Self-Supervised Learning for Object Recognition based on Kernel Discriminant-EM Algorithm

Ying Wu, Thomas S. Huang
Beckman Institute
Univ. of Illinois at Urbana-Champaign
405 N. Mathews, Urbana, IL 61801
{yingwu,huang}@ifp.uiuc.edu

Kentaro Toyama
Microsoft Research
One Microsoft Way
Redmond, WA 98052
kentoy@microsoft.com

Abstract

It is often tedious and expensive to label large training data sets for learning-based object recognition systems. This problem could be alleviated by self-supervised learning techniques, which take a hybrid of labeled and unlabeled training data to learn classifiers. Discriminant-EM (D-EM) proposed a framework for such tasks and current D-EM algorithm employed linear discriminant analysis. However, the algorithm is limited by its dependence on linear transformations. This paper extends the linear D-EM to nonlinear kernel algorithm, Kernel D-EM, *based on kernel multiple discriminant analysis (KMDA). KMDA provides better ability to simplify the probabilistic structures of data distributions in a discrimination space. We propose two novel data-sampling schemes for efficient training of kernel discriminants. Experimental results show that classifiers using KMDA learning compare with SVM performance on standard benchmark tests, and that* Kernel D-EM *outperforms a variety of supervised and semi-supervised learning algorithms for a hand-gesture recognition task and fingertip tracking task.*

1 Introduction

Invariant object recognition is a fundamental but challenging computer vision task, since finding effective object representations is generally a difficult problem. 3D object reconstruction suggests a way to invariantly characterize objects. Alternatively, objects could also be represented by their visual appearance without explicit reconstruction. However, representing objects in the image space is formidable, since the dimensionality of the image space is intractable. Dimension reduction could be achieved by identifying invariant image features. In some cases, domain knowledge could be exploited to extract image features from visual inputs, however, many other cases need to *learn* such features from a set of examples when image features are difficult to define. Many successful examples of learning approaches in the area of face and gesture recognition can be found in the literature [5, 2].

Generally, characterizing objects from examples requires huge training data sets, because input dimensionality is large and the variations that object classes undergo are significant. Labels or supervised information of training samples are needed for recognition tasks. The generalization abilities of many current methods largely depend on training data sets. In general, good generalization requires large and representative labeled training data sets. Unfortunately, collecting labeled data can be a tedious, if not altogether impossible, process. Although unsupervised or clustering schemes have been proposed [1, 14], it is difficult for pure unsupervised approaches to achieve accurate classification without supervision.

This problem can be alleviated by *semi-supervised* or *self-supervised* learning techniques which take hybrid training data sets. This learning paradigm could be looked as an integration of pure supervised and unsupervised learning. These algorithms assume that only a fraction of the data is labeled with ground truth, but still take advantage of the entire data set to generate good classifiers; they make the assumption that nearby data are likely to be generated by the same class. Work in this area has been successfully applied to text classification [3, 4, 7, 10].

Discriminant-EM (D-EM) [15] is a self-supervised learning algorithm for such purposes by taking a small set of labeled data with a large set of unlabeled data. The basic idea of this algorithm is to learn discriminating features and the classifier simultaneously by inserting a multi-class linear discrminant step in the standard expectation-maximization iteration loop. D-EM makes assumption that the probabilistic structure of data distribution in the lower dimensional discrimination space is simplified and could be captured by lower order Gaussian mixtures. Because the discrimination step in D-EM is linear, however, it has difficulty han-

0-7695-1143-0/01 $10.00 © 2001 IEEE

dling data sets which are not linearly separable, and input data is likely to be highly non-linearly-separable, regardless of the features used as input.

Based on nonlinear kernel discriminant analysis, this paper presents a *Kernel D-EM* algorithm. Kernel discriminant analysis transforms the original data space $\mathcal{X}$ to a higher dimensional kernel feature space $\mathcal{F}$ and then projects to a lower dimensional discrimination space Δ, such that nonlinear discriminating features could be identified, and training data could be better classified in a nonlinear feature space. Two novel algorithms are presented for sampling training data for efficient learning of nonlinear kernel discriminants. Our experiments include standard benchmark testing, view-independent hand posture recognition and invariant fingertip tracking.

