Semi-supervised Graph Clustering: A Kernel Approach

Brian Kulis KULIS@CS.UTEXAS.EDU
Sugato Basu SUGATO@CS.UTEXAS.EDU
Inderjit Dhillon INDERJIT@CS.UTEXAS.EDU
Raymond Mooney MOONEY@CS.UTEXAS.EDU
Department of Computer Sciences, University of Texas at Austin, Austin, TX, 78712

Abstract

Semi-supervised clustering algorithms aim to improve clustering results using limited supervision. The supervision is generally given as pairwise constraints; such constraints are natural for graphs, yet most semi-supervised clustering algorithms are designed for data represented as vectors. In this paper, we unify vector-based and graph-based approaches. We show that a recently-proposed objective function for semi-supervised clustering based on Hidden Markov Random Fields, with squared Euclidean distance and a certain class of constraint penalty functions, can be expressed as a special case of the weighted kernel k-means objective. A recent theoretical connection between kernel k-means and several graph clustering objectives enables us to perform semi-supervised clustering of data given either as vectors or as a graph. For vector data, the kernel approach also enables us to find clusters with non-linear boundaries in the input data space. Furthermore, we show that recent work on spectral learning (Kamvar et al., 2003) may be viewed as a special case of our formulation. We empirically show that our algorithm is able to outperform current state-of-the-art semi-supervised algorithms on both vector-based and graph-based data sets.

1. Introduction

Semi-supervised clustering algorithms have recently received a significant amount of attention in the machine learning and data mining communities. In traditional clustering algorithms, only unlabeled data is used to generate clusterings; in semi-supervised clustering, the goal is to incorporate prior information about clusters into the algorithm in order to improve the clustering results. A number of recent papers have explored this problem (Wagstaff et al., 2001; Klein et al., 2002; Xing et al., 2003; Kamvar et al., 2003; Bar-Hillel et al., 2003; Basu et al., 2004).

As is common for most semi-supervised clustering algorithms, we assume that we have pairwise must-link constraints (pairs of points that should belong in the same cluster) and cannot-link constraints (pairs of points that should belong in different clusters) provided with the input. Pairwise constraints occur naturally in many domains, e.g., the Database of Interacting Proteins (DIP) data set in biology contains information about proteins co-occurring in processes, which can be viewed as must-link constraints during gene clustering. Constraints of this form are also natural in the context of the graph clustering problem (a.k.a. graph partitioning or vertex partitioning), where edges in the graph encode pairwise relationships.

Recently, a probabilistic framework for semi-supervised clustering with pairwise constraints was proposed based on Hidden Markov Random Fields (Basu et al., 2004). This framework proposed a general semi-supervised clustering objective based on maximizing the joint likelihood of data and constraints in the HMRF model, as well as a k-means like iterative algorithm for optimizing the objective. However, HMRF-KMEANS can cluster input data only in the form of vectors.

Our main contributions in this paper are as follows:

1. We show that the HMRF semi-supervised clustering objective with squared Euclidean distance and cluster-size weighted penalties is a special case of the weighted kernel k-means objective function (Dhillon et al., 2004a). Given input data in the form of vectors and pairwise constraints, we show how to construct a kernel such that running kernel k-means results in a monotonic decrease of the semi-supervised cluster-

Appearing in *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005.

ing objective function at every iteration of the kernel k-means algorithm.

2. The weighted kernel k-means algorithm can be used to monotonically optimize a wide class of graph clustering objectives such as minimizing the normalized cut (Dhillon et al., 2004b). Hence, our approach can easily be generalized to optimize a number of different semi-supervised graph clustering objectives for which constraint-based supervision is more natural. This equivalence gives us a new semi-supervised clustering algorithm SS-KERNEL-KMEANS that can cluster data given either as vectors or a graph, along with supervision in the form of constraints.

3. For vector-based data we have the ability to apply a kernel function, which allows SS-KERNEL-KMEANS to discover clusters with non-linear boundaries in the input space.

4. We show how a previously proposed algorithm SPECTRAL-LEARNING (Kamvar et al., 2003) can be viewed as a special case of our framework. The SPECTRAL-LEARNING algorithm can be viewed as optimizing an underlying semi-supervised clustering objective; specifically, it optimizes a relaxed version of semi-supervised ratio cut.

5. We empirically demonstrate that SS-KERNEL-KMEANS outperforms a current state-of-the-art algorithm HMRF-KMEANS (Basu et al., 2004); we run experiments on a synthetic data set and a real-life handwritten character recognition set to show that using a kernel function to map vectors to a higher-dimensional space significantly improves results. We also compare SS-KERNEL-KMEANS to SPECTRAL-LEARNING on a gene network data set and demonstrate how we improve graph clustering results with the addition of constraints. This network is a graph-based data set that HMRF-KMEANS cannot cluster.

