transformation will be discussed in Section 2.1. Note that a metric distance function is a more general concept than a positive-definite (PD) kernel function since as long as an inner product (the PD kernel) is given, the corresponding metric distance can be induced, but not vice versa.

- Although MSC is a discriminative approach, the algorithm can predict labels for out-of-sample examples because it learns an optimal separating hyperplane that transforms to a nonlinear decision boundary in the input space.

- MSC has interesting connections with a class of graph partitioning methods. Specifically, the optimal bipartition generated by the algorithm can be viewed as an approximation of an "optimal" graph partition under spectral relaxation.

- MSC can be formulated as an eigenvalue decomposition problem similar to spectral clustering under certain circumstances. Hence it does not require any integer programming solvers.

1.3 Outline of the Paper. The remainder of the paper is organized as follows: In Section 2, we first introduce a feature mapping such that any bipartition of a given training set can be achieved by a hyperplane in the new feature space. Clustering is then formulated as maximizing the sum-of-squared separation distance in the feature space. In Section 3, we propose a class of graph partitioning problems and prove that under certain conditions the optimal bipartition given by a hyperplane is the solution of a graph partitioning problem with spectral relaxation. Section 4 describes the experiments we have performed and provides the results. We conclude in Section 5, together with a discussion of future work.

2 Maximal Separation Clustering.

We focus on problems of clustering examples into two clusters. For applications requiring more than two clusters, the proposed method can be recursively applied, but only local optimality is ensured.

2.1 Distance Feature Mapping. Given a set of n distinct training samples $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{X} : i = 1, ..., n\}$ and a metric distance function [2] $m : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$, we consider linear separation functions:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{d}(\mathbf{x}) \tag{2.1}$$

where $\mathbf{d} : \mathbb{X} \to \mathbb{R}^m$ realizes a set of features induced by the metric distance m and $\mathbf{w}$ are the model parameters. An example $\mathbf{x}_i$ is assigned into one cluster if $f(\mathbf{x}_i) \geq 0$; the other cluster if $f(\mathbf{x}_i) < 0$.

Note that the metric distance m can be any suitable metric and does not have to be the Euclidean distance in $\mathbb{X}$. Since the distance function m provides the only prior knowledge about clustering, $m(\mathbf{x}_i, \mathbf{x}_j)$ should be fully explored in the feature mapping $\mathbf{d}$. We propose to map any input vector $\mathbf{x}$ to an n dimensional space where the j^{th} dimension represents $d_j(\mathbf{x}) = m(\mathbf{x}, \mathbf{x}_j)$.

The mapping $\mathbf{d}_{\mathcal{X}} : \mathbb{X} \to \mathbb{R}^n$ can be written as:

$$\mathbf{d}_{\mathcal{X}}(\mathbf{x}) = \begin{bmatrix} m(\mathbf{x}, \mathbf{x}_1) \\ m(\mathbf{x}, \mathbf{x}_2) \\ \vdots \\ m(\mathbf{x}, \mathbf{x}_n) \end{bmatrix}.$$

Clearly, $\mathbf{d}_{\mathcal{X}}$ is data dependent. Now we validate if the training samples are linearly separable for all possible label assignments in the new feature space generated by $d_{\mathcal{X}}$. Let us first stack n cluster assignments made by (2.1) into a matrix equation:

$$\begin{bmatrix} f(\mathbf{x}_1) \\ f(\mathbf{x}_2) \\ \vdots \\ f(\mathbf{x}_n) \end{bmatrix} = \mathbf{D}\mathbf{w} \tag{2.2}$$

where

$$\mathbf{D} = \begin{bmatrix} m(\mathbf{x}_1, \mathbf{x}_1) & m(\mathbf{x}_1, \mathbf{x}_2) & \cdots & m(\mathbf{x}_1, \mathbf{x}_n) \\ m(\mathbf{x}_2, \mathbf{x}_1) & m(\mathbf{x}_2, \mathbf{x}_2) & \cdots & m(\mathbf{x}_2, \mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ m(\mathbf{x}_n, \mathbf{x}_1) & m(\mathbf{x}_n, \mathbf{x}_2) & \cdots & m(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix}$$

is called the *distance matrix* or the *dissimilarity matrix*. $\mathbf{D}$ is symmetric and nonnegative (all elements are greater than or equal to zero.) For Euclidean distance $m(\cdot, \cdot)$, $\mathbf{D}$ is always invertible with one positive eigenvalue and $n - 1$ negative eigenvalues [18]. It has been

[2] A metric distance function, $m(\cdot, \cdot)$, is a nonnegative function satisfying: 1) $m(\mathbf{x}, \mathbf{y}) \geq 0$, and $m(\mathbf{x}, \mathbf{y}) = 0$ iff $\mathbf{x} = \mathbf{y}$; 2) $m(\mathbf{x}, \mathbf{y}) = m(\mathbf{y}, \mathbf{x})$; and 3) $m(\mathbf{x}, \mathbf{y}) + m(\mathbf{y}, \mathbf{z}) \geq m(\mathbf{x}, \mathbf{z})$.

proved that this property holds for an arbitrary metric distance $m(\cdot,\cdot)$ as well [1]. Consequently, for an arbitrary label assignment $[f(\mathbf{x}_1), f(\mathbf{x}_2), \cdots, f(\mathbf{x}_n)]^T$, equation (2.2) has a unique solution of $\mathbf{w}$. This implies that $\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)$'s are always linearly separable.

In some clustering tasks, a distance metric is not given directly. Instead, a similarity measure is specified. A commonly used class of similarity measures is defined by PD kernels [20, 17, 19]. A PD kernel, $K : \mathbb{X}\times\mathbb{X} \to \mathbb{R}$, computes the inner product in a kernel-induced feature space $\mathcal{H}$ via a mapping $\Phi : \mathbb{X} \to \mathcal{H}$, i.e., $K(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T\Phi(\mathbf{x}')$. Correspondingly,

$$\begin{aligned} m(\mathbf{x}, \mathbf{x}') &= \sqrt{[\Phi(\mathbf{x}) - \Phi(\mathbf{x}')]^T [\Phi(\mathbf{x}) - \Phi(\mathbf{x}')]} \\ &= \sqrt{K(\mathbf{x}, \mathbf{x}) + K(\mathbf{x}', \mathbf{x}') - 2K(\mathbf{x}, \mathbf{x}')} \end{aligned} \tag{2.3}$$

defines a metric distance in $\mathcal{H}$. This suggests that $\mathbf{d}_{\mathcal{X}}$ and $\mathbf{D}$ can also be constructed from a PD kernel. Therefore, for the rest of the paper, we assume that either a metric distance or a similarity measure (described by a PD kernel) is given.

2.2 Computing the Optimal Partition. In the space where $\mathbf{d}_{\mathcal{X}}(\mathbf{x})$ resides, the decision boundary of the separation function $f(\mathbf{x}) = \mathbf{w}^T\mathbf{d}_{\mathcal{X}}(\mathbf{x})$ is a hyperplane defined by $\mathbf{w}^T\mathbf{d}_{\mathcal{X}} = 0$. The geometric distance from a point $\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)$ to the hyperplane is $L_i = \frac{|\mathbf{w}^T\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)|}{\|\mathbf{w}\|}$ where $||\cdot||$ denotes the 2-norm of a vector unless otherwise stated. If we assume the separation function is normalized such that $\|\mathbf{w}\| = 1$, then the above geometric distance can be simply calculated as $L_i = |\mathbf{w}^T\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)|$. The term inside $|\cdot|$ gives the signed distance. The hence proposed *sum-of-squared separation distance* is calculated as follows by taking all samples in $\mathcal{X}$ into consideration:

$$\sum_{i=1}^{n} \left[\mathbf{w}^T\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)\right]^2 = \mathbf{w}^T\mathbf{D}^2\mathbf{w}. \tag{2.4}$$

The optimal clustering corresponds to a unit length $\mathbf{w}$ that separates two clusters with maximal value of (2.4).