2 Nonlinear Discriminant Analysis

Nonlinear discriminant analysis could be achieved by transforming the original data space $\mathcal{X}$ to a nonlinear feature space $\mathcal{F}$ and then performing LDA in $\mathcal{F}$. This section presents a kernel-based approach.

2.1 Linear Multiple Discriminant Analysis

Multiple discriminant analysis (MDA) is a natural generalization of Fisher's linear discriminant analysis (LDA) for the case of multiple classes [6]. The goal of MDA is to find a linear projection $\mathbf{W}$ that maps the original d_1-dimensional data space $\mathcal{X}$ to a d_2-dimensional discrimination space Δ ($d_2 \leq c-1$, c is the number of classes) such that the classes are linearly separable.

More specifically, MDA finds the best linear projection of labeled data, $\mathbf{x} \in \mathcal{X}$, such that the ratio of between-class scatter, S_B, to within-class scatter, S_W, is maximized. Let n be the size of training data set, and n_j be the size of the data set for class j. Then,

$$\mathbf{V}_{opt} = \arg\max_{\mathbf{V}} \frac{|\mathbf{V}^T S_B \mathbf{V}|}{|\mathbf{V}^T S_W \mathbf{V}|}, \quad (1)$$

$$S_B = \sum_{j=1}^{c} n_j (\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T, \quad (2)$$

$$S_W = \sum_{j=1}^{c} \sum_{k=1}^{n_j} (\mathbf{x}_k - \mathbf{m}_j)(\mathbf{x}_k - \mathbf{m}_j)^T, \quad (3)$$

where the total mean and class means are given by $\mathbf{m} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k$, $\mathbf{m}_j = \frac{1}{n_j}\sum_{k=1}^{n_j} \mathbf{x}_k$, for $j = 1, \ldots, c$, and $\mathbf{V}_{opt} = [\mathbf{v}_1, \ldots, \mathbf{v}_{c-1}]$ will contain in its columns $c-1$ eigenvectors corresponding to $c-1$ eigenvalues, i.e., $S_B \mathbf{v}_i = \lambda_i S_W \mathbf{v}_i$.

2.2 Kernel Discriminant Analysis

In *nonlinear* discriminant analysis, we seek a prior transformation of the data, $\mathbf{y} = \phi(\mathbf{x})$, that maps the original data space $\mathcal{X}$, to a feature space (F-space) $\mathcal{F}$, in which MDA can be then performed. Thus, we have

$$\mathbf{V}_{opt} = \arg\max_{\mathbf{V}} \frac{|\mathbf{V}^T S_B^{\phi} \mathbf{V}|}{|\mathbf{V}^T S_W^{\phi} \mathbf{V}|}, \quad (4)$$

$$S_B^{\phi} = \sum_{j=1}^{c} n_j (\mathbf{m}_j - \mathbf{m})(\mathbf{m}_j - \mathbf{m})^T, \quad (5)$$

$$S_W^{\phi} = \sum_{j=1}^{c} \sum_{k=1}^{n_j} (\phi(\mathbf{x}_k) - \mathbf{m}_j)(\phi(\mathbf{x}_k) - \mathbf{m}_j)^T, \quad (6)$$

with $\mathbf{m} = \frac{1}{n}\sum_{k=1}^{n} \phi(\mathbf{x}_k)$, $\mathbf{m}_j = \frac{1}{n_j}\sum_{k=1}^{n_j} \phi(\mathbf{x}_k)$, where $j = 1, \ldots, c$.

In general, because we choose $\phi(\cdot)$ to facilitate *linear* discriminant analysis in the feature space $\mathcal{F}$, the dimension of the feature space may be arbitrarily large, even infinite. As a result, the explicit computation of the mapping induced by $\phi(\cdot)$ could be prohibitively expensive.

The problem can be made tractable by taking a kernel approach that has recently been used to construct nonlinear versions of support vector machines [13], principal components analysis [12], and invariant feature extraction [9, 11]. Specifically, the observation behind kernel approaches is that if an algorithm can be written in such a way that only dot products of the transformed data in $\mathcal{F}$ need to be computed, explicit mappings of individual data from $\mathcal{X}$ become unnecessary.