2. Background and Related Work

In this section, we will give the necessary background for our proposed kernel-based semi-supervised clustering algorithm SS-KERNEL-KMEANS, and also describe related research.

2.1. Graph Clustering and Kernel k-means

In graph clustering (Chan et al., 1994; Shi & Malik, 2000), the input is assumed to be a graph $G = (\mathcal{V}, \mathcal{E}, A)$, where $\mathcal{V}$ is the set of vertices, $\mathcal{E}$ is the set of edges, and A is the edge affinity matrix. A_{ij} represents the edge-weight between vertex i and j.

Let $\text{links}(\mathcal{A}, \mathcal{B}) = \sum_{i \in \mathcal{A}, j \in \mathcal{B}} A_{ij}$. Furthermore, let $\text{degree}(\mathcal{A}) = \text{links}(\mathcal{A}, \mathcal{V})$. We seek to find a k-way disjoint partitioning[1] $\{\mathcal{V}_c\}_{c=1}^k$ of $\mathcal{V}$ to minimize a particular objective. Table 1 gives a list of some common graph clustering objectives.

Name	Objective Function
Ratio Association	$\underset{\mathcal{V}_1,\ldots,\mathcal{V}_k}{\text{maximize}} \sum_{c=1}^k \frac{\text{links}(\mathcal{V}_c, \mathcal{V}_c)}{\lvert\mathcal{V}_c\rvert}$
Ratio Cut	$\underset{\mathcal{V}_1,\ldots,\mathcal{V}_k}{\text{minimize}} \sum_{c=1}^k \frac{\text{links}(\mathcal{V}_c, \mathcal{V}\setminus\mathcal{V}_c)}{\lvert\mathcal{V}_c\rvert}$
Normalized Cut	$\underset{\mathcal{V}_1,\ldots,\mathcal{V}_k}{\text{minimize}} \sum_{c=1}^k \frac{\text{links}(\mathcal{V}_c, \mathcal{V}\setminus\mathcal{V}_c)}{\text{degree}(\mathcal{V}_c)}$

Table 1. Examples of graph clustering objectives

To make the connection between graph clustering and vector-based clustering, we introduce the weighted kernel k-means objective function (Dhillon et al., 2004a). Given a set of data vectors $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^n$ with $\mathbf{x}_i \in \mathbb{R}^n$, the goal of weighted kernel k-means is to find a k-way disjoint partitioning $\{\pi_c\}_{c=1}^k$ of the data (where π_c represents the c^{th} cluster) such that the following objective is minimized:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \alpha_i \|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2$$

$$\text{where } \mathbf{m}_c = \frac{\sum_{\mathbf{x}_i \in \pi_c} \alpha_i \phi(\mathbf{x}_i)}{\sum_{\mathbf{x}_i \in \pi_c} \alpha_i}$$

Each vector $\mathbf{x}_i$ has a pre-specified non-negative weight α_i associated with it, and ϕ is a function mapping vectors in $\mathcal{X}$ to a (generally) higher-dimensional space. If all weights are set to one and ϕ is the identity function, this reduces to standard k-means.

If we expand the distance computation $\|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2$ in the weighted kernel k-means objective function, we obtain the following:

$$\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_i) - \frac{2\sum_{\mathbf{x}_j \in \pi_c} \alpha_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)}{\sum_{\mathbf{x}_j \in \pi_c} \alpha_j} + \frac{\sum_{\mathbf{x}_j, \mathbf{x}_l \in \pi_c} \alpha_j \alpha_l \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_l)}{(\sum_{\mathbf{x}_j \in \pi_c} \alpha_j)^2}.$$

Notice that all computation involving data points is in the form of inner products. As a result, we can use the *kernel trick*: if we can compute the dot product $K_{ij} = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ between $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ efficiently, we are able to compute distances between points in this mapped space without explicitly knowing the mapping of $\mathbf{x}_i$ and $\mathbf{x}_j$ to $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ respectively. It can easily be shown that *any* positive semidefinite matrix K can be thought of as a kernel matrix (Cristianini & Shawe-Taylor, 2000).

[1] k disjoint data subsets, whose union is the whole data

Using the kernel matrix K, the distance computation is rewritten as:

$$K_{ii} - \frac{2\sum_{\mathbf{x}_j \in \pi_c} \alpha_j K_{ij}}{\sum_{\mathbf{x}_j \in \pi_c} \alpha_j} + \frac{\sum_{\mathbf{x}_j, \mathbf{x}_l \in \pi_c} \alpha_j \alpha_l K_{jl}}{(\sum_{\mathbf{x}_j \in \pi_c} \alpha_j)^2}. \quad (1)$$

ALGORITHM 1: Basic Weighted Kernel k-means.

