Adaptive KNN Classification Based on Laplacian Eigenmaps and Kernel Mixtures

Renqiang (Martin) Min*
Department of Computer Science
University of Toronto
Feb 2008

Abstract

K Nearest Neighbor (kNN) is one of the most popular machine learning techniques, but it often fails to work well with inappropriate choice of distance metric or due to the presence of a lot of irrelated features. Linear and non-linear feature transformation methods have been applied to extract class-relevant information to improve kNN classification. In this paper, I describe kNN classification in a large-margin framework, in which a non-linear feature mapping is sought through Laplacian eigenmaps or kernel mixtures, and then a linear transformation matrix is learned to achieve the goal of large margin.

1 Introduction

The classification technique k-Nearest Neighbor (kNN) is one of the most popular machine learning techniques. However, the good classification performance of kNN is highly dependent on the metric used for computing the distances between pairwise data points. In practice, we often use Euclidean distances as similarity metric to calculate the k nearest neighbors of data points of interest. In real applications with the presence of many irrelated features, we often need to learn or choose a good distance metric. Researchers have proposed distance metric learning to extract informative information from feature vectors to improve the classification performance of kNN, and linear transformation methods have been used to remove the redundant information in the feature vectors of data points and thereby to improve the performance of kNN [19, 6, 4] based on different objective functions. In many cases, it's impossible to improve kNN by linear transformation. Thus we need to resort to non-linear transformation so that each data point will stay together with all the data points in its neighborhood in the non-linearly transformed feature space. Kernel-based and neural-network-based non-linear dimensionality reduction method were used to improve kNN [13, 18, 11, 7]. However, the kernel-based approach behaves almost like a template-based approach, and kernel parameters are tuned using time-consuming cross validation. If the chosen kernel cannot well reflect the true structure of data, the resulting performance will be very bad. Although the previous neural-network-based approach can learn a very powerful non-linear feature transformation, it doesn't learn the feature mapping directly toward the goal of improving kNN classification.

In this paper, I propose two non-linear feature transformation methods directly toward the goal of improving kNN classification performance in a large-margin framework, which are respectively based on Laplacian eigenmaps (LE) [2] and Kernel Mixtures.

I will organize this paper as follows: in section 2, I will describe some related methods for distance metric learning and introduce some background knowledge for large-margin kNN classification. In section 3, I will describe large-margin kNN classification based on the non-linear feature mapping generated by LE and kernel mixtures. In section 4, I will discuss the described methods in this paper, show some variants of the described methods, and propose future research directions.

2 Related methods

Previous work on metric learning in [19] and [7] learns a global linear transformation matrix in the original feature space of data points to make similar data points stay closer while making dissimilar data points stay farther apart using additional similarity or label information. However, a global linear transformation

*This manuscript was prepared on Feb 29th, 2008.

is incapable of making data points in the same class stay apart from the data points in other classes. A locally adaptive distance metric learning is used in [10] but it is based on heuristics. In [6], a global linear transformation is applied to the original feature space of data points to learn Mahalanobis metrics, and a non-linear feature transformation is achieved using the kernel trick. Although this method can handle non-linear feature mappings, it requires all data points in the same class collapse to one point. This is unnecessary for kNN classification and may produce poor performance when data points cannot be essentially collapsed to points especially for large datasets. An information-theoretic based approach is used to learn linear transformations and extends to non-linear transformations by the kernel trick in [4]. In [11], a deep autoencoder is used to perform non-linear dimensionality reduction and kNN classification can be performed in the low-dimensional space. However, none of the above methods propose learning transformations in a large-margin framework which achieves the goal of improving kNN classification in a more direct and less restrictive way.

In [12], a global linear transformation is learned to directly improve kNN classification to achieve the goal of large margin. This method has been shown to yield significant improvement over kNN classification, but the linear transformation often fails to give good performance in high-dimensional space and a preprocessing dimensionality reduction step by PCA is often required for success. A recent paper [18] extends the work in [12] to perform linear dimensionality reduction to improve kNN classification in a large-margin framework. The kernelized version of the method in [18] handles linear dimensionality reduction in the high-dimensional feature space Φ induced by a kernel K by assuming each column of the linear transformation matrix as a linear combination of the feature vectors of data points in Φ. Although using kernels to induce non-linear feature transformations is effective and easy to implement, kernel-based methods are like template-based methods and a bad kernel choice will lead to very poor performance.

