Orientation Distance-based Discriminative Feature Extraction for Multi-Class Classification

Bo Liu, Yanshan Xiao
Faculty of Engineering and IT
University of Technology, Sydney
csbliu@it.uts.edu.au
syxiao@it.uts.edu.au

Longbing Cao
Faculty of Engineering and IT
University of Technology, Sydney
lbcao@it.uts.edu.au

Philip S. Yu
Department of Computer Science,
University of Illinois at Chicago
851 S. Morgan Street
Chicago, IL 60607-0753
psyu@cs.uic.edu

ABSTRACT

Feature extraction is an effective step in data mining and machine learning. While many feature extraction methods have been proposed for clustering, classification and regression, very limited work has been done on multi-class classification problems. In fact, the accuracy of multi-class classification problems relies on well-extracted features, the modeling part aside. This paper proposes a new feature extraction method, namely extracting *orientation distance-based discriminative (ODD) features*, which is particularly designed for multi-class classification problems. The proposed method works in two steps. In the first step, we extend the Fisher Discriminant idea to determine more appropriate kernel function and map the input data with all classes into a feature space. In the second step, the ODD features are extracted based on the one-vs-all scheme to generate discriminative features between a pattern and each hyperplane. These newly extracted features are treated as the representative features and are further used in the subsequent classification procedure. Substantial experiments on both UCI and real-world datasets have been conducted to investigate the performance of ODD features based multi-class classification. The statistical results show that the classification accuracy based on ODD features outperforms that of the state-of-the-art feature extraction methods.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Induction*; I.5.4 [**Pattern Recognition**]: Application

General Terms

Theory, Algorithm, Performance

Keywords

Multi-Class Classification; Feature Extraction

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

1. INTRODUCTION

Knowledge discovery or data mining is the process of finding previously unknown and potentially interesting patterns and relations from data. Feature extraction, as a critical step in data mining, is to explore distinctive features from source data, in order to lead to appropriate problem-solving performance. Feature extraction also plays an important role in exploring data, by mapping the input data onto a space, which reflects the inherent structure of the original data. To date, feature extraction has attracted increased attention due to the wide potential of its application in pattern recognition, visualization, time series analysis, etc. [1, 2, 3, 4, 5].

Depending on the principle of the optimization models, the previous feature extraction approaches can be classified into two broad categories: (1): unsupervised feature extraction [6, 7, 8, 9, 10], in which the most representative features are constructed out of the original data. In this procedure, the data label is not considered in the feature extraction procedure. For example, principal component analysis (PCA) is used to represent the source data in terms of minimal mean square error between the representation and the source data, without taking the data label into account. Consequently, PCA has little to do with the discriminative features that are optimal for classification. (2): supervised feature extraction methods [11, 12, 13, 14], which map the original data onto an ideal sub-space, where the samples from different classes are considerably separated; simultaneously they incorporate the data label information into the learning procedure, such that the extracted features can yield the difference of the distinctive features.

Despite much progress made in this area, most existing methods of feature extraction are not particularly designed for multi-class classification. For example, in unsupervised feature extraction, the label information is not incorporated into the learning procedure; consequently, data from different classes may share similar extracted features. This potentially reduces the accuracy of multi-class classification. In supervised feature extraction methods, although the data label is considered in the learning procedure, it is still difficult to choose an ideal subspace in which the projection of different classes in the source data is distinctive. For example, in the feature space of kernel Fisher discriminant analysis (KFDA) [12], KFDA determines a canonical direction for which the data is most separated when it is projected on a line in this direction. However, when the number of classes

is large, it is not easy to determine such a canonical direction so that the projections of data are well separated, even in the case that the classes are well separable in the feature space. Another important observation is that many real-world applications fall into the category of multi-class classification problems, such as pattern recognition, or question classification. Therefore, it is worthwhile to explore new feature extraction methods specifically for the multi-class classification problem, such that the extracted distinctive features can contribute maximally to classification accuracy.

This paper proposes a novel supervised feature extraction method, to extract *orientation distance-based discriminative (ODD) features* in the kernel feature space. It is particularly designed for multi-class classification problems, to enhance the classification accuracy. The main contribution of this paper can be summarized as follows.

1. We propose the use of an extension of the Fisher discriminant idea to determine a kernel function [7], so that this kernel function implicitly maps the input data with all classes into a feature space, in which each class is is individually distinguishable in the feature space.

2. We propose an ODD feature extraction method, so that each example is converted into a vector of orientation with respect to the hyper-planes (SVMs) constructed, based on the one-vs-all scheme [15]. To the best of our knowledge, this is the first work to extract distinctive features on top of the orientation distance between each sample and each hyper-plane.

3. Finally, we conduct experiments on both UCI and real-world data to compare the proposed ODD features with kernel principal component analysis (KPCA), kernel independent component analysis (KICA), and kernel Fisher discriminant features (KFD), on three multi-class classification methods: decision tree, neural network and SVM. The statistical results show that the ODD features outperform those extracted by other methods.

For clarity, the basic notions used in this paper are listed in Table 1.

The rest of the paper is organized as follows: Section 2 presents the work related to our method. Section 3 proposes our method in detail. Experiments are presented in Section 4. The conclusion and future work are discussed in Section 5.

Table 1: Basic notions and meaning

Notions	Meaning
$\mathbf{x}_i$	the i^{th} training example
R^m	input space
F	feature space
$\phi(.)$	non-linear mapping function
$\phi(\mathbf{x})$	image of $\mathbf{x}$ in kernel feature space
$K(.,.)$	kernel function
S	training set for binary classification
$\|S\|$	sample size of set S
$\mathbf{x}_1 \cdot \mathbf{x}_2$	inner product of vectors $\mathbf{x}_1$ and $\mathbf{x}_2$
$D_{\mathbf{w},b}(\mathbf{x})$	hyper-plane
$\mathbf{w}$	normal vector of hyper-plane
b	bias of hyper-plane
ξ_i	slack variable
γ	tradeoff variable
$\mathbf{m}_i$	class center of the i^{th} class
SC_i	within-class scatter of class i
d_{ij}	distance between class i and j
π	parameters set
H_i	the i^{th} hyper-plane
$d(\mathbf{x}_i, H_i)$	distance between $\mathbf{x}$ and hyper-plane H_i
$\|\mathbf{w}\|$	Euclidean norm of vector $\mathbf{w}$
S^{new}	training ODD data set
S_t	testing set for multi-class problem
S_t^{new}	testing ODD data set

2. RELATED WORK

In this Section, we review the previous works related to this paper. Because our proposed ODD feature extraction is an SVM-based technique for multi-class classification problems, we first briefly recall the principle of support vector machines in Section 2.1, and then review the previous feature reduction methods in Section 2.2.

2.1 Support Vector Machines

Support vector machines (SVMs) [15] have been proven to be a powerful classification tool in data mining and machine learning areas. Unlike classical methods that mainly minimize the empirical training error, SVM seeks an optimal separating hyper-plane that maximizes the margin between two classes after mapping the data into a feature space. SVM was originally designed for binary classification. They were extended later for multi-class classification by converting the multi-class problem into a set of binary class problems [15, 16, 17, 18, 19, 20]. Over the years, SVM has been successfully applied to many real-world applications ranging from image classification, text categorization, and face recognition to bioinformatics. The principle of SVM is briefly reviewed as follows.