KERNEL-KMEANS(K, k, t_{max}, $\boldsymbol{\alpha}$, $\{\pi_c^{(0)}\}_{c=1}^k$)
Input: K: kernel matrix, k: number of clusters, t_{max}: optional maximum number of iterations, $\boldsymbol{\alpha}$: weight vector, $\{\pi_c^{(0)}\}_{c=1}^k$: optional initial clusters
Output: $\{\pi_c\}_{c=1}^k$: final partitioning of the points
1. Initialize the k clusters from $\{\pi_c^{(0)}\}_{c=1}^k$, if provided as input, else randomly.
2. Set $t = 0$.
3. For each point $\mathbf{x}_i$ and every cluster c, compute $d(\mathbf{x}_i, \mathbf{m}_c)$ as in Eqn. (1).
4. Find $c^*(\mathbf{x}_i) = \text{argmin}_c d(\mathbf{x}_i, \mathbf{m}_c)$, resolving ties arbitrarily.
5. Compute the updated clusters as

$$\pi_c^{(t+1)} = \{\mathbf{x}_i \ : \ c^*(\mathbf{x}_i) = c\}.$$

6. If not converged or $t_{max} > t$, set $t = t+1$ and go to Step 3; Otherwise, stop and output final clusters $\{\pi_c^{(t+1)}\}_{c=1}^k$.

As a result, we may derive an algorithm analogous to k-means to monotonically decrease the weighted kernel k-means objective function without knowledge of the map ϕ. As shown in Algorithm 1, this algorithm is identical to standard k-means except for the fact that distances are computed using the kernel matrix.

When we consider such a weighted form of the kernel k-means objective function, it has been shown that many graph clustering objectives can be viewed as special cases of this objective (Dhillon et al., 2004b). As a result, the weighted kernel k-means algorithm can be used to optimize a number of graph clustering objectives. This allows us the flexibility of having as input to the algorithm either a graph or data vectors that have been mapped to a kernel space.

Table 2 shows how to set the weights and kernel for three popular graph clustering objectives to be *equivalent* to the weighted kernel k-means objective function. The node weights in this table are weights on the nodes in the graph that correspond to the weights of the data vectors in k-means. Here, D is a diagonal matrix whose entries correspond to the sum of the rows of the affinity matrix A, L is the Laplacian matrix $(D - A)$, and σ is a real number chosen to be sufficiently large such that K is positive definite. See Dhillon et al. (2004b) for further details about this mathematical equivalence.

2.2. Semi-supervised Clustering

Often when clustering, we have some background knowledge about the cluster structure. As mentioned previously, in this work we assume that this knowledge comes in the form of pairwise must-link or cannot-link constraints. Such constraints are natural for graphs, as pairwise relationships are explicitly captured via edges in a graph. However, most semi-supervised clustering with pairwise constraints assumes that the input is in the form of data vectors (Wagstaff et al., 2001; Klein et al., 2002; Xing et al., 2003; Bar-Hillel et al., 2003; Basu et al., 2004). Recent work by Bansal et al. (2002) in semi-supervised clustering considers inputs only in the form of constraints.

2.2.1. HMRF MODEL

Let us briefly describe a recently proposed objective for semi-supervised clustering of data vectors, which we will use in our formulation. Basu et al. (2004) proposed a framework for semi-supervised clustering based on Hidden Markov Random Fields (HMRFs). Choosing squared Euclidean distance as the cluster distortion measure and the generalized Potts potential as the constraint violation potential, the semi-supervised clustering objective can be expressed as:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\mathbf{x}_i - \mathbf{m}_c\|^2 + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ \text{s.t. } l_i \neq l_j}} w_{ij} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ \text{s.t. } l_i = l_j}} w_{ij} \quad (2)$$

where $\mathcal{M}$ is the set of must-link constraints, $\mathcal{C}$ is the set of cannot-link constraints, w_{ij} is the penalty cost for violating a constraint between $\mathbf{x}_i$ and $\mathbf{x}_j$, and l_i refers to the cluster label of $\mathbf{x}_i$. The first term in this objective function is the standard k-means objective function, the second term is a penalty function for violating must-link constraints, while the third term is a penalty function for violating cannot-link constraints. The algorithms developed for minimizing this objective (Basu et al., 2004) use an iterative relocation approach like k-means.

2.2.2. SPECTRAL LEARNING

Recently, spectral methods have become increasingly popular for clustering. These algorithms cluster data given in the form of a graph. One spectral approach to semi-supervised clustering is the SPECTRAL-LEARNING algorithm of Kamvar et al. (2003):

Objective	Node Weights	Kernel
Ratio Association	1 for all nodes	$K = \sigma I + A$
Ratio Cut	1 for all nodes	$K = \sigma I - L$
Normalized Cut	Degree of the node	$K = \sigma D^{-1} + D^{-1}AD^{-1}$

Table 2. Popular graph clustering objectives and corresponding weights and kernels given affinity matrix A

1. Form the affinity matrix A: the entries of A are assumed to be normalized between 0 and 1.

2. For all points i, j that have a must-link constraint, set $A_{ij} = 1$; for all points i, j that have a cannot-link constraint, set $A_{ij} = 0$.