It is not difficult to see that the unit length eigenvector $\mathbf{p}$ corresponding to the largest eigenvalue λ_1 of $\mathbf{D}^2$, maximizes (2.4). Unfortunately, the resulting separation function, $f(\mathbf{x}) = \mathbf{d}_{\mathcal{X}}(\mathbf{x})^T\mathbf{p}$, assigns all training samples in $\mathcal{X}$ to one cluster because $\mathbf{Dp} = \lambda_1\mathbf{p}$ has components either all positive or all negative. This is due to the positivity of $\mathbf{D}^2$. The distance matrix $\mathbf{D}$ is a symmetric and non-negative matrix, so the eigenvalue decomposition exists. The corresponding $\mathbf{D}^2$ is thus a positive matrix [3] which has the same eigenvectors as those of $\mathbf{D}$ and eigenvalues equal to the square of the eigenvalues of $\mathbf{D}$. A positive matrix also has the following properties(Ch.8.2, [15]):

- The largest eigenvalue λ_1 is positive with algebraic multiplicity 1;
- The corresponding eigenvector satisfies either $\mathbf{p} > 0$ (called the Perron vector if $\mathbf{p}^T\mathbf{e} = 1$ where $\mathbf{e}$ is a vector of 1's) or $\mathbf{p} < 0$ [4]. All other eigenvectors have elements with mixed signs.

To avoid trivial clustering like $\mathbf{p}$, the following constraint is imposed:

$$\boldsymbol{\alpha}^T\mathbf{Dw} = 0\ .$$

Here $\boldsymbol{\alpha} > 0$ is a user-specified normalized weight vector satisfying $\boldsymbol{\alpha}^T\mathbf{e} = 1$. Since $\mathbf{Dw}$ contains the signed distances (cluster membership is indicated by the sign), the above constraint enforces that the weighted summation of signed distances is zero. In other words, neither $\mathbf{Dw} > 0$ nor $\mathbf{Dw} < 0$ is allowed. The clustering problem is then formulated as below.

Definition 2.1. (Maximal Separation Clustering) *Given a distance matrix* $\mathbf{D}$ *and a normalized weight vector* $\boldsymbol{\alpha} > 0$ *(*$\boldsymbol{\alpha}^T\mathbf{e} = 1$*), an optimal clustering is given by a separation function* $f(\mathbf{x}) = \mathbf{w}^{*T}\mathbf{d}_{\mathcal{X}}(\mathbf{x})$ *where*

$$\mathbf{w}^* = \underset{\substack{\mathbf{w},\ \|\mathbf{w}\|=1 \\ \boldsymbol{\alpha}^T\mathbf{Dw}=0}}{\operatorname{argmax}} \quad \mathbf{w}^T\mathbf{D}^2\mathbf{w}\ .$$

Since the weight vector should be chosen by a user beforehand, a few insights about different choices of $\boldsymbol{\alpha}$ will be helpful. Three interesting choices are discussed:

- $\boldsymbol{\alpha}_1 \propto \mathbf{e}$
  It assumes that all samples are equally important. Therefore, the signed distances should be weighted uniformly, i.e., $\boldsymbol{\alpha}_1 = \frac{\mathbf{e}}{\mathbf{e}^T\mathbf{e}}$.
- $\boldsymbol{\alpha}_2 \propto \mathbf{De}$
  A sample is weighted according to its overall dissimilarity to the rest of the samples. The weight for the signed distance $\mathbf{w}^T\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)$ is proportional to $\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)^T\mathbf{e} = \sum_{k=1}^{n} m(\mathbf{x}_i, \mathbf{x}_k)$, which is the summation of the distances between $\mathbf{x}_i$ and all samples in $\mathcal{X}$. After normalization, we get $\boldsymbol{\alpha}_2 = \frac{\mathbf{De}}{\mathbf{e}^T\mathbf{De}}$.
- $\boldsymbol{\alpha}_3 \propto \mathbf{D}\boldsymbol{\alpha}_3$
  This scheme carries a similar flavor as that of $\boldsymbol{\alpha}_2$. Here the weight for the signed distance $\mathbf{w}^T\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)$ is proportional to $\mathbf{d}_{\mathcal{X}}(\mathbf{x}_i)^T\boldsymbol{\alpha}_3$, which is a *weighted*

[3] A matrix is positive if all it elements are positive [15].

[4] For a matrix or a vector $\mathbf{A}$, we write $\mathbf{A} > 0$ if all it elements are positive. This notation can be generalized to $\geq$, $<$, and $\leq$.

summation of the distances between $\mathbf{x}_i$ and all samples in $\mathcal{X}$. In this case, $\boldsymbol{\alpha}_3$ has to be an eigenvector of $\mathbf{D}$, i.e., $\lambda\boldsymbol{\alpha}_3 = \mathbf{D}\boldsymbol{\alpha}_3$. Hence the Perron vector $\mathbf{p} > 0$ (satisfying $\mathbf{p}^T\mathbf{e} = 1$) of $\mathbf{D}$ is the only choice.

Empirical comparisons of the above weighting strategies are provided in Section 4.

THEOREM 2.1. *The optimal solution* $\mathbf{w}^*$ *of the MSC problem described in Definition 2.1 is*

$$\mathbf{w}^* = \mathbf{U}_{\boldsymbol{\alpha}}\mathbf{v}, \tag{2.5}$$

where $\mathbf{U}_{\boldsymbol{\alpha}} \in \mathbb{R}^{n\times(n-1)}$ *is a matrix whose columns form an orthonormal basis of the null space of* $\boldsymbol{\alpha}^T\mathbf{D}$*, and* $\mathbf{v}$ *is a unit length eigenvector corresponding to the largest eigenvalue of* $\mathbf{U}_{\boldsymbol{\alpha}}^T\mathbf{D}^2\mathbf{U}_{\boldsymbol{\alpha}}$.

Proof. The feasible region of the optimization problem in Definition 2.5 is

$$\mathcal{F} = \{\mathbf{w} \in \mathbb{R}^n : \|\mathbf{w}\| = 1, \boldsymbol{\alpha}^T\mathbf{D}\mathbf{w} = 0\} .$$

It is not difficult to check that $\mathcal{F}$ can be equivalently written as

$$\mathcal{F} = \{\mathbf{w} = \mathbf{U}_{\boldsymbol{\alpha}}\mathbf{z} : \mathbf{z} \in \mathbb{R}^{n-1}, \|\mathbf{z}\| = 1\} .$$

So the original optimization problem becomes

$$\begin{aligned} \mathbf{z}^* &= \arg\max_{\|\mathbf{z}\|=1} \mathbf{z}^T\mathbf{U}_{\boldsymbol{\alpha}}^T\mathbf{D}^2\mathbf{U}_{\boldsymbol{\alpha}}\mathbf{z} \\ \mathbf{w}^* &= \mathbf{U}_{\boldsymbol{\alpha}}\mathbf{z}^* . \end{aligned}$$

Then (2.5) follows from the fact that $\mathbf{z}^* = \mathbf{v}$. □

2.3 An Algorithmic View.

ALGORITHM 2.1. (Maximal Separation Clustering)