Let $S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_{|S|}, y_{|S|})\}$ be a training set, where $\mathbf{x}_i \in R^m$, $y_i \in \{+1, -1\}$. In the SVM [15], a non-linear mapping function $\phi(.)$ is used to map a data set from the input space (R^m) into a feature space F, where both classes are expected to be much more linearly separable. As illustrated in Figure 1, the training problem of an SVM is to find $\mathbf{w}$ and b to achieve the optimized hyper-plane [15]:

$$D_{\mathbf{w},b}(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b = 0, \tag{1}$$

The decision function (1) satisfies the following condition

$$\begin{aligned} &y_i(\mathbf{w} \cdot \phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad i = 1, \ldots, |S|, \\ &\xi_i \geq 0, \quad for \quad i = 1, \ldots, |S|. \end{aligned} \tag{2}$$

where ξ_i is introduced to relax the margin constraints. To achieve SVM, we need to solve the following problem:

$$\min_{\mathbf{w},b,\xi} J(\mathbf{w}, b, \xi_i) = \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + \gamma \frac{1}{2} \sum_{i=1}^{|S|} \xi_i \tag{3}$$

subject to (2), where γ is a parameter which balances classifier error. By introducing the Lagrange function [15], $\mathbf{w}$, b and support vectors can be determined, and the decision classifier (Equation 1) is then obtained.

For a test instance $\mathbf{x}$, if $D_{\mathbf{w},b}(\mathbf{x}) > 0$, it is classified into the positive class; otherwise, it belongs to the negative class.

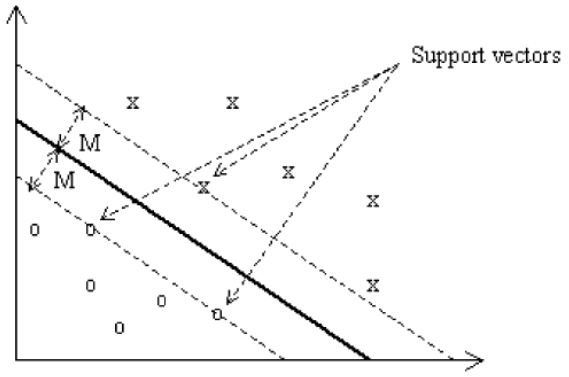


Figure 1: A graphical view of an SVM in a two-dimensional case. M is the distance from the separator to the support vectors in the feature space.

2.2 Feature Selection and Extraction

Feature selection and feature extraction attempt to reduce the dimensionality of data. Both have been widely used in machine learning and various application domains. Nonetheless, the principles of these two techniques are based on different objectives.

Feature Selection: Feature selection [21, 22, 23], also known as variable subset selection, is the technique of selecting a subset of relevant features from the original source features before doing the actual learning. Feature selection methods [24, 25, 26] are always required to find a global NP-hard optimization problem, where greedy approaches - forward selection and backward elimination - are often used to tackle the optimization problem directly.

Feature Extraction: Feature extraction aims to constructe combinations of the source variables while still describing the data with sufficient accuracy. The previous works can be broadly classified into unsupervised and supervised feature extraction categories.

In the unsupervised feature extraction methods, the data label is not considered in the learning procedure. PCA, KPCA [6, 27, 7], ICA and KICA [9, 28, 10] are best-known feature extraction algorithms. The transform of PCA is derived from eigenvectors corresponding to the largest eigenvalues of the covariance matrix for data of all classes. PCA seeks to optimally represent the data in terms of minimal mean-square-error between the representation and the original data. In recent years, some variances of PCA have been proposed. For instance, Bishop [29] adopts some prior information to obtain the intrinsic dimension and the optimal number of clusters for PCA in the latent variable model. Weng et al. [8] proposes an incremental principal component analysis to realize online learning for PCA. Tang et al. [30] uses traditional PCA and the nonorthogonal binary feature extraction method to obtain components. In addition, Kernel methods have recently been provided to implement PCA in a nonlinear fashion in the form of kernel-PCA [7]. ICA has also been proposed as a tool to find interesting projections of the data. [9, 28] maximize negentropy (divergence to a Gaussian density function) to find a subspace on which the data has the least Gaussian projection. The criterion corresponds to finding a projection of data that looks maximally clustered. This appears to be a very useful tool for revealing non-Gaussian structures in the data. KICA [10] has been proposed for non-linear separable data by using kernel functions. However, the limitations of these methods in this category is that they are completely unsupervised with regard to the class labels of the data. Consequently, they lack the ability to enhance class separability, and have little to do with the discriminative features optimal for classification.

In the supervised feature extraction, the data label is used to supervise the feature extraction procedure. In this category, Fisher discriminant analysis (FDA), also called Linear discriminant analysis (LDA), is typical [11, 13, 14]. FDA searches for the transformation by maximizing the ratio of the between-class distance to the within class distance. Unlike PCA which is not concerned with the class information, FDA takes much consideration of the label information of the data. Moreover, Kernel-based LDA (KDA) or Kernel-based FDA (KFDA) has been proposed to offer a flexible ability to handle cases where data is non-linear separable in the input space. In such case, the data is then explored in the kernel-induced feature space to find an optimal direction along which the separability of different classes is maximized. For this category, it is generally believed that FDA and KFDA improve the ability to enhance class separability compared to unsurprised feature extraction methods. However, when the number of classes for multi-class classification problems is large, it is not easy to determine optimal direction where the projections of data are well separated, even if the classes are well separable in feature space. This potentially reduces the performance of multi-class classification problems. In addition, [31] puts forward generalized and heuristic-Free feature construction to improve accuracy.

Our proposed ODD feature extraction method is proposed to improve the performance for multi-class problems. The same as FDA and KFDA, our method is a supervised feature extraction method since the label information is fully utilized in the learning procedure. The difference from KFDA is that, KFDA determines an optimal direction along which the separability of different classes is maximized in the feature space, whereas our ODD method directly constructs optimal hyper-planes (SVMs) in the feature space via one-vs-all scheme [15]. This strategy potentially guarantees the quality of ODD features compared to KFDA features because even if the classes are well separable in the feature space after kernel mapping, there may not exist an optimal direction where the projections of data are well separated.

In addition, our proposed ODD method falls into the feature extraction category, since ODD features are not the subset of source features, but the extracted features between each instance and hyper-plane. In the experiments, we will explicitly compare the ability of ODD features with those of KPCA, KICA and KFDA, since KPCA, KICA and KFDA generally outperform PCA, ICA and FDA respectively.

