A Novel Kernelized Classifier Based on the Combination of Partially Global and Local Characteristics

Riadh Ksantini[2] and Raouf Gharbi[1]

[1] Department of Computer Networks
Université Internationale de Tunis, Tunis, 2035 Tunisia
Tel.: +216 71 809 000
gharbiraouf@outlook.com
[2] University of Windsor, Windsor, ON N9B 3P4 Canada.
SUP'COM. Research Unit: Sécurité Numérique. Tunisia.
ksantini@uwindsor.ca

Abstract. The Kernel Support Vector Machine (KSVM) has achieved promising classification performance. However, since it is based only on local information (Support Vectors), it is sensitive to directions with large data spread. On the other hand, Kernel Nonparametric Discriminant Analysis (KNDA) is an improvement over the more general Kernel Fisher Discriminant Analysis (KFD), where the normality assumption from KFD is relaxed. Furthermore, KNDA incorporates the partially global information in the Kernel space, to detect the dominant normal directions to the decision surface, which represent the true data spread. However, KNDA relies on the choice of the κ-nearest neighbors ($\kappa - NN$'s) on the decision boundary. This paper introduces a novel Combined KSVM and KNDA (CKSVMNDA) model which controls the spread of the data, while maximizing a relative margin separating the data classes. This model is considered as an improvement to KSVM by incorporating the data spread information represented by the dominant normal directions to the decision boundary. This can also be viewed as an extension to the KNDA where the support vectors improve the choice of κ-nearest neighbors ($\kappa - NN$'s) on the decision boundary by incorporating local information. Since our model is an extension to both SVM and NDA, it can deal with heteroscedastic and non-normal data. It also avoids the small sample size problem. Interestingly, the proposed improvements only require a rigorous and simple combination of KNDA and KSVM objective functions, and preserve the computational efficiency of KSVM. Through the optimization of the CKSVMNDA objective function, surprising performance gains were achieved on real-world problems.

 In: R. Bergmann, S. Görg, G. Müller (Eds.): Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB. Trier, Germany, 7.-9. October 2015, published at http://ceur-ws.org

[1]Corresponding Author.

Keywords: Kernel Nonparametric Discriminant Analysis, Kernel Support Vector Machines, Partially Global Information, Local Information, Small Sample Size Problem.

1 Introduction

Supervised learning is the task of finding a function which relates inputs and targets. A training set $\mathcal{X}$ of input vectors $\{x_i\}_{i=1}^N$ is given, where $x_i \in \mathbb{R}^k (k \geq 1) \quad \forall i = 1, 2, ..., N$. The corresponding set $\mathcal{T}$ of tags are $\{t_i\}_{i=1}^N$, where $t_i \in 0, 1 \quad \forall i = 1, 2, ..., N$. The objective is to learn a model of dependency of the targets on the inputs. The ultimate goal is to be able to make accurate predictions of t for unseen values of x. Typically, we base our predictions upon some function $y(x)$ defined over the input/training space $\mathcal{X}$, and learning is the process of inferring the parameters of this function. A new representation of data is necessary to learn non-linear relations with a linear classifier. This is equivalent to applying a fixed non-linear mapping $\mathcal{F}$ of the data to a feature space, in which the linear classifier can be used. Hence, the objective function will be of the form:

$$y(x; w) = \sum_{i=1}^{N} f_i^x w_i + w_0 = \Phi^T(x_i)\mathbf{w} + w_0, \tag{1}$$

where $\Phi(x) = (f_1^x, f_2^x, \ldots, f_N^x) : \mathcal{X} \rightarrow \mathcal{F}$ describes a non-linear mapping from the input space to a feature space for the input variable x. Hence, non-linear classifiers have two stages: (i) a fixed non-linear mapping transforms the data into a feature space $\mathcal{F}$ and then (ii) a linear classifier is used to classify them in $\mathcal{F}$. Analysis of functions of the type (1) is facilitated since the adjustable weight vector $\mathbf{w}$ and the offset w_0 appear linearly, and the objective is to estimate optimum values of the weight coefficients. There are a large number of functions of type (1). Our concentration here is on some relevant state-of-the-art kernel-based models, such as, the Kernel Support Vector Machine (KSVM) and the Nonparametric Discriminant Analysis in kernel space, which we will call KNDA. KNDA extends the linear NDA based on the same principles that the Kernel Fisher Discriminant Analysis (KFD) is built upon. The advantage of KNDA over KFD is the relaxation of normality assumption. KNDA measures the between-class scatter matrix on a local basis in the neighborhood of the decision boundary in the higher dimensional feature space. This is based on the observation that the normal vectors on the decision boundary are the most informative for discrimination. In case of a two-class classification problem, these normal vectors are approximated by the $\kappa - NN$'s from the other class for one point. We can consider KND as a classifier based on the "near-global" characteristics of data. Although KNDA gets rid of the underlying assumptions of KFD and results in better classification performance, no additional importance is given to the boundary samples. In other words, the margin criterion (as calculated in KSVM) is not considered here. Moreover, it is not always an easy task to find

a common and appropriate choice of $\kappa - NN$'s on the decision boundary for all data points to obtain the best linear discrimination.