3. Re-normalize the matrix using additive normalization (Kamvar et al., 2003): $N = \frac{1}{d_{max}}(A + d_{max}I - D)$.

4. Take the top k eigenvectors of A to be the columns of the matrix V, and cluster the rows of V.

In the above algorithm, d_{max} is the maximum row-sum of A and D is the degree matrix. Notice that the additive normalization is equal to $I - \frac{1}{d_{max}}L$, which is equivalent to $d_{max}I - L$ up to a scalar factor, where $L = D - A$ is the Laplacian of A. In the presentation of Kamvar et al. (2003), the SPECTRAL-LEARNING algorithm does not have an explicit underlying objective function. However in Section 3.4, we will show that this algorithm can be viewed as a special case of our unified semi-supervised clustering framework. Note that the Spectral Learning algorithm needs $O(kn)$ memory to store the k eigenvectors for n points, which can be a substantial overhead for large graphs.

Another recent paper (Yu & Shi, 2004) considered a semi-supervised formulation of the normalized cut objective and had a spectral algorithm associated with it. The main differences between this work and our algorithm are: (1) only must-link constraints are considered in their formulation, and there is no way to specify penalty weights for constraint violations: SS-KERNEL-KMEANS considers both must-link and cannot-link constraints and allows constraint violation with an associated penalty, thereby making it mode robust to constraint noise; (2) their formulation solves an expensive constrained eigen-decomposition problem while SS-KERNEL-KMEANS uses an efficient iterative algorithm, making it easily scalable to large data sets.

3. Kernel-based Semi-supervised Clustering

This section describes our proposed semi-supervised clustering algorithm SS-KERNEL-KMEANS that unifies vector-based and graph-based approaches using a kernel approach. Let us first show how to connect the kernel k-means objective with the semi-supervised clustering objective.

3.1. Constructing the Kernel

Recall the objective for semi-supervised clustering on a HMRF from Eqn. (2), using squared Euclidean distance as the clustering distortion measure and the generalized Potts potential for constraint penalty. Let us alter this penalty function: instead of adding a penalty term for a must-link violation if the two points are in different clusters, we give a *reward* for constraint satisfaction if the points are in the same cluster, by subtracting the corresponding penalty term from the objective. If the sum of weights for all must-link constraints is a constant, then this is equivalent to the original objective function up to an additive constant. Therefore minimizing $\mathcal{J}(\{\pi_c\}_{c=1}^k)$ is equivalent to minimizing:

$$\sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\mathbf{x}_i - \mathbf{m}_c\|^2 - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} w_{ij} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} w_{ij}.$$

We also introduce the notion of cluster-size weighted penalties, dividing each w_{ij} by the size of the cluster that the points are in. This gives us:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\mathbf{x}_i - \mathbf{m}_c\|^2 - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{w_{ij}}{|\pi_i|} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{w_{ij}}{|\pi_i|}. \quad (3)$$

Using the following result from Duda and Hart (1973):

$$\sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} 2\|\mathbf{x}_i - \mathbf{m}_c\|^2 = \sum_{c=1}^{k} \sum_{\mathbf{x}_i, \mathbf{x}_j \in \pi_c} \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{|\pi_c|},$$

and re-writing the sums, minimizing the objective in Eqn. (3) becomes equivalent to minimizing:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i, \mathbf{x}_j \in \pi_c} \frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{|\pi_c|} - \sum_{c=1}^{k} \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \pi_c \\ (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}}} \frac{2w_{ij}}{|\pi_c|} + \sum_{c=1}^{k} \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \pi_c \\ (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}}} \frac{2w_{ij}}{|\pi_c|}.$$

Let E be the matrix of pairwise squared Euclidean distances among the data points, such that $E_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|^2$ and W be the constraint matrix such that W_{ij} is $-w_{ij}$ for a cannot link, w_{ij} for a must link, and 0 otherwise. Furthermore, we introduce an indicator vector $\mathbf{z}_c$ for cluster c. This vector is of length n and $\mathbf{z}_c(i) = 0$ if $\mathbf{x}_i$ is not in cluster c, and 1 otherwise. We can now write the above objective in the following way:

$$\mathcal{J}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \frac{\mathbf{z}_c^T(E-2W)\mathbf{z}_c}{\mathbf{z}_c^T\mathbf{z}_c}.$$

where $\mathbf{z}_c^T\mathbf{z}_c$ is the size of cluster π_c, and $\mathbf{z}_c^T(E-2W)\mathbf{z}_c$ gives the sum of $E_{ij} - 2W_{ij}$ over all $\mathbf{x}_i$ and $\mathbf{x}_j$ in π_c.