The approaches to be described are mainly inspired by the work in [12], [18] and [11], I perform non-linear feature extractions to improve large-margin kNN classification using LE and kernel mixtures. The non-linear mapping by LE is related to spectral clustering and allows an effective way to learn the parameters of the weight matrix for LE to better reflect the given neighborhood information of data points. The approach based on kernel mixtures presents us a soft way of choosing kernels instead of choosing one kernel in a hard way.

3 Large-margin kNN classification by non-linear feature mappings

3.1 Large-margin kNN classification

Given a set of data points $\mathcal{D} = \{\vec{x}_i, y_i : i = 1, \ldots, n\}$ and additional neighborhood information η, where $\vec{x}_i \in R^D$, $y_i \in \{1, \ldots, c\}$ for labeled data points, and $\eta_{il} = 1$ if l is one of i's k target neighbors, we seek a distance function $d(i, j)$ for pairwise data points i and j such that the given neighborhood information will be preserved in the transformed feature space corresponding to the distance function. If $d(i, j)$ is based on Mahanalobis distances, then it admits the following form:

$$d_{\mathbf{A}}(i, j) = (\vec{x}_i - \vec{x}_j)^T \mathbf{A}^T \mathbf{A} (\vec{x}_i - \vec{x}_j), \tag{1}$$

where $\mathbf{A}$ is a linear transformation matrix. Based on the goal of margin maximization, we learn the parameters of the distance function, $\mathbf{A}$, such that, for each data point i, the distance between i and each data point j from another class will be at least 1 plus the largest distance between i and its k target neighbors. Using a binary matrix $y_{ij} = 1$ to represent that i and j are in the same class and 0 otherwise for the labeled data points, we can formulate the above problem as an optimization problem:

$$min_{\mathbf{A}} \sum_{ij} \eta_{ij} d_{\mathbf{A}}(i, j) + \tag{2}$$

$$C \sum_{ijl} \eta_{il} (1 - y_{ij}) h(1 + d_{\mathbf{A}}(i, l) - d_{\mathbf{A}}(i, j)),$$

where C is a penalty coefficient penalizing constraint violations and h denotes hinge loss function. If $\mathbf{A}$ is a $D \times D$ matrix, this problem corresponds to the work in [12]; if $\mathbf{A}$ is a $d \times D$ matrix where $d < D$, this problem corresponds to the work in [18]. When a non-square matrix $\mathbf{A}$ is learned for dimensionality reduction, the resulting problem is non-convex, stochastic gradient descent and conjugate gradient descent are often used to solve the problem. When $\mathbf{A}$ is constrained to be a full-rank square matrix, we can solve $\mathbf{A}^T \mathbf{A}$ directly and the resulting problem is convex. Alternating projection or simple gradient-based methods can be applied here [12].

3.2 Large-margin kNN classification using Laplacian eigenmaps

Based on a weight matrix $\mathbf{W}$ for a dataset $\mathcal{D}$ as discussed in section 3.1, in which $\mathbf{W}_{ij}$ is the connection weight (or similarity score) between data point i and j, the Laplacian eigenmaps (LE) [2] finds d-dimensional embedding $\mathbf{Y}$ for the data points such that densely

connected data points by $\mathbf{W}$ will be placed together and loosely connected data points by $\mathbf{W}$ will be placed farther apart. We minimize the following objective function:

$$\begin{aligned} min_{\mathbf{Y}} \quad & \sum_{ij} \mathbf{W}_{ij} ||\mathbf{Y}_i - \mathbf{Y}_j||^2 \\ & = tr(\mathbf{Y}^T \mathbf{L} \mathbf{Y}), \\ s.t. \quad & \mathbf{Y}^T \mathbf{D} \mathbf{Y} = \mathbf{I} \end{aligned} \tag{3}$$

where tr denotes the trace of a matrix, $\mathbf{Y}$ is an $n \times d$ matrix and each row corresponds to one data point, $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{ii} = \sum_j \mathbf{W}_{ij}$, $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian, and $\mathbf{I}$ is a $d \times d$ identity matrix. The constraint of orthogonality on $\mathbf{Y}$ helps remove arbitrary scalings of $\mathbf{Y}$. The solution to the above problem is given by the bottom d eigenvector of the generalized eigenvalue problem $\mathbf{L}\vec{y} = \mathbf{D}\vec{y}$ excluding the very bottom eigenvector $\vec{e}$ where $\vec{e}$ is a vector with all components being 1 [2]. The very bottom eigenvector $\vec{e}$ corresponds to the smallest eigenvalue 0. In the following discussions, we relate LE to spectral clustering [17, 8, 20, 14, 1, 3] showing how to use the results in spectral clustering to learn the parameters of $\mathbf{W}$ in LE.