Another category of kernel-based classifiers is the Kernel Support Vector Machine (KSVM). KSVM is based on the idea of maximizing the margin or degree of separation in the training data. There are many hyperplanes which can divide the data between two classes for classification. One reasonable choice for the optimal hyperplane is the one which represents the largest separation or margin between the two classes. KSVM tries to find the optimal hyperplane using support vectors. The support vectors are the training samples that approximate the optimal separating hyperplane and are the most difficult patterns to classify. In other words, they are consisted of those data points which are closest to the optimal hyperplane. As KSVM deals with a subset of data points (support vectors) which are close to the decision boundary, it can be said that the KSVM solution is based on the "local" variations of the training data.

It has been shown in the literature that maximum margin based classifiers like the KSVM typically perform better than discriminant (or average margin) based methods like the KNDA due to their robustness and local margin consideration. However, KSVM can perform poorly when the data varies in such a way that data points exist far from the classification boundary [12]. This can be the case especially when the data is of high dimension. This is because KSVM does not take into consideration the "near-global" properties of the class distribution (as in the case of KNDA). This limitation of KSVM can be avoided by incorporating variational information from the KNDA which will control the direction of the separating hyperplane of KSVM. In that way we will have a maximum margin based classifier which is not sensitive to skewed data distribution like KSVM.

Several methods exist in literature which have addressed these issues inherent in discriminant based and maximum margin based methods. The ellipsoidal kernel machine was proposed in [11], where a geometric modification is proposed for data normalization by considering hyperellipsoids instead of hyperspheres in the classical KSVM method. Similarly, in [5], radius/margin bound has been used to iteratively optimize the parameters of KSVM efficiently. In [15], a kernel-based method has been proposed which essentially calculates the KFD scatter matrices based on the support vectors provided by KSVM. While these methods were backed by experimental improvements, most of them are a combination of multiple locally optimal algorithms to separately solve the discriminant based problem and margin maximization rather than providing one algorithm with one unique globally optimum solution.

Although the method proposed in [12] is superior to the previously described methods in the sense that it is based on only one convex optimization problem, it does so by introducing new constraints to the optimization problem. New constraints means new Lagrangian variables, which in turns can degrade the computational time. The Gaussian Margin Machine proposed in [3] tries to find the least informative distribution that classifies training data correctly by maintaining a Gaussian distribution of weight vectors. The drawback with this

method is the expensive objective function involving log determinants in the optimization problem.

Another approach to improve the classification performance of KSVM is to include additional training examples. This approach has been used in [1], where additional unlabeled samples are made available to the system in a semi-supervised learning system. The approach in [14] introduces a *neither* class, where additional samples are drawn from the same distribution for the classes under consideration. These additional samples are then used for improved margin consideration. However, we will stick to the simple binary classification model which does not rely on any additional assumption and, hence, is closer to a real-life pattern recognition problem in its truest form.

We propose a novel CKSVMNDA model which combines the KNDA and KSVM methods. In that way, a decision boundary is obtained which reflects both near-global characteristics (realized by KNDA) of the training data in feature space and its local properties (realized by the local margin concept of the KSVM). Being a kernel-based model, CKSVMNDA can deal with nonlinearly separable data efficiently. Rather than introducing new constraints like [12], our method modifies the objective function of KSVM by incorporating the scatter matrices provided by KNDA.

The proposed method improves upon our recently proposed models [6, 7] by preserving the same discriminative way while adding the following significant advantages:

- Unlike the method in [6], our proposed model is more theoretically founded and forms a convex optimization problem because the final matrix used to modify the objective function is positive-definite. As a result, the method generates one global optimum solution. Because of this global extremum, existing numerical methods can be used to solve this problem easily and efficiently.
- The methods in [6, 7] primarily focused on the linear version of SVM while our model derivation emphasizes on the kernel space. As stated before, the kernel space has the advantage of being able to learn non-linear relations by mapping to a higher-dimensional feature space.

We also show that our method is a variation of the KSVM optimization problem, so that even existing KSVM implementations can be used. The experimental results on real and artificial datasets show the superiority of our method both in terms of accuracy.

The rest of the paper is organized as follows: Section 2 provides formulations of the KSVM and KNDA. Section 3 contains derivation of the novel CKSVMNDA model. Section 4 provides a comparative evaluation of the CKSVMNDA model to the KSVM and KNDA methods. This evaluation is carried out on a number of benchmark real datasets. Finally, Section 5 provides some conclusions.