Input: Set $\mathcal{X} = \{\mathbf{x}_i \in \mathbb{R}^d : i = 1, ..., n\}$, a metric distance function $m(\cdot,\cdot)$, and weight $\boldsymbol{\alpha}$.
Output: Separation function $f(\mathbf{x}) = \mathbf{w}^{*T}\mathbf{d}_{\mathcal{X}}(\mathbf{x})$ where the sign of $f(\mathbf{x})$ defines the cluster assignment of $\mathbf{x}$.
Method:

1 Compute the $n \times n$ distance matrix $\mathbf{D}$
2 Find an orthonormal basis of the null space of $\boldsymbol{\alpha}^T\mathbf{D}$ and stack them as columns of $\mathbf{U}_{\boldsymbol{\alpha}}$
3 Compute $\mathbf{z}^*$, an eigenvector corresponding to the largest eigenvalue of $\mathbf{U}_{\boldsymbol{\alpha}}^T\mathbf{D}^2\mathbf{U}_{\boldsymbol{\alpha}}$
4 $\mathbf{w}^* \leftarrow \mathbf{U}_{\boldsymbol{\alpha}}\frac{\mathbf{z}^*}{\|\mathbf{z}^*\|}$
5 OUTPUT $f(\mathbf{x}) = \mathbf{w}^{*T}\mathbf{d}_{\mathcal{X}}(\mathbf{x})$

Note that the input metric distance can be constructed from a PD kernel based on (2.3). The orthonormal basis of the null space of $\boldsymbol{\alpha}^T\mathbf{D}$ is computed using the Symmetric QR Algorithm (Ch.8.3, [10]). Therefore the computational cost of Step 2 is $O(n^3)$. The eigenvalue problem in Step 3 is solved by a Lanczos method (Ch.9, [10]). The running time of a Lanczos method is $O(kn^2)$ where k is the maximum number of matrix-vector computations required. Usually, k is much smaller than n. So the overall running time is $O(n^3 + kn^2)$. It is worth mentioning a special case where $\boldsymbol{\alpha}$ is given as the Perron vector of $\mathbf{D}$, i.e., $\boldsymbol{\alpha} = \boldsymbol{\alpha}_3$. Then one can show that $\mathbf{w}^*$ is a unit length eigenvector corresponding to the second largest eigenvalue of $\mathbf{D}^2$ (a proof will be given in Section 3.3). In this case, the computational cost becomes $O(kn^2)$ because there is no need to compute the null space of $\boldsymbol{\alpha}^T\mathbf{D}$.

3 Connections with Graph Partitioning.

This section provides a graph-theoretic view of the MSC method. We first propose a new graph-theoretic criterion for measuring the goodness of graph bipartition. A graph partitioning problem divides vertices into groups so that the between-group dissimilarity is high, and/or within-group dissimilarity is low. The novel criterion measures the disparity between the between-group dissimilarity and the within-group dissimilarity. Thus we name the resulting bipartition – the *disparity cut*. The maximization of the criterion can be formulated as an eigenvalue problem. Then connections between Algorithm 2.1 and the disparity cut are established.

3.1 Disparity Cut Criterion.

Given a set of samples and a dissimilarity measure, one can construct a weighted undirected graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V} = \{1, 2, \ldots, n\}$ is the vertex set, $\mathbf{E} = \{(i, j) : i, j \in \mathbf{V}\}$ is the set of edges. The vertices represent the samples. An edge is formed between every pair of vertices. The weight d_{ij} of an edge (i, j) indicates the similarity or dissimilarity between vertices i and j. The weights can be organized into an *affinity matrix*, $\mathbf{D}$. To be consistent with notations in Section 2, we assume that d_{ij} describes the dissimilarity. Note that a similarity measure can be converted to a dissimilarity measure via (2.3).

Let sets $\mathbf{A}, \mathbf{B} \subset \mathbf{V}$. In graph theory, a *cut* is defined as:

$$cut(\mathbf{A}, \mathbf{B}) = \sum_{i\in\mathbf{A}, j\in\mathbf{B}} d_{ij} .$$

Finding a bipartition of the graph that minimizes this cut value is known as the minimum cut problem [5]. Efficient algorithms exist for solving this problem. How-

[5] In the minimum cut problem, the affinity matrix is defined by vertex similarities. If affinity matrix captures vertex dissimilarities, as in this paper, the problem should be named the maximum cut problem instead.

ever the minimum cut criterion tends to group small sets of isolated nodes in the graph because the cut defined above does not contain any within-group information [20]. Many modified graph partition criteria have been proposed to produce more balanced partitions [11, 20, 8, 17]. Next, we introduce a new criterion, *disparity cut*.

Let each vertex i be associated with a positive weight, β_i. Without loss of generality we assume $||\boldsymbol{\beta}|| = 1$. For $\mathbf{A} \subset \mathbf{V}$, we define a *weighted cardinality* of $\mathbf{A}$, $|\mathbf{A}|_{\boldsymbol{\beta}}$, to be

$$|\mathbf{A}|_{\boldsymbol{\beta}} = \sum_{i \in \mathbf{A}} \beta_i^2 \ . \tag{3.6}$$

If weights are uniform (i.e., $\boldsymbol{\beta} = \mathbf{e}$), then (3.6) is identical to the standard definition of set cardinality. By taking vertex weights, $\boldsymbol{\beta}$, into consideration, we define a *weighted cut* as

$$cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{B}) = \sum_{i \in \mathbf{A}, j \in \mathbf{B}} \beta_i \beta_j d_{ij} \ .$$

It is not difficult to see that $cut_{\mathbf{e}}(\mathbf{A}, \mathbf{B}) = cut(\mathbf{A}, \mathbf{B})$. Let $\mathbf{A}$ and $\mathbf{B}$ form a bipartition of $\mathbf{V}$ (i.e., $\mathbf{A} \cap \mathbf{B} = \emptyset$, $\mathbf{A} \cup \mathbf{B} = \mathbf{V}$), the disparity cut, $Dcut(\mathbf{A}, \mathbf{B})$, is then defined as

$$\begin{aligned} Dcut(\mathbf{A}, \mathbf{B}) &= 2\, cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{B}) - \\ &\quad \frac{|\mathbf{B}|_{\boldsymbol{\beta}}}{|\mathbf{A}|_{\boldsymbol{\beta}}} cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{A}) - \frac{|\mathbf{A}|_{\boldsymbol{\beta}}}{|\mathbf{B}|_{\boldsymbol{\beta}}} cut_{\boldsymbol{\beta}}(\mathbf{B}, \mathbf{B}) \ . \end{aligned} \tag{3.7}$$

- $cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{B})$ measures the dissimilarity between vertex sets $\mathbf{A}$ and $\mathbf{B}$;
- $cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{A})$ and $cut_{\boldsymbol{\beta}}(\mathbf{B}, \mathbf{B})$ capture the vertex dissimilarities within $\mathbf{A}$ and within $\mathbf{B}$, respectively;
- $\frac{|\mathbf{B}|_{\boldsymbol{\beta}}}{|\mathbf{A}|_{\boldsymbol{\beta}}}$ and $\frac{|\mathbf{A}|_{\boldsymbol{\beta}}}{|\mathbf{B}|_{\boldsymbol{\beta}}}$ indicate the relative size of the two groups. An "unbalanced" bipartition will make one of the ratios a large number.