Suppose we have a graph G for n data points, each node in G represents a data point, and we also have a weight matrix $\mathbf{W}$ with $\mathbf{W}_{ij}$ being the connection weight (or similarity score) between data point i and j. For a d-way clustering, let an $n \times d$ binary matrix $\mathbf{E}$ represent the clustering result of the n data points, and $\mathbf{E}_{ir} = 1$ if data point i belongs to the r-th cluster and $\mathbf{E}_{ir} = 0$ otherwise. Spectral clustering minimizing normalized cut partitions the data points into d clusters by minimizing the following objective function (there are also spectral clustering minimizing ratio cut [9] and minmaxcut [5, 8]):

$$\begin{aligned} NCut(G, \mathbf{E}) &= \sum_{r=1}^{d} \frac{\sum_{i \in cluster(r), j \notin cluster(r)} \mathbf{W}_{ij}}{\sum_{i \in cluster(r), j \in \{1 \ldots n\}} \mathbf{W}_{ij}} \\ &= \sum_{r=1}^{d} \frac{\mathbf{E}_r^T (\mathbf{D} - \mathbf{W}) \mathbf{E}_r}{\mathbf{E}_r^T \mathbf{D} \mathbf{E}_r} \\ &= d - \sum_{r=1}^{d} \frac{(\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r)^T \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}} (\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r)}{(\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r)^T (\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r)} \\ &= d - \sum_{r=1}^{d} \frac{(\frac{\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r}{||\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r||_2})^T \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}} (\frac{\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r}{||\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r||_2})}{(\frac{\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r}{||\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r||_2})^T (\frac{\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r}{||\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r||_2})} \\ &= d - \sum_{r=1}^{d} \frac{\vec{y_r}^T \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}} \vec{y_r}}{\vec{y_r}^T \vec{y_r}} \\ &= d - tr(\mathbf{Y}^T \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}} \mathbf{Y}) \end{aligned} \tag{4}$$

where $\mathbf{E}_r$ is the r-th column of the cluster indicator matrix $\mathbf{E}$, $\vec{y_r} = \frac{\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r}{||\mathbf{D}^{\frac{1}{2}} \mathbf{E}_r||_2}$, $\mathbf{Y} = [\vec{y_1}, \vec{y_2}, \ldots, \vec{y_d}]$, and we can easily verify that $\mathbf{Y}^T \mathbf{Y} = \mathbf{I}$. Minimizing the objective in Equation 4 with respect to the indicator matrix $\mathbf{E}$ with an exact solution is an NP-hard problem. We can relax $\mathbf{Y}$ to be continuous and the solutions $\mathbf{Y}$ will be given by the top d eigenvectors of the normalized weight matrix $\widetilde{\mathbf{W}} = \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}}$ excluding the very top eigenvector having an eigenvalue 1 by Ky Fan's theorem. And $\mathbf{D}^{-\frac{1}{2}} \mathbf{Y}$ will be approximately piecewise constant with respect to the cluster indicator matrix $\mathbf{E}$.

Now I relate d-dimensional embedding by LE to d-way spectral clustering.