3. OUR PROPOSED ALGORITHM

Feature extraction aims at to construct combinations of source variables and to reduce the high dimension of input examples while still describing the data with sufficient accuracy. Due to the fact that most real-world applications fall into the category of multi-class classification and previous works have had difficulty with feature extraction for multi-class classification problems, it is worthwhile to ex-

Algorithm 1 Outline of SVM-based ODD feature extraction

1: Determine proper kernel function by the extension of Fisher discriminant-based method;
Map the source multi-class data into a feature space using the determined kernel function;
2: Construct SVM according to one-vs-all scheme in feature space;
Explore ODD feature between the instance and each hyper-plane;

plore new feature extraction techniques for multi-class classification problems to reduce the dimensions of data efficiently while maintaining accurate performance.

Because of its importance, we propose a supervised orientation distance-based discriminative (ODD) feature extraction technique which extracts distinctive features in the kernel space. In all, our proposed method works in two steps, as shown in Algorithm 1. In the first step, we propose the extension of the Fisher discriminant-based method [32] to determine a kernel function which maps input examples from the input space into a feature space in which each class is found to be distinguishable from others. In the second step, we construct SVMs according to the one-vs-all scheme to separate the classes into distinctive domains and extract the orientation distance-based discriminative (ODD) features by calculating the orientation distance between each sample and each hyper-plane. In the following, we exhibit the two steps in detail.

3.1 Selection of Kernel Function

In the kernel method, the input data is mapped from the input space (R^m) into another higher-dimensional feature space (F) via a non-linear mapping function ($\phi(.)$) or kernel function ($K(.,.)$). In the feature space, the classes are expected to be much more linearly separable from each other [33]. In addition, the inner products of the two vectors in the feature space can be obtained efficiently and directly from the original data items using a kernel function. RBF as shown in (4) is a typical kernel function in many real-world applications.

$$K(\mathbf{x}, \mathbf{x}_i) = \exp(-\| \mathbf{x} - \mathbf{x}_i \|_2 / 2\sigma^2) \tag{4}$$

Let us take a toy problem for example as illustrated in Figure 2, two classes are non-linearly separable in the input space (left panel (A)), but are linearly separably from each other after mapping them into a three-dimensional feature space via nonlinear mapping. Due to the flexible generation of the kernel method, it has demonstrated its power to enhance the performance of many machine learning tools such as the support vector machine (SVM) [15], the kernel principal component analysis (KPCA) [34] and support vector data description [35].

In the first step of our proposed feature extraction method, the kernel method is adopted to embed the source multi-class classification data into a feature space where each class is expected to be distinguishable from the other. As for the choice of kernel function, most previous works have adopted the strategy of employing a type of kernel function first and then determining the proper parameters for the selected type of kernel function [15]. In the first step of our method, as with the previous work, we first employ a specific type of

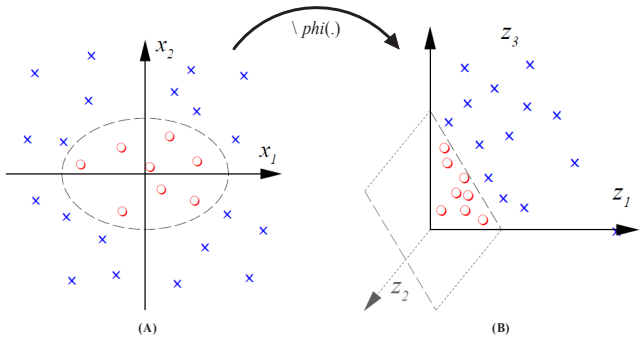


Figure 2: Illustration of kernel mapping for two-class example. (A): Both classes are non-linearly separable in the input space (B): In the feature space, both classes are linearly separable after non-linear mapping $\phi(.) = (z_1, z_2, z_3) = (x_1^2, x_2^2, \sqrt{2}x_1 \cdot x_2)$ or kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j)$.

kernel function and then propose the approach to determine proper kernel parameters which maximizes the *between classes distance* of the classes, while minimizing the *within classes scatter* of the classes. For example, if RBF kernel function as in (4) is selected as the preferred function, we then need to determine kernel parameter σ. First of all, several notions are defined for a training set S with C classes.

Definition 1. Assume the sample size of Class i is l_i, and $\mathbf{x}_{ij} \in$ *Class* i, the *class center* (m_i) and the *within-class scatter* of Class i (SC_i) in the feature space are defined as

$$m_i = \frac{1}{l_i} \sum_{j=1}^{l_i} \phi(\mathbf{x}_{ij}), \tag{5}$$

$$\begin{aligned} SC_i^2 &= \frac{1}{l_i^2} \sum_{j=1}^{l_i} \|\phi(\mathbf{x}_{ij}) - m_i\|^2 \\ &= \sum_{j=1}^{l_i} K(\mathbf{x}_{ij}, \mathbf{x}_{ij}) - \frac{1}{l_i} \sum_{j=1}^{l_i} \sum_{k=1}^{l_i} K(\mathbf{x}_{ij}, \mathbf{x}_{ik}). \end{aligned} \tag{6}$$

In definition 1, the class center of each class is implicitly defined, SC_i is denoted as the average distance between each sample and its class center which can be explicitly calculated via kernel function without knowing the actual formulation of the non-linear mapping function. In general, the smaller value is S_i, the compacter is class i and more similar are instances of class i.

Definition 2. The *class-distance* between Classes i and j is denoted as d_{ij}:

$$\begin{aligned} d_{ij}^2 &= \|m_i - m_j\|^2 \\ &= \frac{1}{l_i^2} \sum_{k=1}^{l_i} \sum_{n=1}^{l_i} K(\mathbf{x}_{ik}, \mathbf{x}_{in}) + \frac{1}{l_j^2} \sum_{k=1}^{l_j} \sum_{n=1}^{l_j} K(\mathbf{x}_{jk}, \mathbf{x}_{jn}) \\ &- \frac{2}{l_i l_j} \sum_{k=1}^{l_i} \sum_{n=1}^{l_j} K(\mathbf{x}_{ik}, \mathbf{x}_{jn}) \end{aligned} \tag{7}$$

In definition 2, the distance between classes i and j is defined as the distance between their class centers. As with SC_i, the actual value can be explicitly calculated by kernel function.

In general, the smaller the value of d_{ij}, the more two classes are likely to overlap.

Definition 3. The *discriminant function* $J_F(\pi)$ is

$$J_F(\pi) = \frac{\sum_{i=1}^{C-1}\sum_{j=i+1}^{C} d_{ij}^2}{\sum_{i=1}^{C} SC_i^2}, \quad (8)$$

where π is the parameter set in a selected type of kernel function. For example, in polynomial kernel function $K(\mathbf{x}, \mathbf{x}_i) = (1 + \mathbf{x} \cdot \mathbf{x}_i)^d$, the parameter is d, and in RBF function $K(\mathbf{x}, \mathbf{x}_i) = \exp(-\| \mathbf{x} - \mathbf{x}_i \|_2/2\sigma^2)$, the parameter is σ.

In definition 3, from the intuitive understanding and the essence of the Fisher discriminant idea, the kernel parameter can be chosen by maximizing $J_F(\pi)$ such that the classes are found distinguishable from each other. This is because, the larger $J_F(\pi)$ is, the larger the distance between each pair of classes (d_{ij}), the compacter is each class (S_i).