A "good" bipartition should generate two "balanced" groups that have high between-group dissimilarity and low within-group dissimilarity. This goal is achieved in this article by maximizing the disparity cut criterion. Finding the maximum disparity cut is NP-complete. Nevertheless, it is possible to find an approximation via spectral relaxation. This is described as below.

3.2 Spectral Relaxation. Given a weighted undirected graph $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$ with affinity matrix $\mathbf{D}$ and vertex weights $\boldsymbol{\beta}$, a bipartition of $\mathbf{V}$ into $\mathbf{A}$ and $\mathbf{B}$ can be defined by a *partition vector* $\mathbf{q} \in \{1, -1\}^n$ with elements

$$q_i = \begin{cases} 1, & i \in \mathbf{A} \ , \\ -1, & i \in \mathbf{B} \ . \end{cases}$$

Let Γ denote the diagonal matrix formed by the vertex weights $\boldsymbol{\beta}$, i.e., $\Gamma_{ii} = \beta_i$. Then we have the following identities:

$$\begin{aligned} cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{A}) &= \frac{1}{4} (\mathbf{e} + \mathbf{q})^T \Gamma \mathbf{D} \Gamma (\mathbf{e} + \mathbf{q}) \ , \\ cut_{\boldsymbol{\beta}}(\mathbf{B}, \mathbf{B}) &= \frac{1}{4} (\mathbf{e} - \mathbf{q})^T \Gamma \mathbf{D} \Gamma (\mathbf{e} - \mathbf{q}) \ , \\ cut_{\boldsymbol{\beta}}(\mathbf{A}, \mathbf{B}) &= \frac{1}{4} (\mathbf{e} + \mathbf{q})^T \Gamma \mathbf{D} \Gamma (\mathbf{e} - \mathbf{q}) \ . \end{aligned}$$

If we define the ratio of weighted cardinality, r, as

$$r = \frac{|\mathbf{A}|_{\boldsymbol{\beta}}}{|\mathbf{B}|_{\boldsymbol{\beta}}} \ ,$$

then (3.7) is equivalent to

$$\begin{aligned} Dcut(\mathbf{A}, \mathbf{B}) &= \frac{(\mathbf{e} + \mathbf{q})^T \Gamma \mathbf{D} \Gamma (\mathbf{e} - \mathbf{q})}{2} - \\ &\quad \frac{(\mathbf{e} + \mathbf{q})^T \Gamma \mathbf{D} \Gamma (\mathbf{e} + \mathbf{q})}{4r} - \frac{r (\mathbf{e} - \mathbf{q})^T \Gamma \mathbf{D} \Gamma (\mathbf{e} - \mathbf{q})}{4} \\ &= \frac{[(\mathbf{e} + \mathbf{q}) - r (\mathbf{e} - \mathbf{q})]^T \Gamma \mathbf{D} \Gamma [r (\mathbf{e} - \mathbf{q}) - (\mathbf{e} + \mathbf{q})]}{4r} \ . \end{aligned}$$

Let

$$\mathbf{y} = \frac{(\mathbf{e} + \mathbf{q}) - r (\mathbf{e} - \mathbf{q})}{2},$$

and $\mathbf{y}$ can be viewed as a generalized partition vector with elements

$$y_i = \begin{cases} 1, & i \in \mathbf{A} \ , \\ -r, & i \in \mathbf{B} \ . \end{cases}$$

Then we have

$$Dcut(\mathbf{A}, \mathbf{B}) = -\frac{\mathbf{y}^T \Gamma \mathbf{D} \Gamma \mathbf{y}}{r} \ .$$

In addition, it is straightforward to derive that

$$\begin{aligned} \boldsymbol{\beta}^T \Gamma \mathbf{y} &= 0 \ , \\ \mathbf{y}^T \Gamma^2 \mathbf{y} &= r \|\boldsymbol{\beta}\|^2 = r \ . \end{aligned}$$

Therefore, we can write $Dcut$ in terms of the generalized partition vector $\mathbf{y}$ as

$$Dcut(\mathbf{A}, \mathbf{B}) = -\frac{\mathbf{y}^T \Gamma \mathbf{D} \Gamma \mathbf{y}}{\mathbf{y}^T \Gamma^2 \mathbf{y}} \ .$$

Finding the maximum disparity cut can then be stated as the following discrete optimization problem.

DEFINITION 3.1. (Maximal Disparity Cut) *Given a weighted undirected graph $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$ with affinity matrix $\mathbf{D}$ and unit length vertex weights $\boldsymbol{\beta}$, the maximum disparity cut is defined by the generalized partition vector*

$$\mathbf{y}^* = \underset{\substack{\mathbf{y}, \boldsymbol{\beta}^T \Gamma \mathbf{y} = 0, \\ \mathbf{y} \in \{1, -r\}^n}}{\operatorname{argmax}} \ -\frac{\mathbf{y}^T \Gamma \mathbf{D} \Gamma \mathbf{y}}{\mathbf{y}^T \Gamma^2 \mathbf{y}} \ .$$

Even though the above discrete optimization problem is still NP-complete, it is possible to find an approximation by relaxing the condition that y is either 1 or $-r$. When we only require y to be continuous and set $\mathbf{z} = \Gamma\mathbf{y}$, the following optimization problem is obtained.

DEFINITION 3.2. (Spectral Relaxation for Maximal Disparity Cut) *Given a weighted undirected graph* $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$ *with affinity matrix* $\mathbf{D}$ *and unit length vertex weights* $\boldsymbol{\beta}$*, a continuous approximation of the optimal generalized partition vector is*

$$\mathbf{z}^* = \underset{\substack{\mathbf{z}, \boldsymbol{\beta}^T\mathbf{z}=0, \\ \|\mathbf{z}\|=1}}{\text{argmax}} \quad -\mathbf{z}^T\mathbf{D}\mathbf{z} \ . \tag{3.8}$$

The corresponding bipartition is $\mathbf{A} = \{i \in \mathbf{V} : z_i^* \geq 0\}$ *and* $\mathbf{B} = \{i \in \mathbf{V} : z_i^* < 0\}$.

Since Γ is a diagonal matrix with positive diagonal entries, $\mathbf{z}^*$ and $\Gamma^{-1}\mathbf{z}^*$ generate identical bipartitions because their corresponding elements have identical signs. This optimization problem is solved as below.

THEOREM 3.1. *The optimal solution* $\mathbf{z}^*$ *of the problem described in Definition 3.2 is*

$$\mathbf{z}^* = \mathbf{U}_{\boldsymbol{\beta}}\mathbf{v}, \tag{3.9}$$

where $\mathbf{U}_{\boldsymbol{\beta}} \in \mathbb{R}^{n\times(n-1)}$ *is a matrix whose columns form an orthonormal basis of the null space of* $\boldsymbol{\beta}^T$*, and* $\mathbf{v}$ *is a unit length eigenvector corresponding to the smallest eigenvalue of* $\mathbf{U}_{\boldsymbol{\beta}}^T\mathbf{D}\mathbf{U}_{\boldsymbol{\beta}}$.

The proof is similar to that of Theorem 2.1, therefore, is omitted.

3.3 Connections between MSC and Dcut. As shown in Theorem 2.1, the maximal separation bipartition relies on the weight vector $\boldsymbol{\alpha}$. Similarly, the spectral relaxed maximal disparity cut depends on the choice of the vertex weights $\boldsymbol{\beta}$ as shown in Theorem 3.1. We prove that under certain conditions they generate identical bipartitions.