Theorem 1 *If λ' is an eigenvalue and $\vec{y}'$ is the corresponding eigenvector of the generalized eigenvalue problem $\mathbf{L}\vec{y} = \mathbf{D}\vec{y}$ for LE, then $1 - \lambda'$ and $\mathbf{D}^{\frac{1}{2}} \vec{y}'$ will be, respectively, the eigenvalue and the corresponding unnormalized eigenvector of $\widetilde{\mathbf{W}} = \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \mathbf{D}^{-\frac{1}{2}}$.*

By the problem definition, $\mathbf{L}\vec{y}' = \mathbf{D}\vec{y}'$, then we have, $\mathbf{W}\vec{y}' = \mathbf{D}\vec{y}' - \lambda' \mathbf{D}\vec{y}'$. Let λ'' and $\vec{y}''$ be the eigenvalue and eigenvector of the normalized weight matrix and let $\vec{y}'' = \mathbf{D}^{\frac{1}{2}} \vec{y}'$, we have,

$$\widetilde{\mathbf{W}} \vec{y}'' = \mathbf{D}^{-\frac{1}{2}} \mathbf{W} \vec{y}' = (1 - \lambda') \mathbf{D}^{\frac{1}{2}} \vec{y}' = \lambda'' \vec{y}'',$$

which proves our statement in Theorem 1.

Theorem 2 *The d-dimensional embedding found by Laplacian eigenmaps in Equation 3 is equivalent to a relaxed solution to the spectral clustering minimizing normalized cut in Equation 4 up to a fixed scaling.*

Proof. We can easily prove it using Theorem 1.

Since we have prior neighborhood or class information for a set of data points, we can find a d-dimensional embedding produced by LE that is consistent with the partition of the dataset according to the given neighborhood or class information. Because calculating the d-dimensional embedding is equivalent to finding a d-way clustering, we learn the parameters of the weight matrix such that the consistency between the relaxed solution to the problem in Equation 4 and the given partition of the dataset (target clustering/embedding) is as large as possible as in [1]. We optimize the weight matrix by maximizing the following objective function as in [1]:

$$\begin{aligned} max_{\mathbf{W}} \quad & \sum_{r=1}^{d} \frac{{\mathbf{E}_r}^T \mathbf{D}^{\frac{1}{2}} \mathbf{U} \mathbf{U}^T \mathbf{D}^{\frac{1}{2}} \mathbf{E}_r}{{\mathbf{E}_r}^T \mathbf{D} \mathbf{E}_r}, \\ s.t. \quad & \mathbf{U} \in R^{n \times d}, \mathbf{U} \text{ is any orthogonal basis of} \end{aligned}$$

the d-th principal subspace of $\widetilde{\mathbf{W}}$.

(5)

After obtaining a d-dimensional embedding $\mathbf{Y}$ by LE for a dataset $\mathcal{D}$, we can calculate a $d \times \ell$ linear transformation matrix using $\mathbf{Y}$ as feature vectors to improve large-margin kNN classification in a brute-force way by solving the optimization problem in Equation 2. In the following, we will prove that we do not need to calculate the embedding $\mathbf{Y}$ explicitly and learning a linear transformation matrix for $\mathbf{Y}$ is equivalent to working with a special kernel matrix directly if the normalized weight matrix is positive definite.

Theorem 3 *If the normalized weight matrix $\widetilde{\mathbf{W}}$ for n data points is positive definite and $\mathbf{Y}'$ is the relaxed solution to the d-way spectral clustering, the solution to learning a linear transformation matrix $\mathbf{A}$ for d-dimensional LE embedding $\mathbf{Y} = \mathbf{D}^{-\frac{1}{2}}\mathbf{Y}'$ to improve large-margin kNN classification in Equation 2 can be readily obtained by working on a kernel matrix $\mathbf{K} = \mathbf{D}^{-1}\mathbf{W}\mathbf{D}^{-1} - \frac{\vec{e}\vec{e}^T}{\vec{e}^T D\vec{e}}$ directly.*