In order to maximize $J_F(\pi)$, two typical optimization methods, i.e., the gradient descent method [15] and grid search [36] can be used here. In the former, the initial value of the kernel parameter is set as π_0, and the step moved to where the derivative $\frac{dJ_F(\pi)}{d\pi}$ has maximum value. However, this method always exhibits a local optimal solution. In the latter method, the kernel parameters are chosen in the parameters set to maximize $J_F(\pi)$. More specifically, assume the parameter set π has $|\pi|$ parameters as follows:

$$\pi = \{\pi_1, \pi_2, \ldots, \pi_{|\pi|}\},$$

Suppose π_i has pre-specified n_i choices

$$\{\pi_{i1}, \pi_{i2}, \ldots, \pi_{in_i}\},$$

there are in total $n_{Total} = n_1 * n_2 * \ldots * n_{|\pi|}$ choices. One can compute (8) on each choice in π to explore the parameters with maximum value.

Because the grid search method has been widely used in SVM to tune hyper-parameters [36] and our ODD feature extraction is an SVM-based technique, we employ the grid search method to explore proper kernel parameters. The pseudo code of determination of the kernel parameters algorithm is outlined in Algorithm 2. After this, the parameters in the selected kernel function are determined, which means the kernel function $K_1(.,.)$ is confirmed.

Algorithm 2 Determination of kernel parameters.

Input: Training set S // with C classes
a specific type of kernel function. // K(.,.)
π // parameter set $\pi = \{\pi_1, \pi_2, \ldots, \pi_{|\pi|}\}$
preset choices for each π_i // each π_i has n_i choices
Output: kernel parameters

```
Set set π_return to store parameters;
Set value = 0 as a temporal variable;
for (each set of choice π^i in π ) do
    Calculate the class center and within-class scatter of
    each class according to (5) and (6);
    Compute the class-distance between each pair of
    classes according to (7);
    Calculate the discriminant function according to (8);
    if J_F(π^i) > value then
        value = J_F(π^i);
        π_return = π^i;
    end if
end for
Return π_return.
```

3.2 ODD Feature Extraction

After the determination of kernel function $K_1(.,.)$, the classes of the dataset are implicitly mapped into a feature space where the inner product of two vectors in the feature space can be explicitly calculated by $K_1(.,.)$. In the second step of our ODD feature extraction method, we first construct hyper-planes (SVMs) according to the one-vs-all scheme [15], i.e., considering one class as positive class and remaining classes as negative class, and then extract the orientation distance-based discriminative (ODD) features between instances and each hyper-plane in the feature space to represent the source data instead of the source input features. The second step of our proposed ODD feature extraction approach is detailed as follows.

3.2.1 Hyper-plane Construction

First, we transfer a multi-class classification into binary problems based on the one-vs-all scheme. More specifically, for a c-class problem, we take class i as positive class and the remaining classes as negative class to compose C binary classification problems. Suppose S_i is the new formed training set for the i^{th} decomposed binary classification problem. The optimal hyper-plane for the i^{th} binary classification problem is trained as follows.

$$\begin{aligned} & D_i(\mathbf{x}) = \mathbf{w}_i \cdot \phi(\mathbf{x}) + b_i = 0 \\ s.t \quad & y_{ij}(\mathbf{w}_i \cdot \phi(\mathbf{x}_{ij}) + b) \geq 1 - \xi_{ij}, \quad j = 1, \ldots, |S_i|, \\ & \xi_{ij} \geq 0, \quad for \quad j = 1, \ldots, |S_i|. \end{aligned} \quad (9)$$

where $\mathbf{w}_i$ is the normal vector for the i^{th} hyper-plane, $\phi(\mathbf{x})$ is a mapping function corresponding to $K_1(.,.)$, b_i is a bias for the i^{th} hyper-plane.

By introducing the Lagrange function [15], $\mathbf{w}_i$ and b_i can be determined, and hyper-planes $D_i(\mathbf{x}) = 0$ are then obtained. After this step, we obtain C hyper-planes in the feature space, i.e., $D_i(\mathbf{x}) = 0$ ($i = 1, 2, \ldots, C$) which portion the feature space.

3.2.2 Feature Extraction

Second, for each instance in the training set S, we can calculate the orientation distances between the sample $\mathbf{x}_i$ and each hyper-plane. Firstly, we have the following Theorem.

Theorem 1. For a hyper-plane $D_i(\mathbf{x}) = \mathbf{w}_i \cdot \phi(\mathbf{x}) + b_i = 0$ in the feature space, the distance between an instance $\phi(\mathbf{x}_k)$ and $D_i(\mathbf{x}) = 0$ is calculated as follows.

$$d(\phi(\mathbf{x}_k), H_i) = \frac{D_i(\mathbf{x}_k)}{\|\mathbf{w}_i\|}, \ for \ i = 1, 2, \cdots, C. \quad (10)$$

where $\|\mathbf{w}_i\| = \sqrt{(\mathbf{w}_i \cdot \mathbf{w}_i)}$ represents the Euclidean norm, H_i denotes the i^{th} hyper-plane.

Proof: The formulation of SVM: $D_i(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b_i = 0$ is a hyper-plane in the feature space.

As illustrated in Figure 3, assume one instance $\mathbf{o}$ resides on the surface of hyper-plane, that is

$$D_i(\mathbf{o}) = \mathbf{w} \cdot \phi(\mathbf{o}) + b_i = 0 \quad (11)$$

Then the distance of instance $\mathbf{p}$ and $D_i(\mathbf{x}) = 0$ can be cal-

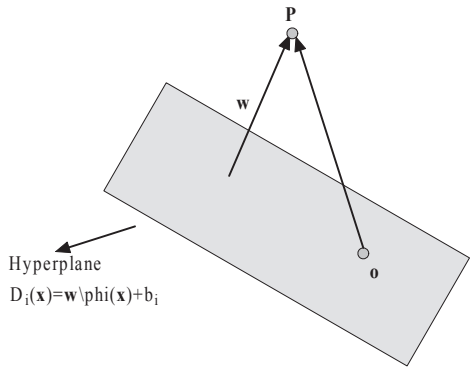


Figure 3: An illustration of calculating the distance between an instance p and hyper-plane $D_i(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b_i = 0$ in feature space.

culated as follows.

$$d(p, H_i) = \frac{\overrightarrow{OP} \cdot \mathbf{w}}{\|\mathbf{w}\|} = \frac{(\mathbf{p} - \mathbf{o}) \cdot \mathbf{w}}{\|\mathbf{w}\|} = \frac{\mathbf{p} \cdot \mathbf{w}}{\|\mathbf{w}\|} - \frac{\mathbf{o} \cdot \mathbf{w}}{\|\mathbf{w}\|} \quad (12)$$

According to Equation (11), we have

$$d(p, H_i) = \frac{\mathbf{p} \cdot \mathbf{w}}{\|\mathbf{w}\|} + b_i. \quad (13)$$

End of Proof

After calculating the orientation distance between each sample $\mathbf{x}_k$ and each hyper-plane, we then obtain the new C dimensional vector in the feature space, i.e.,

$$\frac{D_1(\mathbf{x}_k)}{\|\mathbf{w}_1\|}, \frac{D_2(\mathbf{x}_k)}{\|\mathbf{w}_2\|}, \ldots, \frac{D_C(\mathbf{x}_k)}{\|\mathbf{w}_C\|}$$

It is noted that the value of the orientation distance can be either positive or negative, which depends on the position of the sample and corresponding hyper-plane.