LEMMA 3.1. *Let the affinity matrix* $\mathbf{D}$ *in Definition 3.2 be the distance matrix in Definition 2.1 and* $\mathbf{p}$ *be the Perron vector of* $\mathbf{D}$*. If the smallest eigenvalue of* $\mathbf{D}$ *is algebraically simple and we choose the weight vectors* $\boldsymbol{\alpha} = \mathbf{p}$ *and* $\boldsymbol{\beta} = \frac{\mathbf{p}}{\|\mathbf{p}\|}$*, then the maximal separation bipartition in Definition 2.1 is identical to that generated by the spectral relaxed maximal disparity cut in Definition 3.2.*

Proof. Let $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ be eigenvalues of $\mathbf{D}$. Since $\mathbf{D}$ is a nonnegative distance matrix, from the properties of (metric) distance matrices [18, 1] and nonnegative matrices (Ch.8.2,[15]), we have that

$$\lambda_1 > 0 > \lambda_2 \geq \ldots \geq \lambda_n \ ,$$

and $\mathbf{p}$ is an eigenvector corresponding to λ_1. Moreover, since $\sum_{i=1}^n \lambda_i = \text{Trace}(\mathbf{D}) = 0$, we have

$$\lambda_1 > |\lambda_i|, \ i = 2, \ldots, n.$$

Since λ_n is algebraically simple, $-\lambda_n$ is the largest eigenvalue of $-\mathbf{D}$ and λ_n^2 is the second largest eigenvalue of $\mathbf{D}^2$.

Let $\mathbf{u}_2, \ldots, \mathbf{u}_n$ be unit length eigenvectors associated with $\lambda_2, \ldots, \lambda_n$, respectively. Clearly, $\mathbf{u}_2, \ldots, \mathbf{u}_n$ form an orthonormal basis of the null space of $\boldsymbol{\beta}^T$. Therefore

$$\mathbf{z}^* = \mathbf{u}_n \ .$$

Note that $\mathbf{u}_2, \ldots, \mathbf{u}_n$ also form an orthonormal basis of the null space of $\boldsymbol{\alpha}^T\mathbf{D}$. Since

$$\lambda_1^2 > \lambda_n^2 > \lambda_{n-1}^2 \geq \ldots \geq \lambda_2^2$$

are eigenvalues of $\mathbf{D}^2$ with corresponding eigenvectors $\mathbf{p}, \mathbf{u}_n, \mathbf{u}_{n-1}, \ldots, \mathbf{u}_2$, respectively, we get

$$\mathbf{w}^* = \mathbf{u}_n \ .$$

The proof then follows from $\mathbf{D}\mathbf{w}^* = \lambda_1\mathbf{z}^*$. □

Therefore, the maximal separation bipartition in Definition 2.1 is equivalent to the spectral relaxed maximal disparity cut when the affinity matrix is defined by the distance matrix and the Perron vector of the distance matrix is used as the weight vectors. Given an arbitrary positive weight vector $\boldsymbol{\alpha}$ for maximal separation bipartition, can we find positive vertex weights $\boldsymbol{\beta}$ such that $\mathbf{z}^* = \frac{\mathbf{D}\mathbf{w}^*}{\|\mathbf{D}\mathbf{w}^*\|}$, i.e., two bipartitions are identical? A necessary condition is presented as below.

LEMMA 3.2. *Let the affinity matrix* $\mathbf{D}$ *in Definition 3.2 be the distance matrix in Definition 2.1 and* $\mathbf{w}^*$ *be calculated by (2.5). Let* $\mathbf{z}^*$ *be computed by (3.9) and* $\mathbf{z}^*$ *is not an eigenvector of* $\mathbf{D}$*. If* $\mathbf{z}^* = \frac{\mathbf{D}\mathbf{w}^*}{\|\mathbf{D}\mathbf{w}^*\|}$*, then*

$$\boldsymbol{\beta} = \frac{\left(\mathbf{I} - \frac{\mathbf{D}\mathbf{w}^*\mathbf{w}^{*T}\mathbf{D}}{\|\mathbf{D}\mathbf{w}^*\|^2}\right)\mathbf{D}^2\mathbf{w}^*}{\|\left(\mathbf{I} - \frac{\mathbf{D}\mathbf{w}^*\mathbf{w}^{*T}\mathbf{D}}{\|\mathbf{D}\mathbf{w}^*\|^2}\right)\mathbf{D}^2\mathbf{w}^*\|} \tag{3.10}$$

where $\mathbf{I}$ *is an identity matrix.*

Proof. For the constrained optimization problem (3.8), the Lagrange function is

$$\mathcal{L}(\mathbf{z}, \sigma_1, \sigma_2) = -\mathbf{z}^T\mathbf{D}\mathbf{z} - \sigma_1\left(\mathbf{z}^T\mathbf{z} - 1\right) - \sigma_2\boldsymbol{\beta}^T\mathbf{z}$$

where σ_1 and σ_2 are Lagrange multipliers. Setting the respective derivatives to zero yields

$$2\mathbf{D}\mathbf{z} + 2\sigma_1\mathbf{z} + \sigma_2\boldsymbol{\beta} = \mathbf{0}\,, \tag{3.11}$$

$$\mathbf{z}^T\mathbf{z} = 1\,, \tag{3.12}$$

$$\boldsymbol{\beta}^T\mathbf{z} = 0\,. \tag{3.13}$$

Left multiplying both sides of (3.11) with $\mathbf{z}^T$ and applying identities (3.12) and (3.13), we obtain

$$\sigma_1 = -\mathbf{z}^T\mathbf{D}\mathbf{z}\,. \tag{3.14}$$

Similarly, we get

$$\sigma_2 = -\frac{2\boldsymbol{\beta}^T\mathbf{D}\mathbf{z}}{\|\boldsymbol{\beta}\|^2} \tag{3.15}$$

by multiplying both side of (3.11) with $\boldsymbol{\beta}^T$ and applying identities (3.13) and (3.14). Substituting (3.14) and (3.15) into (3.12) gives

$$\left(\mathbf{I} - \mathbf{z}\mathbf{z}^T\right)\mathbf{D}\mathbf{z} = \frac{\boldsymbol{\beta}^T\mathbf{D}\mathbf{z}}{\|\boldsymbol{\beta}\|^2}\boldsymbol{\beta}\,.$$

Since $\mathbf{z}^*$ is not an eigenvector of $\mathbf{D}$, we have $\boldsymbol{\beta}^T\mathbf{D}\mathbf{z}^* \neq 0$. Therefore

$$\boldsymbol{\beta} = \frac{\|\boldsymbol{\beta}\|^2}{\boldsymbol{\beta}^T\mathbf{D}\mathbf{z}^*}\left(\mathbf{I} - \mathbf{z}^*\mathbf{z}^{*T}\right)\mathbf{D}\mathbf{z}\,. \tag{3.16}$$

By substituting $\mathbf{z}^* = \frac{\mathbf{D}\mathbf{w}^*}{\|\mathbf{D}\mathbf{w}^*\|}$ into (3.16) and enforcing the unit length constraint on $\boldsymbol{\beta}$, we get (3.10). □

4 Experiments.

Based on an artificial data set, the USPS data set, and a COREL image data set, we evaluate the performance of Algorithm 2.1. Comparisons with the normalized cut (Ncut) method in [20] are provided in some of our experimental results.

4.1 Artificial Data Set. The artificial data set consists of 200 samples belonging to two classes. Each class contains 100 samples. The samples in the first class are distributed as a two dimensional Gaussian with zero means and identity covariance matrix. The samples in the second class are generated by the following stochastic process:

$$\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}$$

where $[Y_1, Y_2]^T$ is distributed as a two dimensional Gaussian with means $[3, 3]^T$ and identity covariance matrix, and θ is a random variable with uniform distribution over the interval $[0, 2\pi]$. Figure 2 shows 200 randomly generated samples from the two classes.