Referring to Equation 4, we know that any column of the solution $\mathbf{V}$, must lie in the span of all training samples in $\mathcal{F}$, i.e., $\mathbf{v}_i \in \mathcal{F}$. Thus, for some $\underline{\alpha} = [\alpha_1, \cdots, \alpha_n]^T$,

$$\mathbf{v} = \sum_{k=1}^{n} \alpha_k \phi(\mathbf{x}_k) = \Phi \underline{\alpha}, \quad (7)$$

where $\Phi = [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_n)]$. We can therefore project a data point $\mathbf{x}_k$ onto one coordinate of the linear subspace of $\mathcal{F}$ as follows (we will drop the subscript on $\mathbf{v}_i$ in the ensuing):

$$\mathbf{v}^T \phi(\mathbf{x}_k) = \underline{\alpha}^T \Phi^T \phi(\mathbf{x}_k) \quad (8)$$

$$= \underline{\alpha}^T \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_k) \\ \vdots \\ k(\mathbf{x}_n, \mathbf{x}_k) \end{bmatrix} = \underline{\alpha}^T \xi_k, \quad (9)$$

$$\xi_k = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_k) \\ \vdots \\ k(\mathbf{x}_n, \mathbf{x}_k) \end{bmatrix}, \quad (10)$$

where we have rewritten dot products, $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$, with kernel notation, $k(\mathbf{x}, \mathbf{y})$. Similarly, we can project

each of the class means onto an axis of the feature space subspace using only dot products:

$$\mathbf{v}^T\mathbf{m}_j = \underline{\alpha}^T \frac{1}{n_j}\sum_{k=1}^{n_j}\begin{bmatrix}\phi^T(\mathbf{x}_1)\phi(\mathbf{x}_k)\\ \vdots \\ \phi^T(\mathbf{x}_n)\phi(\mathbf{x}_k)\end{bmatrix} \quad (11)$$

$$= \underline{\alpha}^T\begin{bmatrix}\frac{1}{n_j}\sum_{k=1}^{n_j}k(\mathbf{x}_1,\mathbf{x}_k)\\ \vdots \\ \frac{1}{n_j}\sum_{k=1}^{n_j}k(\mathbf{x}_n,\mathbf{x}_k)\end{bmatrix} \quad (12)$$

$$= \underline{\alpha}^T\mu_j. \quad (13)$$

It follows that

$$\mathbf{v}^T S_B \mathbf{v} = \underline{\alpha}^T K_B \underline{\alpha}, \quad (14)$$

where $K_B = \sum_{j=1}^{c} n_j(\mu_j - \mu)(\mu_j - \mu)^T$, and

$$\mathbf{v}^T S_W \mathbf{v} = \underline{\alpha}^T K_W \underline{\alpha}, \quad (15)$$

where $K_W = \sum_{j=1}^{c}\sum_{k=1}^{n_j}(\xi_k - \mu_j)(\xi_k - \mu_j)^T$. The goal of Kernel Multiple Discriminant Analysis (KMDA), then, is to find

$$\mathbf{A}_{opt} = \arg\max_{\mathbf{A}} \frac{|\mathbf{A}^T K_B \mathbf{A}|}{|\mathbf{A}^T K_W \mathbf{A}|}, \quad (16)$$

where $\mathbf{A} = [\underline{\alpha}_1, \cdots, \underline{\alpha}_{c-1}]$, and computation of K_B and K_W requires only kernel computations.

3 Sampling Data for Efficiency

Because K_B and K_W are $n \times n$ matrices, where n is the size of training set, the nonlinear mapping is dependent on the entire training samples. For large n, the solution to the generalized eigensystem is costly. Approximate solutions could be obtained by sampling representative subsets of the training data, $\{p_k | k = 1, \ldots, M, M < n\}$, and using $\tilde{\xi}_k = [k(\mathbf{x}_1, \mathbf{x}_k), \cdots, k(\mathbf{x}_M, \mathbf{x}_k)]^t$ to take the place of ξ_k.