Let us consider the three-class problem as shown in Figure 4, where each arrowhead denotes the orientation of each hyper-plane, $D_1(\mathbf{x}) = 0$, $D_2(\mathbf{x}) = 0$, $D_3(\mathbf{x}) = 0$ represents the hyper-planes. An input vector is denoted as $(\mathbf{x}, 1)$, where 1 represents the label of sample $\mathbf{x}$. After we construct three hyper-planes based on the one-vs-all scheme and calculate the orientation distances between $\mathbf{x}$ and each hyper-plane, $(\mathbf{x}, 1)$ is transformed into $(\mathbf{x}^{new}, 1)$:

$$(\mathbf{x}, 1) \longrightarrow (\mathbf{x}^{new}, 1) = ((\frac{D_1(\mathbf{x})}{\|\mathbf{w}_1\|}, \frac{D_2(\mathbf{x})}{\|\mathbf{w}_2\|}, \frac{D_3(\mathbf{x})}{\|\mathbf{w}_3\|}), 1). \quad (14)$$

Considering the orientation of each hyper-plane, we have

$$\frac{D_1(\mathbf{x})}{\|\mathbf{w}_1\|} > 0, \frac{D_2(\mathbf{x})}{\|\mathbf{w}_2\|} < 0, \frac{D_3(\mathbf{x})}{\|\mathbf{w}_3\|} < 0.$$

The new training set S^{new} can be transformed from the source training set S as follows.

$$S = \begin{pmatrix} \mathbf{x}_1 & y_1 \\ \vdots & \vdots \\ \mathbf{x}_{|S|} & y_{|S|} \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1m} & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ x_{|S|1} & \cdots & x_{|S|m} & y_{|S|} \end{pmatrix}_{|S|*(m+1)}$$

Algorithm 3 ODD features extraction.

Input: Training set S; // with C classes
Testing set S_t;
kernel function $K_1(.,.)$; // obtained from step one
Output: S^{new} and S_t^{new}

```
Decompose C-class problem into C binary classification problem according to one-vs-all scheme;
for (j=1; j++;j≤ C) do
    Training SVM for the i^th binary classification problem;
    Obtain decision function D_i(x) = 0 according to optimization problem (9);
end for
for each instance x_i in S do
    Calculate new ODD features according to (10);
    Put ODD features and the source label into S^new;
end for
Obtain S^new in (15);
for each instance x_i in S_t do
    Calculate new ODD features according to (10);
    Put ODD features into S_t^new;
end for
Obtain S_t^new in (17);
Return S^new and S_t^new.
```

$$\longrightarrow \begin{pmatrix} \frac{D_1(\mathbf{x}_1)}{\|\mathbf{w}_1\|} & \cdots & \frac{D_C(\mathbf{x}_1)}{\|\mathbf{w}_C\|} & y_1 \\ \vdots & \ddots & \vdots & \vdots \\ \frac{D_1(\mathbf{x}_{|S|})}{\|\mathbf{w}_1\|} & \cdots & \frac{D_C(\mathbf{x}_{|S|})}{\|\mathbf{w}_C\|} & y_{|S|} \end{pmatrix}_{|S|*(C+1)} = S^{new}. \quad (15)$$

In this way, the original training sample $\mathbf{x}_k$ is represented by using a $C-$ dimensional distance vector in (16)

$$\mathbf{x}_k \rightarrow \mathbf{x}_k^{new} = (\frac{D_1(\mathbf{x}_k)}{\|\mathbf{w}_1\|}, \frac{D_2(\mathbf{x}_k)}{\|\mathbf{w}_2\|}, \cdots, \frac{D_{C-1}(\mathbf{x}_k)}{\|\mathbf{w}_{C-1}\|}, \frac{D_C(\mathbf{x}_k)}{\|\mathbf{w}_C\|}). \quad (16)$$

For the testing set $S_t = \{\mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_n^t\}$, $\mathbf{x}_i^t \in R^m$, the orientation distances between a sample and each hyper-plane is also computed, and S_t is transformed into S_t^{new} as follows.

$$\begin{pmatrix} \mathbf{x}_1^t \\ \vdots \\ \mathbf{x}_{|S_t|}^t \end{pmatrix} = \begin{pmatrix} x_{11}^t & \cdots & x_{1m}^t \\ \vdots & \ddots & \vdots \\ x_{|S_t|1}^t & \cdots & x_{|S_t|m}^t \end{pmatrix}_{|S_t|*m}$$

$$\longrightarrow S_t^{new} = \begin{pmatrix} \frac{D_1(\mathbf{x}_1^t)}{\|\mathbf{w}_1\|} & \cdots & \frac{D_C(\mathbf{x}_1^t)}{\|\mathbf{w}_C\|} & \\ \vdots & \ddots & \vdots & \vdots \\ \frac{D_1(\mathbf{x}_{|S_t|}^t)}{\|\mathbf{w}_1\|} & \cdots & \frac{D_C(\mathbf{x}_{|S_t|}^t)}{\|\mathbf{w}_C\|} & \end{pmatrix}_{|S_t|*C}. \quad (17)$$

In all, the procedure of ODD feature extraction is outlined in Algorithm 3.

3.3 Computational Complexity Analysis

The computational complexity of ODD feature extraction is analyzed as follows. Binary SVM generally suffers from an $O(n)^\lambda$ training cost where n represents the training sample size and $\lambda < 2$. For a $C-$class problem, assume each class equally has l/C samples, the computational complexity of our ODD feature extraction according to one-vs-all scheme

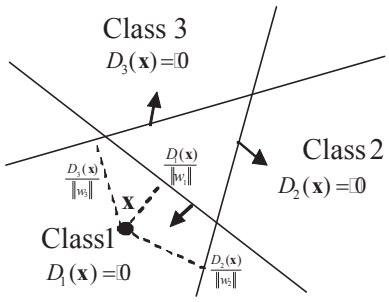


Figure 4: Orientation distance discriminant (ODD) feature between sample x and each hyper-plane in terms of one-vs-all scheme.

is

$$T_{ODD} = C \cdot O(l)^{\lambda} \quad \lambda < 2. \tag{18}$$

Compared to other feature extraction methods, such as KPCA, KICA and KFDA, the computational complexity of ODD is acceptable.

4. EXPERIMENTS

In order to evaluate the performance of our proposed ODD features, we conduct experiments on both UCI datasets and a real-world dataset. For comparison, three classical feature extraction techniques, including unsupervised and supervised methods, are used here as baselines. The first is called kernel principal component analysis (KPCA). The second is referred to as kernel independent component analysis (KICA). These two methods belong to unsupervised techniques without taking data label into the learning phase. The third one is kernel linear discriminant analysis (LDA or FDA), which incorporates label information into the learning process.