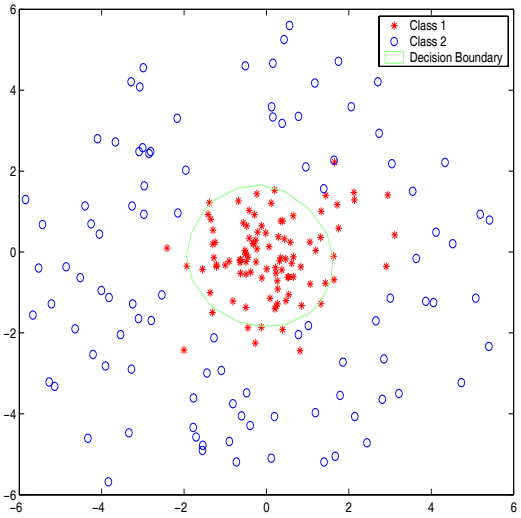

Figure 2: An artificial data set. Samples in Class 1 and Class 2 are denoted by stars and circles, respectively. The curve in the middle is the decision boundary of maximal separation clustering.

We compare the performance of Algorithm 2.1 with that of Ncut. The affinity matrix in Ncut is defined by Gaussian kernel $K(\mathbf{x}, \mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{\sigma^2}}$. Since Gaussian kernel computes the inner produce in a kernel-induced feature space $\mathcal{H}$, a metric distance in $\mathcal{H}$ can be defined according to (2.3) and is used in Algorithm 2.1 to calculate the distance matrix $\mathbf{D}$. Because we know the "true" class label for each sample, the classification error is one way to capture the goodness of clustering. Let $\mathbf{C}_1$ and $\mathbf{C}_2$ be a bipartition of the data set $\mathbf{C}$. The classification error is defined as

$$\mathrm{Err}(\mathbf{C}_1, \mathbf{C}_2) = \frac{1}{|\mathbf{C}|}\min\left(|\mathbf{C}_{1,1}| + |\mathbf{C}_{2,2}|, |\mathbf{C}_{1,2}| + |\mathbf{C}_{2,1}|\right)$$

where $|\mathbf{C}|$ is the size of the data set $\mathbf{C}$, $\mathbf{C}_{i,j}$ consists of samples in $\mathbf{C}_i$ that belong to class j $(i, j = 1, 2)$.

The kernel parameter, σ^2, is allowed to take values of 1, 3, 5, ..., 29. Under each value, experiments are repeated over 10 randomly generated data sets for each algorithm. The minimum of the average classification errors is listed in Table 1 along with the corresponding σ^2 values and 95% confidence intervals. Clearly, for this artificial data set, the MSC algorithm performs significantly better than Ncut. For the MSC algorithm, the Perron weight vector works better than the other two weighting schemes. But the difference is not statistically significant.

Since the MSC algorithm learns a decision boundary (the curve in the center of Figure 2), the clustering generalizes to unseen inputs. Each decision boundary is

Table 1: Comparing the MSC algorithm and Ncut on artificial data sets. The numbers listed are the average classification errors over 10 randomly generated data sets and the corresponding 95% confidence intervals. MSC1: MSC with $\boldsymbol{\alpha}_1$ weight vector; MSC2: MSC with $\boldsymbol{\alpha}_2$ weight vector; MSC3: MSC with $\boldsymbol{\alpha}_3$ weight vector. α_1, α_2, and α_3 are defined in Section 2.2

	σ^2	Average Classification Error	95% Confidence Interval
MSC1	7	3.9%	[2.81%, 4.99%]
MSC2	15	3.6%	[2.53%, 4.67%]
MSC3	7	3.1%	[2.47%, 3.73%]
Ncut	23	29.95%	[27.79%, 32.11%]

Table 2: The generalization performance of the MSC algorithm on artificial data sets. The numbers listed are the average generalization errors over 10 randomly generated testing sets and the corresponding 95% confidence intervals.

	σ^2	Average Generalization Error	95% Confidence Interval
MSC1	7	4.15%	[3.14%, 5.16%]
MSC2	15	4.85%	[3.72%, 5.98%]
MSC3	7	3.90%	[2.52%, 5.28%]

tested over a new set of 200 samples generated by the above distributions (each class contains 100 samples). The average generalization errors over 10 testing sets are reported in Table 2 along with the corresponding 95% confidence intervals.

4.2 USPS Data Set. The USPS data set contains 9298 grayscale images of handwritten digits. The images are size normalized to fit in a 16×16 pixel box while preserving their aspect ratio. The data set is divided into a training set of 7291 samples and a testing set of 2007 samples. For each sample, the input feature vector consists of 256 grayscale values. Since MSC deals with binary clustering, the training set is divided into 45 subsets, $\mathbf{S}_{i,j},\ i = 0, \ldots, 9,\ j = 1, \ldots, 9,\ i \neq j$. The subset $\mathbf{S}_{ij}$ consists of digits i and j. In the same way, the testing set is divided into 45 subsets.

We compare the performance of the MSC algorithm with that of Ncut using the training set. The affinity matrix in Ncut is defined by Gaussian kernel. The distance matrix in Algorithm 2.1 is computed using (2.3). The kernel parameter, σ^2, is allowed to take values of 50, 100, ..., 600. For each value of σ^2, the average classification error (defined in Section 4.1) is computed over the 45 subsets. The numbers reported in Table 3 are the minimum average classification errors along with the corresponding σ^2 values and standard deviations.

Table 3: Comparing the MSC algorithm and Ncut on the USPS data set. The numbers listed are the average classification errors over 45 subsets and the corresponding standard deviations.

	σ^2	Average Classification Error	Standard Deviation
MSC1	350	7.86%	8.05%
MSC2	300	7.68%	8.17%
MSC3	250	7.70%	8.12%
Ncut	100	7.05%	8.39%

Table 4: The generalization performance of the MSC algorithm on the USPS data set. The numbers listed are the average generalization errors over 45 subsets and the corresponding standard deviations.

	σ^2	Average Generalization Error	Standard Deviation
MSC1	350	9.26%	7.54%
MSC2	300	8.91%	7.78%
MSC3	250	9.06%	7.67%

As we can see Ncut performs slightly better than the proposed method for the USPS data set. However, the difference is not statistically significant. The separation functions learned from the training sets are also applied to the testing sets. The average generalization errors over 45 subsets are reported in Table 4 along with the corresponding standard deviations.

4.3 COREL Data Set. The image data set employed in our empirical study consists of 2000 images taken from 20 CD-ROMs published by COREL Corporation. Each COREL CD-ROM of 100 images represents one distinct topic of interest. Therefore, the data set has 20 thematically diverse image categories. All the images are in JPEG format with size 384×256 or 256×384. The image category names and some randomly selected sample images from each category are shown in Figure 3. Each image is represented as a collection of regions obtained from image segmentation. Nine features are extracted from each region. They capture the color, texture, and shape properties of the region. For a detailed discussion of the image segmentation algorithm and imagery features, please refer to [5]. The image data set and region features are available at `http://www.cs.uno.edu/~yixin/ddsvm.html`.