Proof. Let $\widetilde{\mathbf{K}} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2} - \frac{\mathbf{D}^{1/2}\vec{e}}{||\mathbf{D}^{1/2}\vec{e}||}\frac{\vec{e}^T\mathbf{D}^{1/2}}{||\mathbf{D}^{1/2}\vec{e}||}$, we have $\mathbf{K} = \mathbf{D}^{-1/2}\widetilde{\mathbf{K}}\mathbf{D}^{-1/2}$, and let $\mathbf{V}$ contain all the eigenvectors of $\widetilde{\mathbf{W}}$ excluding the very top eigenvector having eigenvalue 1 as columns, $\mathbf{Y}^T = \mathbf{D}^{-\frac{1}{2}}\mathbf{V}$ (note that each row of $\mathbf{Y}$ represents one data point), and $\widetilde{\mathbf{K}} = \mathbf{V}\Lambda\mathbf{V}^T$, where $\mathbf{\Lambda}$ is diagonal and contains the corresponding eigenvalues of $\widetilde{\mathbf{W}}$ excluding 1. Because the normalized weight matrix is positive definite, $\mathbf{V}$ must span all the subspaces with dimensionality less than or equal to $n-1$. (I) Case $d = n-1$. Suppose $\mathbf{A}^*$ is a local (global) minimum found by explicitly using $\mathbf{Y}$, $\mathbf{A}^*$ must be able to be expressed in the form $\mathbf{A}^* = \mathbf{\Lambda}\widetilde{\mathbf{V}}^T\mathbf{Q}^*$ because the dimensionality of $\mathbf{A}^*$ is at most $n-1$, where $\mathbf{Q}$ is a linear transformation matrix, and $\widetilde{\mathbf{V}}$ can be $\mathbf{V}$ or only contains a subset of the columns of $\mathbf{V}$. Since $\mathbf{Q}$ is free to change and can be a low-rank matrix (with rank d), $\widetilde{\mathbf{V}} = \mathbf{V}$ will result in the same solution for $\mathbf{A}$. In fact, since the rank of $\widetilde{\mathbf{W}}$ is n, we can always use $\mathbf{Q}$ to parameterize the linear transformation matrix for a d-dimensional embedding where $d \leq n-1$. Note that the objective function in Equation 2 is linear with respect to the dot product between the feature vectors of pairwise data points.

$$\begin{aligned}\mathbf{Y}\mathbf{A}^*(\mathbf{A}^*)^T\mathbf{Y}^T &= \mathbf{D}^{-\frac{1}{2}}\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\mathbf{Q}^*\mathbf{Q}^{*T}\mathbf{V}\mathbf{\Lambda}\mathbf{V}^T\mathbf{D}^{-\frac{1}{2}} \\ &= \mathbf{D}^{-\frac{1}{2}}\widetilde{\mathbf{K}}\mathbf{D}^{-\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}\mathbf{Q}^*\mathbf{Q}^{*T}\mathbf{D}^{\frac{1}{2}}\mathbf{D}^{-\frac{1}{2}}\widetilde{\mathbf{K}}\mathbf{D}^{-\frac{1}{2}} \\ &= \mathbf{K}\mathbf{T}^*\mathbf{K},\end{aligned}$$

where $\mathbf{T}^* = \mathbf{D}^{\frac{1}{2}}\mathbf{Q}^*\mathbf{Q}^{*T}\mathbf{D}^{\frac{1}{2}}$. From the above derivation, we can see that, if $\mathbf{A}^*$ is a local or global minimum minimizing the objective in Equation 2 using the feature vectors $\mathbf{Y}$ for $d = n-1$, $\mathbf{T}^*$ must be a local or global minimum of the same problem using the kernel matrix K as feature vectors. (II) Case $d < n-1$. We can embed the data based on LE to $n-1$ dimensional space first, then we learn an $n-1$ by d linear transformation matrix B, if the $n-1-d$ extra dimensions don't help further reduce the objective in Eq. 2, B will have some zero rows and be degenerated into a dxd matrix. Therefore, the solution B will be as at least equally good as the solution obtained using d-dimensional LE embedding.

Theorem 4 *To improve large-margin kNN classification, learning a linear transformation matrix $\mathbf{A}$ in the high-dimensional feature space induced by a kernel matrix $\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}$ through minimizing the objective function in Equation 2, is equivalent to learning a linear transformation matrix $\mathbf{A}'$ by using columns of $\mathbf{K}$ as new feature vectors,if we constrain that $\mathbf{A} = \mathbf{A}'\Phi^T$, that is, each row of $\mathbf{A}$ is constrained to be the linear combination of the feature vectors of data points in the high-dimensional space.*

Proof. Since the objective function in Eq. 2 is linear with respect to the dot products between the feature vectors of pairwise data points,

$$\begin{aligned}\mathbf{\Phi}^T\mathbf{A}^T\mathbf{A}\mathbf{\Phi} &= \mathbf{\Phi}^T\Phi\mathbf{A}'^T\mathbf{A}'\Phi^T\Phi \\ &= \mathbf{K}\mathbf{A}'^T\mathbf{A}'\mathbf{K}\end{aligned}$$

The above derivation obviously shows that constraining the learned linear transformation matrix in the high-dimensional feature space is exactly equivalent to using $\mathbf{K}$ as feature vectors to learn a linear transformation matrix.