After feature extraction by the above four methods, we employ three famous multi-class classification methods, i.e., decision three, neural network, SVMs with one-vs-one scheme, to observe the functionalities of the ODD features, KPCA features, KICA features and FDA features. In the experiments, RBF kernel function in (4) is used in the four kernel-based feature extraction methods.

All the experiments are conducted in Matlab environment, the test platform is a Dual 2.8GHz Intel Core2 T9600 PC with 3.45GB RAM.

4.1 Evaluation Metrics

The performance of classification systems is typically evaluated in terms of F-measure [37, 38]. For the i^{th} binary classification problem shown in Table 2, F-measure trades off precision p_i and recall r_i which are defined as follows.

$$P_i = \frac{TP_i}{TP_i + FP_i} \qquad R_i = \frac{TP_i}{TP_i + FN_i} \tag{19}$$

For the C-class classification problem, the macro-average precision and recall of the category space is obtained from the overall number of instances correctly accepted and wrongly accepted

$$P_{av} = \frac{\sum_{i=1}^{C} P_i}{C} \qquad R_{av} = \frac{\sum_{i=1}^{C} R_i}{C} \tag{20}$$

Table 2: Confusion Matrix for i^{th} binary classification problem

		Actual Label	
		Positive	Negative
Predicted Label	Positive	True Positive (TP_i)	False Negative (FN_i)
	Negative	False Positive (FP_i)	True Negative (TN_i)

Table 6: UCI datasets description.

Dataset	# of samples	# of Class	# of Features
Dermatology	366	6	34
Soybean	683	19	35
Vowel	990	10	11
Vehicle	846	4	18
Isolet	7797	26	617
Thyroid	7200	3	21
Pageblock	5473	5	10
Pendigits	10992	10	16

The F-measure is defined as the harmonic mean of precision P_{av} and recall R_{av}:

$$F = \frac{2P_{av}R_{av}}{P_{av} + R_{av}} \tag{21}$$

From this definition, it is clear that the F-measure reacts to an average effect of both precision and recall. When either of them (P_{av} or R_{av}) is small, the value will be small. Only when both are large will the F-measure exhibit large value. For the theoretical bases and practical advantages of F-measure, please refer to [38] for detail.

4.2 Datasets

The datasets used in our experiments consist of eight UCI datasets and one real-world question classification dataset. The detailed information about these eight UCI datasets is illustrated in Table 6. For the real-world question classification dataset, it contains very original text questions, consists of 1264 questions with 6 coarse categories ("location", "Time", "Human", "Object", "Description" and "Number"). We extract part-of-speech (POS) features for each question.

At the pre-processing stage, all records in each dataset are normalized to [-1, 1]. The parameter σ in (4) is searched in

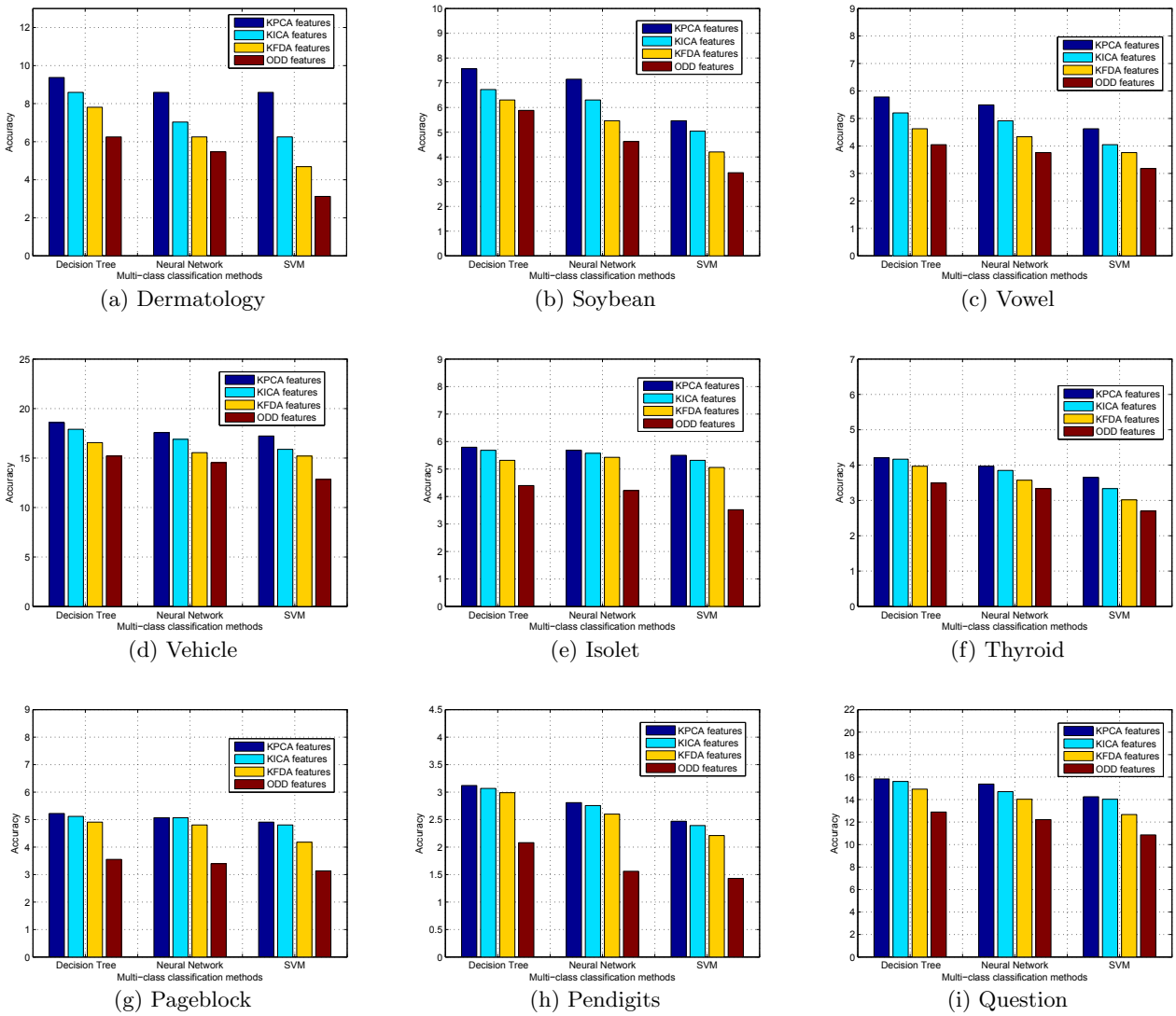
$$\{\sigma_0/8, \sigma_0/4, \sigma_0/2, \sigma_0, 2\sigma_0, 4\sigma_0, 8\sigma_0\}, \tag{22}$$

where σ_0 is the mean norm of the training data.