Let $\mathbf{B}_i = \{\mathbf{x}_{ij} \in \mathbb{R}^9 : j = 1, \ldots, N_i\}$ be the collection of region features for image i. The distance between two images, with respective collection of regions features $\mathbf{B}_k$ and $\mathbf{B}_l$, is defined by the Hausdorff distance [9]

$$H(\mathbf{B}_k, \mathbf{B}_l) = \max\left(h(\mathbf{B}_k, \mathbf{B}_l), h(\mathbf{B}_l, \mathbf{B}_k)\right)$$

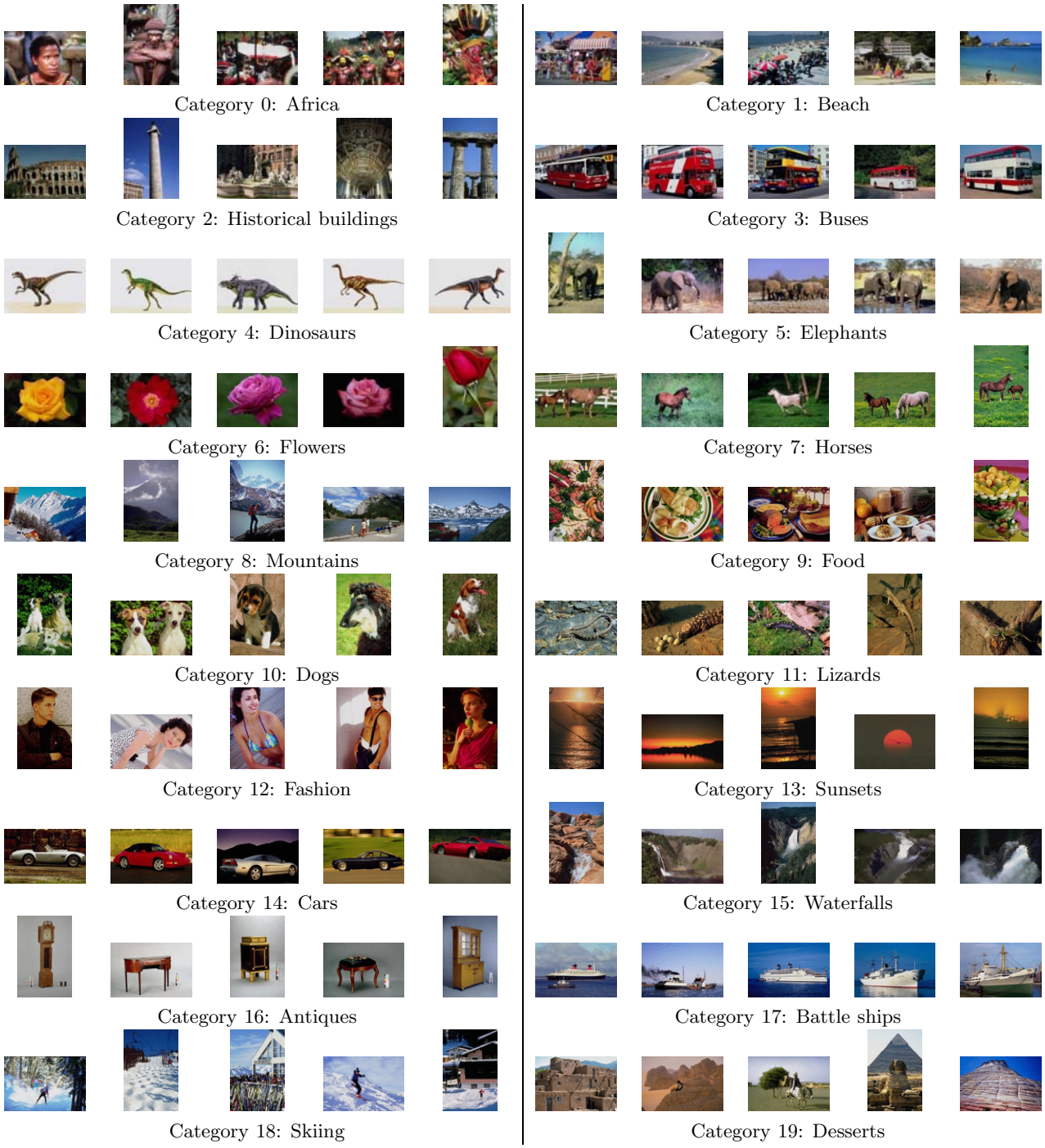

Figure 3: Sample images taken from 20 image categories.

where

$$h(\mathbf{B}_k, \mathbf{B}_l) = \max_{\mathbf{x}\in\mathbf{B}_k} \min_{\mathbf{y}\in\mathbf{B}_l} \|\mathbf{x}-\mathbf{y}\| \ .$$

Since the Hausdorff distance is a metric, it is applied to construct the distance matrix in Algorithm 2.1.

The MSC algorithm is recursively applied to the data set. Each time, the largest cluster is bipartitioned. Clearly, t iterations produce $t+1$ clusters. We use *purity* and *entropy* to measure the goodness of image clustering. Assume we are given a set of n images ($n = 2000$ in this experiment) belonging to c distinctive classes denoted by $0, \ldots, c-1$ ($c = 20$ in this experiment) and the

Table 5: Maximal separation clustering of images with $\boldsymbol{\alpha}_1$ weight vector.

Cluster	Size	Purity	Entropy	Dominant Class
1	65	0.2769	0.7397	Antiques
2	138	0.1594	0.8709	Horses
3	90	0.2667	0.7445	Cars
4	111	0.2162	0.8037	Lizards
5	120	0.1500	0.8160	Beach
6	83	0.2651	0.6563	Battle ships
7	111	0.2613	0.7869	Fashion
8	129	0.3333	0.7690	Food
9	106	0.3868	0.6793	Horses
10	83	0.5783	0.4564	Dinosaurs
11	81	0.1852	0.7625	Waterfalls
12	123	0.3821	0.6062	Buses
13	97	0.2371	0.7923	Lizards
14	101	0.5149	0.5440	Sunsets
15	124	0.2258	0.6615	Mountains
16	91	0.1868	0.8178	Africa
17	86	0.5930	0.4603	Flowers
18	93	0.2258	0.7799	Waterfalls
19	98	0.2653	0.7582	Elephants
20	70	0.3571	0.5619	Dinosaurs

Table 6: Maximal separation clustering of images with $\boldsymbol{\alpha}_2$ weight vector.

Cluster	Size	Purity	Entropy	Dominant Class
1	109	0.2844	0.7384	Elephants
2	111	0.4865	0.5437	Buses
3	89	0.2921	0.7782	Fashion
4	135	0.2148	0.7078	Mountains
5	106	0.3019	0.7481	Waterfalls
6	100	0.2200	0.8197	Lizards
7	108	0.4444	0.6142	Horses
8	70	0.3143	0.5938	Dinosaurs
9	84	0.5595	0.5252	Flowers
10	134	0.1493	0.8197	Beach
11	41	0.2195	0.7470	Lizards
12	151	0.2119	0.8411	Lizards
13	125	0.3840	0.7480	Food
14	81	0.2716	0.6567	Battle ships
15	104	0.1731	0.8434	Africa
16	91	0.1868	0.7654	Sunsets
17	100	0.5300	0.5323	Sunsets
18	78	0.2564	0.7555	Antiques
19	90	0.3111	0.7150	Cars
20	93	0.5484	0.5092	Dinosaurs

images are grouped into $t+1$ clusters $\mathbf{C}_j$, $j = 1, \ldots, m$. Cluster $\mathbf{C}_j$'s purity can be defined as

$$p(\mathbf{C}_j) = \frac{1}{|\mathbf{C}_j|} \max_{k=1,\ldots,c} |C_{j,k}|$$

where $\mathbf{C}_{j,k}$ consists of images in $\mathbf{C}_j$ that belong to class k. Each cluster may contain images of different classes. Purity gives the ratio of the dominant class size in the cluster to the cluster size itself. The value of purity is always in the interval $[\frac{1}{c}, 1]$ with a larger value means that the cluster is a "purer" subset of the dominant class. Entropy is another cluster quality measure, which is defined as follows:

$$h(\mathbf{C}_j) = -\frac{1}{\log c} \sum_{k=1}^{c} \frac{|\mathbf{C}_{j,k}|}{|\mathbf{C}_j|} \log \frac{|\mathbf{C}_{j,k}|}{|\mathbf{C}_j|} \ .$$