3.1 PCA-based Kernel Vector Selection

The first approach we propose is blind to the class labeling. We select representatives, or *kernel vectors*, by identifying those training samples which are likely to play a key role in $\Xi = [\xi_1, \ldots, \xi_n]$. Ξ is an $n \times n$ matrix, but $rank(\Xi) \ll n$, when the size of training data set is very large. This fact suggests that some training samples could be ignored in calculating kernel features ξ.

We first compute the principal components of Ξ. Denote the $n \times n$ matrix of concatenated eigenvectors with $\mathbf{P}$. Thresholding elements of $\mathtt{abs}(\mathbf{P})$ by some fraction of the largest element of it allows us to identify salient PCA coefficients. For each column corresponding to a non-zero eigenvalue, choose the training samples which correspond to a salient PCA coefficient, i.e., choose the training samples corresponding to rows that survived the thresholding. Do so for every non-zero eigenvalue and we arrive at a decimated training set, which represents data at the periphery of each data cluster.

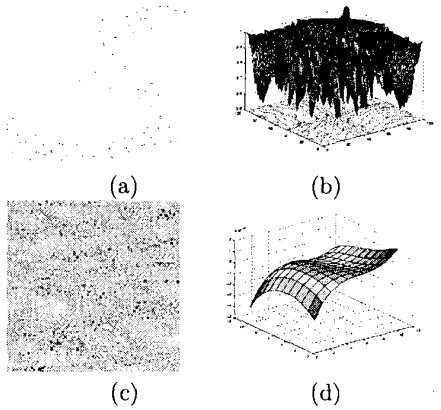


Figure 1: KMDA with a 2D 2-class non-linearly-separable example. (a) Original data (b) the kernel features of the data (c) the normalized coefficients of PCA on Ξ, in which only a small number of them are large (in black) (d) the nonlinear mapping.

3.2 Evolutionary Kernel Vector Selection

Another approach is to take advantage of class labels in the data. We maintain a set of kernel vectors at every iteration which are meant to be the key pieces of data for training. M initial kernel vectors, $KV^{(0)}$, are chosen at random. At iteration k, we have a set of kernel vectors, $KV^{(k)}$, which are used to perform KMDA such that the nonlinear projection $\mathbf{y}_i^{(k)} = \mathbf{V}^{(k)T}\phi(\mathbf{x}_i) = \mathbf{A}_{opt}^{(k)T}\xi_I^{(k)} \in \Delta$ of the original data $\mathbf{x}_i$ can be obtained. We assume Gaussian distribution $\theta^{(k)}$ for each class in the nonlinear discrimination space Δ, and the parameters $\theta^{(k)}$ can be estimated by $\{\mathbf{y}^{(k)}\}$, such that the labeling and training error $e^{(k)}$ can be obtained by $\bar{l}_i^{(k)} = \arg\max_j p(l_j|\mathbf{y}_i, \theta^{(k)})$.

If $e^{(k)} < e^{(k-1)}$, we randomly select M training samples from the correctly classified training samples as kernel vector $KV^{(t+1)}$ at iteration $k+1$. Another possibility is that if any current kernel vector is correctly classified, we randomly select a sample in its topological neighborhood to replace this kernel vector in the next iteration. Otherwise, i.e., $e^{(k)} \geq e^{(k-1)}$, and we terminate.

The evolutionary kernel vector selection algorithm is summarized below in Figure 2.

Evolutionary Kernel Vector Selection: Given a set of training data $\mathcal{D} = (X, L) = \{(\mathbf{x}_i, l_i), i = 1, \ldots, N\}$, to identify a set of M kernel vectors $KV = \{\nu_i, i = 1, \ldots, M\}$.

```
k = 0; e = ∞; KV^(0) =random_pick(X); // Init
do{
    A_opt^(k) =KMDA(X, KV^(k));// Perform KMDA
    Y^(k) =Proj(X, A_opt^(k)); // Project X to Δ
    Θ^(k) =Bayes(Y^(k), L); //Bayesian classifier
    L̄^(k) =Labeling(Y^(k), Θ^(k)); // Classification
    e^(k) =Error(L̄^(k), L); // Calculate error
    if(e^(k) < e)
       e = e^(k); KV = KV^(k); k++;
       KV^(k) =random_pick({x_i : l̄_i^(k) ≠ l_i});
    else
       KV = KV^(k-1); break;
    end
}
return KV;
```