Now define a matrix $\tilde{Z}$ such that $\tilde{Z}_{\cdot c}$, the column c of $\tilde{Z}$, is equal to $\mathbf{z}_c/(\mathbf{z}_c^T\mathbf{z}_c)^{1/2}$. $\tilde{Z}$ is an orthonormal matrix ($\tilde{Z}^T\tilde{Z} = I_k$), and the objective that we want to minimize can be re-written as:

$$\sum_{c=1}^{k} \tilde{Z}_{\cdot c}^T(E-2W)\tilde{Z}_{\cdot c} = \text{trace}(\tilde{Z}^T(E-2W)\tilde{Z}).$$

It has been shown that the kernel k-means objective function can also be expressed as a trace optimization problem (Dhillon et al., 2004a). In particular, the kernel k-means objective is written as a maximization of trace(Y^TKY), with Y analogous to $\tilde{Z}$ (an orthonormal indicator matrix). The only difference between these objectives is that our semi-supervised objective is expressed as a trace minimization, while kernel k-means is expressed as a trace maximization. To make our problem into a maximization problem, we note that for squared Euclidean distance, $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{x}_i \cdot \mathbf{x}_i + \mathbf{x}_j \cdot \mathbf{x}_j - 2\mathbf{x}_i \cdot \mathbf{x}_j$. Let S be the similarity matrix ($S_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$) and let $\tilde{S}$ be the matrix such that $\tilde{S}_{ij} = S_{ii} + S_{jj}$. Then $E = \tilde{S} - 2S$.

We replace E in the trace minimization, leading to a minimization of trace($\tilde{Z}^T(\tilde{S} - 2S - 2W)\tilde{Z}$). Since trace($\tilde{Z}^T\tilde{S}\tilde{Z}$) is a constant, this leads to a maximization of trace($\tilde{Z}^T(S+W)\tilde{Z}$). We define a matrix $K = S + W$; with this matrix, we get that our problem is expressed as a maximization of trace($\tilde{Z}^TK\tilde{Z}$), and is mathematically equivalent to unweighted kernel k-means.

Note that this matrix K may not be positive semidefinite, a requirement for kernel k-means to converge. One can avoid this problem by performing diagonal shifting (Dhillon et al., 2004b) for kernelizing K, as shown in Step 2 of Algorithm 2. In our experience, one can run kernel k-means directly on K – even though there is no theoretical guarantee in this case, the algorithm generally monotonically converges in practice.

ALGORITHM 2: Semi-Supervised Kernel k-means.

SS-KERNEL-KMEANS(S, k, $\mathcal{M}$, $\mathcal{C}$, W, t_{max})
Input: S: input similarity matrix, k: number of clusters, $\mathcal{M}$: set of must-link constraints, $\mathcal{C}$: set of cannot-link constraints, W: constraint penalty matrix, t_{max}: optional maximum number of iterations
Output: $\{\pi_c\}_{c=1}^k$: final partitioning of the points
1. Form the matrix $K = S + W$.
2. Diagonal-shift K by adding σI to guarantee positive definiteness of K.
3. Get initial clusters $\{\pi_c^{(0)}\}_{c=1}^k$ using constraints.
4. Return $\{\pi_c\}_{c=1}^k$ = KERNEL-KMEANS(K, k, t_{max}, $\mathbf{1}$, $\{\pi_c^{(0)}\}_{c=1}^k$), where $\mathbf{1}$ is the vector of all ones.

3.2. The Algorithm

Summarizing the result from the previous section, we have shown an equivalence between unweighted kernel k-means and HMRF-KMEANS with squared Euclidean distance and cluster-size weighted penalties. We can now develop an algorithm for semi-supervised clustering by constructing the appropriate kernel matrix K (see Algorithm 2). This algorithm simply constructs the kernel as derived in the previous section, performs proper cluster initialization and runs kernel k-means.

A key advantage to this approach is that the algorithm assumes a similarity matrix as input. This similarity matrix may come about as the result of applying a kernel function on vector data, or if the input is a graph affinity matrix (diagonal-shifting this matrix to enforce positive definiteness does not change the globally optimal solution).

For cluster initialization, we follow the approach of Basu et al. (2004): we take the transitive closure of the constraints to form neighborhoods, and then perform a size-weighted farthest-first traversal on these neighborhoods to get the k initial clusters.

3.3. Generalized Semi-supervised Graph Clustering

As mentioned previously, there is a direct mathematical connection between the weighted kernel k-means objective function and a wide class of graph clustering objectives. So far we have considered the unweighted case while deriving the connection between HMRF-KMEANS and kernel k-means. In this section, we generalize this result – due to space restrictions, we skip the detailed analysis and only state the results.