Since entropy considers the distribution of semantic classes in a cluster, it is a more comprehensive measure than purity. Note that we have normalized entropy so that the value is between 0 and 1. Contrary to the purity measure, an entropy value near 0 means the cluster is comprised mainly of 1 category, while an entropy value close to 1 implies that the cluster contains a uniform mixture of all categories. For example, if half of the images of a cluster belong to one class and the rest of the images are evenly divided into 19 different classes, then the entropy is 0.7228 and the purity is 0.5.

Table 5, Table 6, and Table 7 show the purity and entropy of clusters generated by Algorithm 2.1 (with weight vector $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$, and $\boldsymbol{\alpha}_3$, respectively) after 19 bipartitions (i.e., 20 clusters). Size of each cluster and the name of the dominant class in each cluster are also listed. Since the images belong to 20 classes, ideally, each of the 20 clusters should contains 100 images from a unique classes. In our experiments, the average purities under $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$, and $\boldsymbol{\alpha}_3$ are 0.3034, 0.3180, and 0.3155, respectively. And the average entropies are 0.7034, 0.6999, 0.6992. Although the results we obtained is far from perfect, they are significantly better than a random guess where the average purity would be 0.05 and the average entropy would be 1.0. It is observed that in each of the three experiments, there are 4 classes which do not appear as the dominant class in any of the clusters: *Historical buildings*, *Dogs*, *Skiing*, and *Desserts*. It is interesting to observe that in all three experiments, *Historical buildings* is the second largest class in the cluster where the dominant class is *Battle ships*; *Dogs* is the second largest class in the cluster where *Horses* is the dominant class; and *Skiing* is the second largest class in the cluster where *Beach* is the dominant class. *Desserts* does not even show up as the second largest class in any clusters. But it turns out to be the third largest class in the cluster where the top two largest classes are *Battle ships* and *Historical buildings*.

5 Conclusions and Future Work.

In this paper, we propose a new clustering algorithm which computes an "optimal" hyperplane maximizing the sum of squared distance in a feature space induced

Table 7: Maximal separation clustering of images with $\boldsymbol{\alpha}_3$ weight vector.

Cluster	Size	Purity	Entropy	Dominant Class
1	100	0.5300	0.5323	Sunsets
2	50	0.2200	0.7720	Lizards
3	101	0.1980	0.8214	Africa
4	81	0.2840	0.6540	Battle ships
5	100	0.2900	0.7474	Waterfalls
6	151	0.2053	0.8331	Lizards
7	113	0.2566	0.7616	Elephants
8	93	0.5376	0.5128	Dinosaurs
9	86	0.5698	0.5057	Flowers
10	129	0.1550	0.8076	Beach
11	113	0.4779	0.5472	Buses
12	101	0.2178	0.8167	Lizards
13	125	0.3760	0.7439	Food
14	135	0.2148	0.7029	Mountains
15	108	0.4259	0.6345	Horses
16	93	0.2903	0.7757	Fashion
17	89	0.1798	0.7641	Sunsets
18	74	0.2568	0.7462	Antiques
19	88	0.2955	0.7222	Cars
20	70	0.3286	0.5833	Dinosaurs

by the training data and the given metric distance. The separating hyperplane transforms to a nonlinear decision boundary in the input space. Hence the clustering generalizes to unseen samples. The connection between the proposed clustering algorithm and spectral graph partition methods is discussed. Specifically, we prove that, under proper weight vectors, the proposed clustering algorithm is equivalent to a spectral relaxed graph cut – disparity cut. The disparity cut criterion takes into account the between-cluster dissimilarity, the within-cluster dissimilarity, and the size of the clusters. We provide extensive experimental results to verify the method.

Acknowledgements.

This work was supported in part by the University of New Orleans, The Research Institute for Children, and NASA/EPSCoR DART Grant NCC5-573. The authors would like to thank Bin Fu for valuable discussions.

References

[1] J. W. Auer, An Elementary Proof of the Invertibility of Distance Matrices, *Linear and Multilinear Algebra*, 40:119–124, 1995.

[2] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh, Clustering with Bregman Divergences, *Proc. 4th SIAM Int'l Conf. on Data Mining*, pages 234–245, 2004.

[3] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet, Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering, *Advances in Neural Information Processing Systems 16*, 2003.

[4] P. S. Bradley, U. M. Fayyad, and C. Reina, Scaling Clustering Algorithms to Large Databases, *Proc. 4th Int'l Conf. on Knowledge Discovery and Data Mining*, pages 9–15, 1998.

[5] Y. Chen and J. Z. Wang, Image Categorization by Learning and Reasoning with Regions, *Journal of Machine Learning Research*, 5:913–939, 2004.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum Likelihood from Incomplete Data via the EM Algorithm, *Journal of the Royal Statistical Society*, Series B, 39:1–38, 1977.

[7] I. S. Dhillon, Y. Guan, and B. Kulis Kernel k-means, Spectral Clustering and Normalized Cuts, *Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, 2004.

[8] C. Ding, X. He, H. Zha, M. Gu and H. Simon, Spectral Min-Max Cut for Graph Partitioning and Data Clustering, *Proc. 1st IEEE Int'l Conf. on Data Mining*, pages 107–114, 2001.

[9] G. B. Folland *Real Analysis: Modern Techniques and Their Applications*, 2nd edition, John Wiley & Sons, Inc., 1999.

[10] G. H. Golub and C. F. Van Loan, *Matrix Analysis*, 3rd ed., Johns Hopkins University Press, 1996.

[11] L. Hagen and A. B. Kahng, New Spectral Methods for Ratio Cut Partitioning and Clustering, *IEEE Transactions on Computer-Aided Design*, 11(9):1074–1085, 1992.

[12] J. A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, 1975.

[13] J. A. Hartigan and M. A. Wong, *Algorithm AS136: A k-means Clustering Algorithm*, Applied Statistics, 28:100-108, 1979.

[14] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer-Verlag, 2001.

[15] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1990.

[16] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, 1990.

[17] A. Y. Ng, M. I. Jordan, and Y. Weiss, On Spectral Clustering: Analysis and an Algorithm, *Advances in Neural Information Processing Systems 14*, 2001.

[18] I. J. Schoenberg, On Certain Metric Spaces Arising from Euclidean Spaces by a Change of Metric and Their Imbedding in Hilbert Space, *The Annals of Mathematics*, 38(4):787–793, 1937.

[19] N. Shental, A. Zomet, T. Hertz, and Y. Weiss, Pairwise Clustering and Graphical Models, *Advances in Neural Information Processing Systems 16*, 2003.

[20] J. Shi and J. Malik, Normalized Cuts and Image Segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[21] V. Vapnik, *Statistical Learning Theory*, Wiley-Interscience, 1998.