4.3 Experiment Settings and Results

Experiments are established to compare the performance of KPCA, KICA, KFDA and ODD features. To avoid sampling bias, we generate 10 groups of training data and testing data for each dataset by randomly selecting 65% data as a training set and the remainder as a testing set at each time.

For each group of training and testing sets, the experiments consist of two steps. In the first step, we extract KPCA, KICA, KFDA, ODD features from the source data. In the second step, we perform decision three (C4.5), neural network (BP), and SVM-based one-vs-one methods on the extracted features to compare the quality of these extracted features. Specifically, in the first step, for ODD feature ex-



(a) Dermatology (b) Soybean (c) Vowel

(d) Vehicle (e) Isolet (f) Thyroid

(g) Pageblock (h) Pendigits (i) Question

Figure 5: The performance comparison of KPCA, KICA, KFDA and ODD features using decision tree, neural network and SVM-based one-vs-one method

traction, we perform the extension of the Fisher discriminant method on each choice of (22) to identify the parameter σ in (4) such that the discriminant function J_F in (8) achieves its maximum value. After that, one-vs-all scheme is conducted to extract ODD features between each instance and hyperplanes. In this procedure, another parameter γ in SVM is set 1 when constructing binary SVM, since the dataset is normalized and this setting is acceptable in tuning parameters [39]. With respect to KPCA and KICA feature extraction, we keep the extracted features on each choice of (22) because we do not know which σ in (22) leads to a better performance of multi-class classification algorithms on them. Since there exist seven choices for σ in (22), we obtain seven sets of KPCA features, KICA features and KFDA features respectively.

In the second step, we directly conduct the three multi-class classification algorithms on ODD features and obtain the performance. For the seven sets of KPCA features, we conduct the multi-class classification algorithm on each. We then obtain the seven highest testing accuracies and retain the most high to represent the accuracy on the KPCA feature. The same conduction is performed on the seven sets of KICA and report the maximum performance.

Finally, the testing errors on KPCA, KICA, KFDA and ODD features are shown in Figure 4 using three multi-class classification methods. The results show that, for each multi-class classification algorithm, the testing error on ODD features is always smaller than that on KPCA and KICA features. From the experiments, the results indicate that ODD features can improve the accuracy of multi-class classification algorithms. We report the average testing error and the standard deviations for KPCA, KICA, KFDA and ODD features of ten groups of training and testing sets from Table 3 to 5. It is clear that the average testing error on ODD fea-

Table 3: The testing error and standard deviation on ODD, KPCA, KICA and KFDA features using decision tree method.

Data set	KPCA	KICA	KDA	ODD
Dermatology	9.321 ± 5.436	8.872 ± 5.234	7.643 ± 4.653	6.353 ± 4.73
Soybean	7.421 ± 8.923	6.821 ± 7.943	6.353 ± 7.982	5.342 ± 6.382
Vowel	5.983 ± 8.492	5.432 ± 8.429	4.782 ± 7.948	4.082 ± 7.843
Vehicle	17.834 ± 10.489	17.832 ± 9.742	16.893 ± 9.843	15.732 ± 9.742
Isolet	5.832 ± 4.293	5.528 ± 4.623	5.423 ± 4.193	4.345 ± 3.982
Thyroid	4.29 ± 5.932	4.153 ± 5.273	3.982 ± 5.103	3.682 ± 4.924
Pageblock	5.122 ± 4.932	5.113 ± 4.621	4.982 ± 4.371	3.645 ± 3.984
Pendigits	3.101 ± 3.293	3.067 ± 3.115	2.983 ± 2.942	2.098 ± 2.832
Question	15.923 ± 10.293	15.632 ± 10.15	14.583 ± 9.843	12.234 ± 9.234

Table 4: The testing error and standard deviation on ODD, KPCA, KICA and KFDA features using neural network method.

Data set	KPCA	KICA	KDA	ODD
Dermatology	8.654 ± 5.643	7.962 ± 5.784	6.142 ± 5.423	5.574 ± 5.532
Soybean	6.982 ± 7.49	6.873 ± 7.832	5.329 ± 7.535	4.213 ± 7.632
Vowel	5.582 ± 7.493	5.021 ± 7.783	4.562 ± 7.535	3.892 ± 7.423
Vehicle	17.593 ± 10.943	17.213 ± 9.492	16.092 ± 8.942	14.729 ± 9.021
Isolet	5.572 ± 4.823	5.823 ± 4.424	5.532 ± 4.213	4.353 ± 3.802
Thyroid	3.892 ± 5.824	3.681 ± 5.764	3.414 ± 5.325	3.213 ± 4.974
Pageblock	5.061 ± 4.724	5.061 ± 4.693	4.802 ± 4.429	3.395 ± 3.894
Pendigits	2.804 ± 3.463	2.743 ± 3.652	2.538 ± 3.564	1.632 ± 3.134
Question	14.932 ± 11.293	14.394 ± 9.492	13.394 ± 8.492	11.823 ± 8.134

Table 5: The testing error and standard deviation on ODD, KPCA, KICA and KFDA features using SVM-based one-vs-one method.

Data set	KPCA	KICA	KDA	ODD
Dermatology	8.632 ± 5.923	6.982 ± 5.732	5.492 ± 5.632	3.821 ± 6.553
Soybean	5.573 ± 7.932	5.093 ± 7.832	4.432 ± 7.643	3.672 ± 6.923
Vowel	4.532 ± 7.942	4.032 ± 6.384	3.7213 ± 6.463	3.021 ± 6.146
Vehicle	17.392 ± 9.842	16.023 ± 10.942	15.353 ± 9.843	12.837 ± 9.632
Isolet	5.562 ± 3.942	5.424 ± 3.742	5.098 ± 3.998	3.423 ± 3.623
Thyroid	3.762 ± 5.254	3.423 ± 5.034	3.081 ± 4.824	2.587 ± 4.723
Pageblock	4.982 ± 4.294	4.874 ± 4.183	4.163 ± 3.982	3.154 ± 3.872
Pendigits	2.478 ± 3.742	2.436 ± 3.253	2.295 ± 3.155	1.432 ± 3.345
Question	14.953 ± 9.392	14.234 ± 8.943	12.834 ± 8.342	10.893 ± 6.435

tures is always smaller than that of KPCA, KICA, KFDA and the standard deviation of ODD features is also comparable with others.

5. CONCLUSIONS AND FUTURE WORK

While many feature extraction methods have been proposed, they are often not suitable for identifying discriminative features for multi-class classification, and potentially result in low classification accuracy. This paper has proposed a novel feature extraction method, to extract orientation distance-based discriminative (ODD) features, specifically designed for multi-class classification problems. The proposed method works well in two steps. In the first step, the kernel function is determined by extending the Fisher discriminant idea. In the second step, ODD features are then extracted based on the one-vs-all scheme to generate discriminative features. Substantial experiments on eight UCI datasets and a real-world dataset have shown that our proposed ODD features method outperforms state-of-the-art feature extraction methods, including KPCA, KICA and KFDA.

We are extending our work in several directions. We would like to apply the ODD feature extraction method to the online environment, and apart from the question classification, we are trying to apply the proposed method to bioinformatics with multi-class classification data.