Consider a semi-supervised version of the normalized

graph cut problem, which seeks to minimize:

$$\sum_{c=1}^{k} \frac{\text{links}(\mathcal{V}_c, \mathcal{V} \setminus \mathcal{V}_c)}{\text{degree}(\mathcal{V}_c)} - \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{M} \\ l_i = l_j}} \frac{w_{ij}}{\deg(\mathcal{V}_{l_i})} + \sum_{\substack{\mathbf{x}_i, \mathbf{x}_j \in \mathcal{C} \\ l_i = l_j}} \frac{w_{ij}}{\deg(\mathcal{V}_{l_i})}. \quad (4)$$

Notice that, instead of cluster-size weighted penalties, we use degree-weighted penalties. Combining the analysis of (Dhillon et al., 2004b) with the analysis earlier in this paper, we can arrive at a trace maximization for this objective function. In the resulting algorithm, given affinity matrix A and constraint matrix W as before, we set $A' = A+W$ as in the unweighted case, and run weighted kernel k-means on $\sigma D^{-1} + D^{-1}A'D^{-1}$ with node weights equal to the degree of the node. Running weighted kernel k-means with this kernel and weights will then monotonically decrease the objective in Eqn. (4) for a graph with affinity matrix A.

Similar results may be obtained for semi-supervised ratio cut and semi-supervised ratio association. For example, for semi-supervised ratio cut, we would set $A' = A+W$ as in the unweighted case (adding weights based on the constraints), and then we would run weighted kernel k-means on $\sigma I - L$ (L is the Laplacian of A' here) with all node weights equal to one. In general, one simply uses the same kernels as in Table 2 with the prior addition of the constraint matrix to the affinity matrix as is done in the unweighted case; constraint penalties are the sum of the node weights of all nodes in a cluster.

3.4. Spectral Learning and Semi-supervised Ratio Cut

We can view SPECTRAL-LEARNING (Kamvar et al., 2003), described in Section 2.2.2, in our framework as follows: for a must-link constraint, if A_{ij} is the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$, we set $W_{ij} = 1 - A_{ij}$ (and hence the corresponding value in $A + W$ is 1). Similarly, for cannot-links, we set $W_{ij} = -A_{ij}$ (and hence the corresponding value in $A + W$ is 0). With this particular choice of constraint weights, the matrix $A+W$ is identical to the matrix from SPECTRAL-LEARNING before additive normalization. This leaves just the additive normalization step, which is the same normalization required for semi-supervised ratio cut (up to a diagonal shift, which does not change the globally optimal solution).

A well-known relaxation to our trace maximization problem, trace($Y^T KY$), is achieved by taking the top k eigenvectors of K. The matrix formed by these eigenvectors is the optimal Y matrix, under the relaxation that Y is an arbitrary orthonormal matrix. Such a relaxation leads to the SPECTRAL-LEARNING algorithm that was detailed in Section 2.2.2.

To summarize, the SPECTRAL-LEARNING algorithm of Kamvar et al. (2003) may be viewed as solving a relaxed semi-supervised ratio cut objective. Moreover, our earlier analysis shows that the fast iterative kernel k-means algorithm can be used to optimize this objective, thus removing any requirement for the expensive operation (for large data sets) of eigen-decomposition.

4. Experimental Results

In this section, we present experimental results comparing SS-KERNEL-KMEANS to HMRF-KMEANS, which optimizes an HMRF-based objective function using squared Euclidean distance as the distortion measure. We present results on a synthetic data set and a real-world handwritten character recognition data set, demonstrating that SS-KERNEL-KMEANS has better performance on these data sets with a proper choice of the kernel. We also present the results of SS-KERNEL-KMEANS on a gene network, which cannot be clustered with HMRF-KMEANS since the input is in the form of a graph.

4.1. Data sets

We performed experiments on the following 3 datasets:

1. `TwoCircles`: A synthetic data set comprising of 200 data points in 2 dimensions — 100 points in an inner circle are labeled to be in one class, and 100 data points in a surrounding outer circle are labeled to be in another class, as shown in Figure 1.

2. `Digits`: A subset of 10% of the data points chosen randomly from three classes $\{3, 8, 9\}$ of the *Pendigits* handwritten character recognition data set from the UCI Machine Learning Repository[2]. It has 317 points in a 16 dimensional-space.

3. `GeneNetwork`: An interaction network between 216 yeast genes, where each gene is labeled with one of 3 KEGG (Ogata et al., 1999) functional pathway labels. This data is a subgraph of a high-quality probabilistic functional network of yeast genes (Lee et al., 2004): each edge weight in this network represents a probability of linkage between two genes, estimated by integrating diverse functional genomic data sources. The genes do not have an explicit vector representation in this data set.