Figure 2: Evolutionary Kernel Vector Selection

4 Kernel D-EM Algorithm

As an extension to Expectation-Maximization (EM), [15] proposed a three-step algorithm, called Discriminant-EM (D-EM), which loops between an expectation step, a discrimination step (via MDA), and a maximization step. D-EM estimates the parameters of a generative model in a discrimination space.

We now apply KMDA to D-EM. *Kernel D-EM (KDEM)* is a generalization of D-EM, in which instead of a simple linear transformation of the data, KMDA is used to project the data nonlinearly into a feature space where the data is better separated linearly. The nonlinear mapping, $\phi(\cdot)$, is implicitly determined by the kernel function, which must be determined in advance. The transformation from the original data space $\mathcal{X}$ to the discrimination space Δ, which is a linear subspace of the feature space $\mathcal{F}$, is given by $\mathbf{V}^T\phi(\cdot)$ implicitly or $\mathbf{A}^T\xi$ explicitly. A low-dimensional generative model is used to capture the transformed data in Δ.

$$p(l|\Theta) = \sum_{j=1}^{c} p(\mathbf{V}^T\phi(\mathbf{x})|c_j;\theta_j)p(c_j|\theta_j) \qquad (17)$$

Empirical observations suggest that the transformed data often approximates a Gaussian in Δ, and so in our current implementation, we use low-order Gaussian mixtures to model the transformed data in Δ. Kernel D-EM can be initialized by selecting all labeled data as kernel vectors, and training a weak classifier based on only unlabeled samples. Then, the three steps of Kernel D-EM are iterated until some appropriate convergence criterion:

- E-step: set $\hat{\mathcal{Z}}^{(k+1)} = E[\mathcal{Z}|\mathcal{D}; \hat{\Theta}^{(k)}]$
- D-step: set $\mathbf{A}_{opt}^{k+1} = \arg\max_A \frac{|\mathbf{A}^T K_B \mathbf{A}|}{|\mathbf{A}^T K_W \mathbf{A}|}$, and identify kernel vectors $KV^{(k+1)}$
- M-step: set $\hat{\Theta}^{(k+1)} = \arg\max_\theta p(\Theta|\mathcal{D}; \hat{\mathcal{Z}}^{(k+1)})$

The E-step gives unlabeled data probabilistic labels, which are then used by the D-step to separate the data. As mentioned before, this assumes that the class distributions are moderately smooth.

5 Experiments

In this section, we compare KMDA with other supervised learning techniques on some standard data sets. Experimental results of Kernel D-EM on hand posture recognition and invariant fingertip tracking are presented.

5.1 Benchmark Test for KMDA

We first verify the ability of KMDA with our data-sampling algorithms. Several benchmark data sets[1] are used in our experiments. The benchmark data has 100 different realizations. In [9], results of different approaches on these data sets have been reported. The proposed KMDA algorithms were compared to a single RBF classifier (RBF), a support vector machine (SVM), AdaBoost, and the kernel Fisher discriminant (KFD) [8]. RBF kernels were used in all kernel-based algorithms.

Benchmark	Banana	B-Cancer	Heart
RBF	10.8±0.06	27.6±0.47	17.6±0.33
AdaBoost	12.3±0.07	30.4±0.47	20.3±0.34
SVM	11.5±0.07	26.0±0.47	16.0±0.33
KFD	10.8±0.05	25.8±0.46	16.1±0.34
KMDA-pca	10.7±0.25	27.5±0.47	16.5±0.32
KMDA-evol	10.8±0.56	26.3±0.48	16.1±0.33
#-KVs	120	40	20

Table 1: Benchmark Test: the average test error as well as standard deviation.