6. ACKNOWLEDGMENT

This work is sponsored in part by QCIS (Quantum Computation and Intelligent Systems), CMCRC (Capital Markets CRC Limited), Australian Research Council Discovery Grants (DP1096218, DP0988016, DP0773412) and ARC Linkage Grant (LP0989721, LP0775041), as well as US NSF through grants IIS-0905215, DBI-0960443 and OISE-0968341.

7. REFERENCES

[1] J. Zhang and L. Gruenwald. Opening the black box of feature extraction: incorporating visualization into high-dimensional data mining processes. In *ICDM*, pages 1550–4786. IEEE, 2006.

[2] W. Zuo, D. Zhang, J. Yang, and K. Wang. Bdpca plus lda: a novel fast feature extraction technique for face recognition. *IEEE TSMC: Part B: Cybernetics*, 36(4):946–953, 2006.

[3] Y. Liu, L. V. Lita, R. S. Niculescu, K. Bai, P. Mitra, and C. L. Giles. Real-time data pre-processing technique for efficient feature extraction in large scale datasets. In *CIKM '08*, pages 981–990. ACM, 2008.

[4] T. Sun, S. Chen, J. Yang, and P. Shi. A novel method of combined feature extraction for recognition. In *ICDM*, pages 1550–4786. IEEE, 2008.

[5] Y. Xu, S. Furao, J. Zhao, and O. Hasegawa. To obtain

orthogonal feature extraction using training data selection. In *CIKM 2009*, pages 1819–1822. ACM, November 2009.

[6] P. A. Devijver and J. Kittler. Pattern recognition: A statistical approach. *Prentice Hall, London*, 1982.

[7] B. Schölkopf, S. Mika, C. Burges, P. Knirsch, K. Müller, G. Rütsch, and A. Smola. Input space vs. feature space in kernel-based methods. *IEEE TNN*, 10:1000–1017, 1999.

[8] J. Weng, Y. Zhang, and W. S. Hwang. Candid covariance-free incremental principal component analysis. *IEEE TPAMI*, 25(8):1034–1040, 2003.

[9] J. Chien and B. C. Chen. A new independent component analysis for speech recognition and separation. *IEEE TPAMI*, 14(4):1245–1254, 2006.

[10] B. Xu, X. Jin, P. Guo, and F. Bie. Kica feature extraction in application to fnn based image registration. In *IJCNN*, pages 3602–3608, 2006.

[11] M. Girolami, A. Cichocki, and S. I. Amari. A common neural network model for unsupervised exploratory data analysis and independent component analysis. *IEEE TNN*, 9(6):1495–1501, November 1998.

[12] S. Mika, G. Rätsch, J. Weston, B. Schoölkopf, and K. R. Müller. Fisher discriminant analysis with kernels. *Neural Networks for Signal Processing*, pages 41–48, 1999.

[13] H. Zhao, S. Sun, Z. Jing, and J. Yang. Local structure based supervised feature extraction. *Pattern Recognition*, 39:1546–1550, 2006.

[14] J. Yang, A. F. Frangi, J. Y. Yang, D. Zhang, and Z. Jin. Kpca plus lda: A complete kernel fisher discriminant framework for feature extraction and recognition. *IEEE TPAMI*, 27(2):230–244, 2005.

[15] V. Vapnik. *Statistical learning theory*. Springer, 1998.

[16] E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *JMLR*, 1:113–141, 2000.

[17] B. Liu, Z. Hao, and E. Tsang. Nesting one-against-one algorithm based on svms for pattern classification. *IEEE TNN*, 19:2044–2052, 2008.

[18] B. Liu, L. Cao, P. S. Yu, and C. Zhang. Multi-space-mapped svms for multi-class classifications. *ICDM 2008*, pages 911–916, 2008.

[19] A. Passerini, M. Pontil, and P. Frasconi. New results on error correcting output codes of kernel machines. *IEEE TNN*, 15:45–54, 2004.

[20] R. Tibshirani and T. Hastie. Margin trees for high-dimensional classification. *JMLR*, 3:637–652, 2007.

[21] J. Ren, Z. Qiu, W. Fan, H. Cheng, and P. S. Yu. Forward semi-supervised feature selection. In *PAKDD*, pages 970–976, 2008.

[22] K. Shima, M. Todoriki, and A. Suzuki. Svm-based feature selection of latent semantic features. *Pattern Recognition Letters*, pages 1051–1057, 2004.

[23] F. Pan, T. Converse, D. Ahn, F. Salvetti, and G. Donato. Feature selection for ranking using boosted trees. In *CIKM 2009*, pages 2025–2028. ACM, November 2009.

[24] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of zero-norm with linear models and kernel method. *JMLR*, 3:1439–1461, 2003.

[25] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[26] L. Song, A. Smola, A. Gretton, K. Borgwardt, and J. Bedo. Supervised feature selection via dependence estimation. In *ICML*, pages 823–830, 2007.

[27] F. Zhang. A polygonal line algorithm based nonlinear feature extraction method. In *ICDM*, pages 281–288. IEEE, 2004.

[28] I. Dagher and R. Nachar. Face recognition using ipca-ica algorithm. *IEEE TPAMI*, 28(6):996–1000, 2006.

[29] C. Bishop. Bayesian pca. In *NIPS 1999*, pages 382–338, 1999.

[30] F. Tang, R. Crabb, and H. Tao. Representing images using nonorthogonal haar-like bases. *IEEE TPAMI*, 29(12):2120–2134, 2007.

[31] W. Fan, E. Zhong, J. Peng, O. Verscheure, K. Zhang, J. Ren, R. Yan, and Q. Yang. Generalized and heuristic-free feature construction for improved accuracy. *SDM*, pages 629–640, 2010.

[32] V. Roth and V. Steinhage. Nonlinear discriminant analysis using kernel function. *Advances in Neural Information Processing Systems*, pages 568–574, 2000.

[33] B. Schölkopf and A. Smola. *Learning with kernels*. Cambridge, MA: MIT Press, 2001.

[34] S. Mika, B. Schölkopf, A. J. Smola, K.R. Miller, M. Scholz, and G. Rütsch. Kernel pca and de-noising in feature spaces. *in Advances in Neural Information Processing Systems, (Eds.) Cambridge, MA: MIT Press*, pages 536–542, June 1999.

[35] D. Tax and R. Duin. Support vector data description. *Machine Learning*, 54:45–66, 2004.

[36] C. Hsu. A comparison of methods for multiclass support vector machines. *IEEE TNN*, 13:415–425, 2002.

[37] S. Mika, G. Rätsch, J. Weston, B. Schoölkopf, and K. R. Müller. Fuzzy support vector machine for multi-class text categorization. *Information Processing and Management*, 43(4):914–929, 2007.

[38] J. William and M. Shaw. *On the foundation of evaluation*. American society for information science, 1986.

[39] S. Keerthi. Efficient tuning of svm hyperparameters using radius/margin bound and iterative algorithms. *IEEE TNN*, 5:1225–1229, 2002.