[2] http://www.ics.uci.edu/ mlearn/MLRepository.html

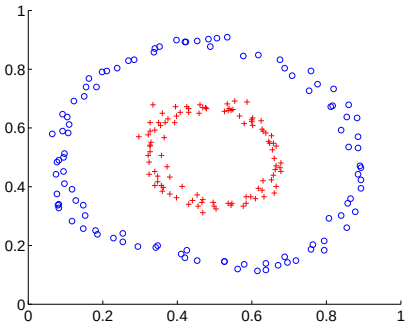

Figure 1. `TwoCircles` data set with correct clustering

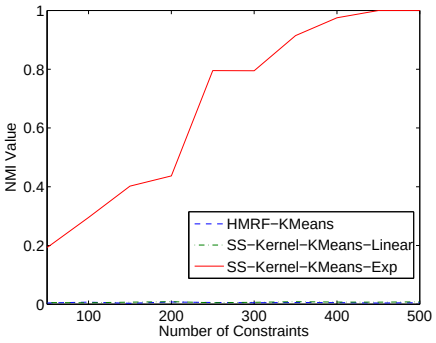

Figure 2. Results on `TwoCircles` data set

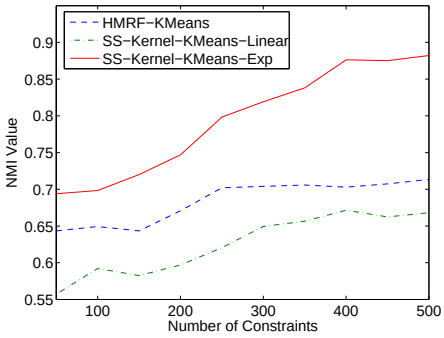

Figure 3. Results on `Digits` data set

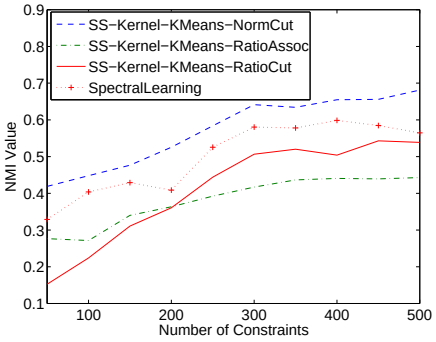

Figure 4. Results on `GeneNetwork` data set

4.2. Methodology

Figures 2-4 show learning curves with 2-fold cross validation. These plots show the improvement in clustering quality on the test set with increasing amount of pairwise supervision provided from the training set. Each point on the learning curve is an average of results over 20 runs. We generate constraints only using the training set, cluster the whole data without the labels, and evaluate the cluster quality only on the test set (Basu et al., 2004). We set each penalty weight w_{ij} to be equal to $n/(kC)$, where n is the number of data points, k is the number of clusters, and C is the total number of constraints. To evaluate clusters, we use Normalized Mutual Information (NMI): it is a standard technique for determining the quality of clusters, by measuring the amount of statistical information shared by the random variables representing the cluster distribution and the underlying class distribution of the data points (Strehl et al., 2000).

4.3. Results and Discussion

Figure 2 shows the results on the `TwoCircle` data set. This synthetic dataset is used to demonstrate the effectiveness of SS-KERNEL-KMEANS: this data set is not linearly separable in the original input space, but an exponential kernel $K(\mathbf{x}, \mathbf{y}) = \exp(-\|\mathbf{x} - \mathbf{y}\|^2/2\sigma^2)$ is able to linearly separate the 2 classes in the mapped space. SS-KERNEL-KMEANS with the exponential kernel (SS-Kernel-KMeans-Exp) gives improved results with increasing number of pairwise constraints, till it reaches a NMI score of 1.0 on the test set. On the other hand, HMRF-KMEANS and SS-KERNEL-KMEANS with a linear kernel (SS-Kernel-KMeans-Linear) are not able to get any improvement in performance on this dataset (NMI close to 0), since both algorithms split the data linearly into two clusters and are completely unable to re-construct the non-linear class structure, even with as many as 200 constraints.

Figure 3 shows the results on the `Digits` dataset. SS-KERNEL-KMEANS gives the best performance when used with an exponential kernel, outperforming both HMRF-KMEANS with squared Euclidean distance and SS-KERNEL-KMEANS with a linear kernel.

Since the `GeneNetwork` dataset is not vector-based, it cannot be clustered using HMRF-KMEANS. Three graph clustering objectives were used while clustering this dataset using SS-KERNEL-KMEANS —

normalized cut (SS-Kernel-KMeans-NormCut), ratio cut (SS-Kernel-KMeans-RatioCut) and ratio association (SS-Kernel-KMeans-RatioAssoc). The fourth plot in the figure (Spectral-Learning) corresponds to the SPECTRAL-LEARNING algorithm (Kamvar et al., 2003). As shown in Figure 4, all these algorithms have an improvement in performance with increasing supervision, but SS-KERNEL-KMEANS with normalized cut has the best performance for this dataset.