3.3 Large-margin kNN classification using kernel mixtures

As discussed in section 3.2, learning a linear transformation matrix based on the feature vectors produced by LE is equivalent to working with a special kernel matrix that is easily derived from the weight matrix for LE, if the normalized weight matrix $\widetilde{\mathbf{W}}$ is positive semi-definite. The work in [18, 4, 6] used one kernel to kernelize distance metric learning by constraining the learned linear transformation matrix following the idea from [15, 16]. Theorem 4 shows that it is equivalent to using the rows of the based kernel as feature vectors to learn a linear transformation matrix.

Since kernel-based methods are almost like template-based methods, choosing a good kernel best suitable for the problem at hand is often difficult. An undesirable choice often leads to poor performance. Instead of selecting kernels in a hard way, we propose using

kernel mixtures, a convex combination of a set of candidate kernels, to select kernel in a soft and flexible way.

Given m candidate kernels $\{\mathbf{K}^{(1)}, \ldots, \mathbf{K}^{(m)}\}$, let $\mathbf{K} = \sum_{k=1}^{m} \theta_k \mathbf{K}^{(k)}$, we will use the columns of the combined kernel $\mathbf{K}$ as feature vectors to learn a linear transformation matrix $d \times n$ A, following Theorem 4, this is equivalent to learning a constrained linear transformation matrix in the high-dimensional feature space induced by the kernel $\mathbf{K}$. Thus, we will have the following optimization problem:

$$\begin{aligned} min_{\mathbf{A},\vec{\theta}} \quad & \sum_{ij} \eta_{ij} d_{\mathbf{A}}(i,j) + \\ & C \sum_{ijl} \eta_{il}(1-y_{ij}) h(1 + d_{\mathbf{A}}(i,l) - d_{\mathbf{A}}(i,j)), \\ s.t. \quad & \mathbf{K} = \sum_{k=1}^{m} \theta_k \mathbf{K}^{(k)}, \\ & d_{\mathbf{A}}(i,j) = (K_i - K_j)^T \mathbf{A}^T \mathbf{A} (K_i - K_j) \\ & \sum_k \theta_k = 1 \\ & \theta_k \geq 0, k = 1, \ldots, m, \end{aligned} \tag{6}$$

where K_i is the i-th column of kernel matrix K. The above problem is not convex, we can optimize $\mathbf{A}$ and $\vec{\theta}$ using simple stochastic gradient descent in an iterative and alternating fashion until convergence.

Instead of making kernel K be a convex combination of a pre-defined set of template kernels, we can also make the distance $D(i,j)$ be the convex combination of a set of distances computed from an ensemble of kernels. In details, $D(i,j) = \sum_k \theta_k D_k(i,j)$, where $D_k(i,j) = (K_i^{(k)} - K_j^{(k)})^T A^{(k)^T} A^{(k)} (K_i^{(k)} - K_j^{(k)})$, $K_i^{(k)}$ is the i-th column of the k-th kernel $K^{(k)}$. Then we have the following optimization problem:

$$\begin{aligned} min_{\mathbf{A}^{(1)},\ldots,A^{(k)},\vec{\theta}} \quad & \sum_{ij} \eta_{ij} D(i,j) + \\ & C \sum_{ijl} \eta_{il}(1-y_{ij}) \\ & h(1 + D(i,l) - D(i,j)), \\ s.t. \quad & D(i,j) = \sum_{k=1}^{m} \theta_k D_k(i,j), \\ & D_k(i,j) = (K_i^{(k)} - K_j^{(k)})^T \\ & A^{(k)^T} A^{(k)} (K_i^{(k)} - K_j^{(k)}), \\ & \sum_k \theta_k = 1, \\ & \theta_k \geq 0, k = 1, \ldots, m. \end{aligned} \tag{7}$$