In Table 1, KMDA-pca is KMDA with PCA selection, and KMDA-evol is KMDA with evolutionary selection, where #-KVs is the number of kernel vectors. The benchmark tests show that the proposed

[1]The standard benchmark data sets in our experiments are obtained from `http://www.first.gmd.de/~raetsch`.

approaches achieve comparable results as other state-of-the-art techniques, in spite of the use of a decimated training set.

5.2 Hand Posture Recognition

Next, we examine results for KDEM on a hand gesture recognition task. The task is to classify among 14 different hand postures, each of which represents a gesture command mode, such as navigating, pointing, grasping, etc. Our raw data set consists of 14,000 unlabeled hand images together with 560 labeled images (approximately 40 labeled images per hand posture), most from video of subjects making each of the hand postures. These 560 labeled images are used to test the classifiers by calculating the classification errors.

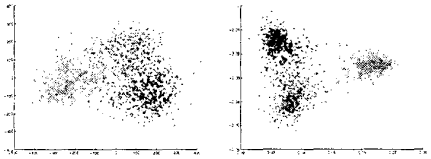
Hands are localized in video sequences by adaptive color segmentation and hand regions are cropped and converted to gray-level images [15]. Gabor wavelet filters with 3 levels and 4 orientations are used to extract 12 texture features. 10 coefficients from the Fourier descriptor of the occluding contour are used to represent hand shape. We also use area, contour length, total edge length, density, and 2nd moments of edge distribution, for a total of 28 low-level image features (I-Feature). For comparison, we also represent images by coefficients of the 22 largest principal components of the total data set resized to 20×20 pixels (these are "eigenimages", or E-Features) [15]. In our experiments, we use 140 (10 for each) and 10000 (randomly selected from the whole database) labeled and unlabeled images respectively, for training with both EM and D-EM. Table 2 shows the comparison.

Algorithm	MLP	NN-G	EM	LDEM	KDEM
I-Feature	33.3%	15.8%	21.4%	9.2%	5.3%
E-Feature	39.6%	20.3%	20.8%	7.6%	4.9%

Table 2: View-independent hand posture recognition: Comparison among multilayer perceptron (MLP),Nearest Neighbor with growing templates (NN-G), EM, linear D-EM (LDEM) and KDEM

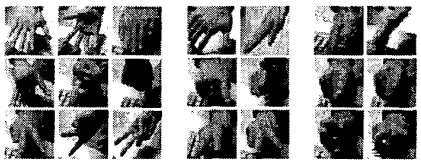
We observed that multilayer perceptrons are often trapped in local minima and nearest neighbor suffers from the sparsity of the labeled templates. The poor performance of pure EM is due to the fact that the generative model does not capture the ground-truth distribution well, since the underlying data distribution is highly complex. It is not surprising that LDEM and KDEM outperform other methods, since the D-step optimizes separability of the classes.

Finally, note the effectiveness of KDEM. We find that KDEM often appears to project classes to approximately Gaussian clusters in the transformed space, which facilitates their modeling with Gaussians. Figure 4 shows typical transformed data sets for linear and nonlinear discriminant analysis, in a projected 2D subspace of 3 different hand postures.



(a) (b)

Figure 3: Data distribution in the projected subspace (a) Linear KMDA (b) Kernel KMDA. Different postures are more separated and clustered in the nonlinear subspace by KMDA.



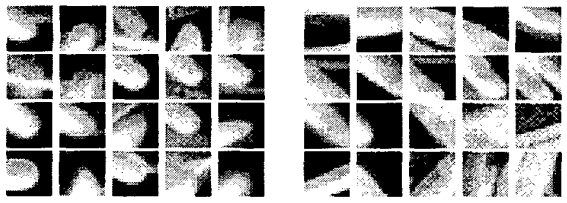
(a) (b) (c)

Figure 4: (a) Some correctly classified images by both LDEM and KDEM (b) images that are mislabeled by LDEM, but correctly labeled by KDEM (c) images that neither LDEM or KDEM can correctly labeled.

5.3 Fingertip Tracking

In some vision-based gesture interface systems, fingers could be used as accurate pointer input devices. Also, fingertip detection and tracking play an important role in recovering hand articulations. A difficulty of the task is that fingertip motion often undergoes arbitrary rotations, which makes it hard to invariantly characterize fingertips. The proposed Kernel D-EM algorithm is employed to discriminate fingertips and non-fingertips.