The benefit of using SS-KERNEL-KMEANS is clearly demonstrated in this experiment. A closer look showed that the `GeneNetwork` data set has 3 clusters of different sizes and different edge densities, which explains why performing degree normalization in the normalized cut objective gives good results. While SPECTRAL-LEARNING can only use the ratio cut objective, SS-KERNEL-KMEANS can work with other graph clustering objectives like normalized cut, therefore making it useful in domains where graph clustering objectives other than ratio cut are more effective.

5. Conclusions and Future Work

In this paper, a new algorithm SS-KERNEL-KMEANS was developed to optimize a semi-supervised clustering objective that can cluster both vector-based and graph-based data. The analysis for this algorithm used kernel methods: by constructing an appropriate kernel, we were able to prove an equivalence between a special case of the HMRF-based semi-supervised clustering objective and the kernel k-means objective function. The resulting algorithm provided a number of advantages over previously studied methods for semi-supervised clustering: (1) our analysis allows us to (locally) optimize the semi-supervised objective for both vector-based and graph-based inputs; (2) for vector-based inputs, the inputs can be mapped to a higher-dimensional kernel space to get non-linear cluster boundaries; (3) we can easily generalize the result to handle a number of semi-supervised graph clustering objectives; (4) the analysis allows us to link prior work on spectral learning to our semi-supervised clustering framework.

There are a number of interesting potential avenues for future research in kernel methods for semi-supervised clustering. Along with learning the kernel matrix before the clustering, one could additionally incorporate kernel matrix learning into the clustering iteration, as was done in Basu et al. (2004). One way to incorporate learning into the clustering step is to devise a way to learn the weights in weighted kernel k-means, by using the constraints. Another possibility would be to explore the generalization of techniques in this paper beyond squared Euclidean distance, for unifying semi-supervised graph clustering with kernel-based clustering on an HMRF using other popular clustering distortion measures, e.g., KL-divergence, cosine distance (Basu et al., 2004).

Acknowledgments: This research was supported by NSF grant CCF-0431257, NSF-ITR award IIS-0325116, NSF Career Award ACI-0093404, and a Google Research Grant.

References

Bansal, N., Blum, A., & Chawla, S. (2002). Correlation clustering. *Proc. of the 43rd IEEE Symp. on Foundations of Computer Science (FOCS-02)* (pp. 238–247).

Bar-Hillel, A., Hertz, T., Shental, N., & Weinshall, D. (2003). Learning distance functions using equivalence relations. *Proc. 20th Intl. Conf. on Machine Learning*.

Basu, S., Bilenko, M., & Mooney, R. (2004). A probabilistic framework for semi-supervised clustering. *Proc. 10th Intl. Conf. on Knowledge Discovery and Data Mining*.

Chan, P., Schlag, M., & Zien, J. (1994). Spectral k-way ratio cut partitioning. *IEEE Trans. CAD-Integrated Circuits and Systems*, *13*, 1088–1096.

Cristianini, N., & Shawe-Taylor, J. (2000). *Introduction to support vector machines*. Cambridge University Press.

Dhillon, I., Guan, Y., & Kulis, B. (2004a). Kernel k-means, spectral clustering and normalized cuts. *Proc. 10th Intl. Conf. on Knowledge Discovery and Data Mining*.

Dhillon, I., Guan, Y., & Kulis, B. (2004b). *A unified view of kernel k-means, spectral clustering and graph cuts* (Technical Report TR-04-25). University of Texas at Austin.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. Wiley.

Kamvar, S. D., Klein, D., & Manning, C. (2003). Spectral learning. *Proc. 17th Intl. Joint Conf. on Artificial Intelligence*.

Klein, D., Kamvar, D., & Manning, C. (2002). From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. *Proc. 19th Intl. Conf. on Machine Learning*.

Lee, I., Date, S. V., Adai, A. T., & Marcotte, E. M. (2004). A probabilistic functional network of yeast genes. *Science*, *306(5701)*, 1555–1558.

Ogata, H., Goto, S., Sato, K., Fujibuchi, W., Bono, H., & Kanehisa, M. (1999). KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, *27*, 29–34.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *22*, 888–905.

Strehl, A., Ghosh, J., & Mooney, R. (2000). Impact of similarity measures on web-page clustering. *Workshop on Artificial Intelligence for Web Search (AAAI)*.

Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. (2001). Constrained k-means clustering with background knowledge. *Proc. 18th Intl. Conf. on Machine Learning*.

Xing, E. P., Ng, A. Y., Jordan, M. I., & Russell, S. (2003). Distance metric learning, with application to clustering with side-information. *Advances in Neural Information Processing Systems 15*.

Yu, S., & Shi, J. (2004). Segmentation given partial grouping constraints. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, *26*, 173–183.