In the above optimization problem, $\mathbf{A}^{(1)}, \ldots, A^{(k)}$ and $\vec{\theta}$ are parameters to be learned, or we can learn $\{\mathbf{Q}^{(k)} = A^{(k)^T} A^{(k)}\}$ directly. These parameters can be optimized using an EM-like algorithm, which is analogous to estimating the covariance matrices and the prior probabilities in a Gaussian Mixture Model (GMM) although there are some subtle differences here. Unlike the objective function in GMM, we have a minimization function here, consisting of the sum of hinge losses. We compute $\vec{\theta}$ in a way similar to estimating the prior probability in GMM. Given $\{\mathbf{A}^{(k)}\}$ computed in the previous step, we can assign each data point i to only one of the m kernels, say kernel k, which has the smallest distance margin violation $\ell_i^{(k)}$ among the m kernels defined as follows:

$$\ell_i^{(k)} = \sum_{jl} \eta_{il}(1-y_{ij}) h(1 + D_k(i,l) - D_k(i,j)), \quad k = 1, \ldots, m. \tag{8}$$

After $\vec{\theta}$ is computed, we then turn to estimate $\{\mathbf{A}^{(k)}\}$ as estimating covariance matrices in GMM. This EM-like procedure iterates until convergence criteria is satisfied.

4 Discussions

In this paper, I proposed using non-linear feature mappings found by Laplacian Eigenmaps and kernel mixtures to embed a set of data points into a new feature space, and then I learn a linear transformation matrix to improve kNN classification in a max-margin way. I show that, to achieve the large-margin goal, learning a linear transformation for the feature vectors produced by Laplacian Eigenmaps is equivalent to directly working with a kernel that is easily derived from the weight matrix for Laplacian Eigenmaps. This result enables us to learn a linear transformation matrix for LE on a special kernel directly. I also relate Laplacian Eigenmaps to Spectral Clustering for the purpose of defining similarity measure for kNN classification (also discussed in previous research work for other purposes), and this allows us to use the given partition information of labeled data to learn the weight matrix for Laplacian Eigenmaps.

In the future, I plan to use a deep auto-encoder as in [11] to learn a low-dimensional embedding and at the same time I will use the label information to gently tune part of the embedding vector to improve large-margin kNN classification. To favour classification, I will try to make part of the components of the embedding vector focus on reconstructing the original input feature vector. Besides, in the RBF kernel used in LE, I only used one free parameter σ. I believe that the performance of large-margin kNN classification based on LE can be further improved if a diagonal

covariance matrix is learned. I will investigate efficient approaches for approximating the subspace of a large parametric matrix which is used in LE (see section 3.2) instead of using expensive power method [1].

References

[1] F. Bach and M. Jordan. Learning spectral clustering. *Technical report,UC Berkeley*, 2003.

[2] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003.

[3] F. R. K. Chung. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*. American Mathematical Society, February 1997.

[4] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 209–216, New York, NY, USA, 2007. ACM.

[5] C. H. Q. Ding, X. He, H. Zha, M. Gu, and H. D. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 107–114, Washington, DC, USA, 2001. IEEE Computer Society.

[6] A. Globerson and S. Roweis. Metric learning by collapsing classes. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *NIPS 18*, pages 451–458. MIT Press, Cambridge, MA, 2006.

[7] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *NIPS 17*, pages 513–520. MIT Press, Cambridge, MA, 2005.

[8] M. Gu, H. Zha, C. Ding, X. He, and H. Simon. Spectral relaxation models and structure analysis for k-way graph clustering and bi-clustering, 2001.

[9] L. W. Hagen and A. B. Kahng. Net partitions yield better module partitions. Number 9, pages 1074–1085, September 1992.

[10] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(6):607–616, 1996.

[11] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[12] J. B. K. Weinberger and L. Saul. Distance metric learning for large margin nearest neighbor classification. In Y. Weiss and B. Sch editors, *NIPS 18*.

[13] R. Min. A non-linear dimensionality reduction method for improving nearest neighbour classification. *Thesis, DCS, University of Toronto*, 2005.

[14] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. 2001.

[15] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[16] B. Schölkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, December 2001.

[17] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[18] L. Torresani and K. chih Lee. Large margin component analysis. In B. Sch editor, *NIPS 19*. MIT Press.

[19] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. T. S. Becker and K. Obermayer, editors, *NIPS 15*. MIT Press.

[20] H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. *Neural Info. Processing Systems (NIPS 2001)*, pages 1057–1064, 2001.