We have collected 1,000 training samples including both fingertips and non-fingertips. Non-fingertip samples are collected from the background of the working space. Some training samples are shown in Figure 5. 50 samples for each two classes are manually labeled. Training images are resized to 20×20 and converted to gray-level images. Each training sample is represented by its coefficients of the 22 largest principal components. Kernel D-EM algorithm is performed on such training dataset to obtain a kernel transformation and a Bayesian classifier. Assume at time $t-1$, fingertip



(a) (b)

Figure 5: (a) Fingertip samples, (b) Non-fingertip samples.

location is X_{t-1} in image. At time t, the predicted location of fingertip is $\widetilde{X}_t$ according to Kalman prediction. For simplicity, the size of search window is fixed by 10×10 centered at $\widetilde{X}_t$. For each location in the search window, a fingertip candidate is constructed by the 20×20 sized image centered at that location. Thus, 100 candidates will be tested. A probabilistic label of such fingertip candidate is obtained by classifying it. The one with the largest probability is determined as the tracked location at time t. We run the tracking algorithm on sequences containing a large amount of fingertip rotation and complex backgrounds. The tracking result is fairly accurate.

6 Conclusion and Future Work

We presented two novel algorithms for efficient kernel-based, nonlinear, multiple discriminant analysis. These algorithms identify "kernel vectors" which are the defining training data for the purposes of classification. Benchmark tests show that KMDA with these adaptations performs comparably with the best known supervised learning algorithms. We also presented a semi-supervised discriminant analysis technique, Kernel D-EM, which employs both labeled and unlabeled data in training. On real experiments for recognizing hand postures and tracking fingertips, KDEM outperforms naïve supervised learning and existing semi-supervised algorithms.

Examination of the experimental results reveals that KMDA often maps data sets corresponding to each class into approximately Gaussian clusters in the tranformed space, even when the initial data distribution is highly non-Gaussian. In future work, we will investigate this phenomenon more closely.

Acknowledgments

This work was supported in part by National Science Foundation Grants CDA-96-24396 and IRI-96-34618 and NSF Alliance Program. The authors would like to appreciate the anonymous reviewers for their comments.

References

[1] R. Basri, D. Roth, and D. Jacobs. Clustering appearances of 3D objects. In *Proc. of IEEE Int'l Conf. Computer Vision and Pattern Recognition*, 1998.

[2] P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. In *Proc. of European Conference on Computer Vision*, April 1996.

[3] K. Bennett and A. Demiriz. Semi-supervised support vector machines. In *Proc. of Neural Information Processing Systems*, Denver, 1998.

[4] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. Conf. Computational Learning Theory*, 1998.

[5] Y. Cui and J. Weng. Hand sign recognition from intensity image sequences with complex background. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 88-93, 1996.

[6] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[7] T. Joachims. Transductive inference for text classification using support vector machines. In *Proc. of Int'l Conf. on Machine Learning*, 1999.

[8] Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, Alex Smola, and Klaus-Robert Müller. Fisher discriminant analysis with kernels. In *IEEE workshop on Neural Networks for Signal Proceesing*, 1999.

[9] Sebastian Mika, Gunnar Rätsch, Jason Weston, Bernhard Schölkopf, Alex Smola, and Klaus-Robert Müller. Invariant feature extraction and classification in kernel spaces. In *Proc. of Neural Information Processing Systems*, Denver, Nov. 1999.

[10] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 1999.

[11] Volker Roth and Volker Steinhage. Nonlinear discriminant analysis using kernel functions. In *Proc. of Neural Information Processing Systems*, Denver, Nov. 1999.

[12] Bernhard Schölkopf, Alexander Smola, and Klaus Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299-1319, 1998.

[13] V. Vapnik. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[14] M. Weber, M. Welling, and P. Perona. Towards automatic discovery of object categories. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, 2000.

[15] Ying Wu and Thomas S. Huang. View independent recognition of hand postures. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume II, pages 88-94, Hilton Head Island, South Carolina, June 2000.