2 KSVM and KNDA

Let $\mathcal{X}_1 = \{x_i\}_{i=1}^{N_1}$ and $\mathcal{X}_2 = \{x_i\}_{i=N_1+1}^{N_1+N_2}$ be two different classes constituting an input space of $N = N_1 + N_2$ samples or vectors in $\mathbb{R}^M$ where, class $\mathcal{X}_1$ contains N_1 samples and class $\mathcal{X}_2$ contains N_2 samples. Let the associated tags with these vectors be represented by $\mathcal{T} = \{t_i\}_{i=1}^{N}$, where $t_i \in \{0, 1\}\ \ \forall i = 1, 2, \ldots, N$. Since real-life data has inherent non-linearity, KSVM tries to map the data samples to a higher dimensional feature space $\mathcal{F}$, where linear classification might be achieved. Let the function Φ map the classes $\mathcal{X}_1$ and $\mathcal{X}_2$ to two higher dimensional feature classes $\mathcal{F}_1 = \{\Phi(x_i)\}_{i=1}^{N_1}$ and $\mathcal{F}_2 = \{\Phi(x_i)\}_{i=N_1+1}^{N}$, respectively.

However, in case when the dimension of $\mathcal{F}$ is very high, it is not possible to do mapping directly. In such a case, the *kernel trick* [13] is used. Instead of explicitly calculating the mapping, a kernel function $\mathcal{K}$ is used, which calculates the dot products of the higher dimensional data samples instead of the samples themselves. Mathematically it can be written as

$$\mathcal{K}(x_i, x_j) =< \Phi(x_i).\Phi(x_j) >, \forall i, j \in \{1, 2, \ldots, N\}.$$

Our target is to learn the weight vector $\mathbf{w}$ which minimizes (or maximizes) some objective function of the form of Equation (1).

2.1 The Kernel Support Vector Machine

As stated before, KSVM tries to map the samples to a higher dimensional feature space in the hope that the classification problem will be linear in that space. In the feature space, KSVM tries to find the optimal decision hyperplane. The optimal hyperplane is the one with the largest margin, or, in other words, the plane which has largest minimal distance from any of the samples. Maximizing the distance of samples to the optimal decision hyperplane is equivalent to minimizing the norm of $\mathbf{w}$. As a result, this becomes part of the objective function. However, it might be the case that the problem is non-linear even in the higher dimensional space. To solve this, the margin constraint is relaxed or *slacked*. Also, a penalty factor is introduced in the objective function to control the amount of slack. This penalty factor is of the form of a loss function, usually a hinge loss function. Incorporating all these, the KSVM optimization problem can be written as:

$$\min_{\mathbf{w}\neq 0, w_0} \left\{ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} max(0, 1 - t_i(\Phi^T(x_i)\mathbf{w} + w_0)) \right\},$$

Here, $max(0, 1 - t_i(\Phi^T(x_i)\mathbf{w} + w_0))$ is the hinge loss function. For correctly classified training samples, this function does not incur any loss. For misclassification, the loss factor is controlled by C. Note that although KSVM is generally described as an optimization problem with constraints on the weights, we are presenting it slightly differently with the hinge-loss function so that it will be

easier to derive the probabilistic interpretation of our proposed method later. This representation can easily be converted to the more familiar constrained optimization problem.

Since the weight vector $\mathbf{w}$ resides in the feature space, it cannot be calculated directly. Instead, the Lagrangian dual problem is solved [10]. The optimal weight vector for this problem is a linear combination of the data points and is of the form $\mathbf{w}^* = \sum_{i=1}^{N} t_i \alpha_i^* \Phi(x_i)$, where $\{\alpha_i\}_{i=1}^{N}$ are the Lagrangian variables. The decision function for any test sample x is obtained by:

$$g(x) = \sum_{i=1}^{N} t_i \alpha_i^* \mathcal{K}(x, x_i) + w_0^*, \tag{2}$$

where w_0^* is computed using the primal-dual relationship, and where only samples with non-zero Lagrange multipliers α_i contribute to the solution. The corresponding data samples are called Support Vectors (SVs). These points are the crucial samples for classification. Therefore, KSVM considers only those data points which are close to the decision hyperplane and are critical to find the decision boundary. In other words, KSVM only considers the local variations in data samples. The overall distributions of the training samples are not taken into consideration. Incorporating some kind of global distribution (e.g. results from classifiers like KNDA) can provide better classification.

2.2 The Kernel Nonparametric Discriminant Analysis

The NDA can be extended to the feature space $\mathcal{F}$. We call this the Kernel Nonparametric Discriminant Analysis (KNDA). Instead of calculating the simple mean vectors, the nearest neighbor mean vectors are calculated to formulate the between-class scatter matrix of the NDA. In our feature space, this vector can be defined as:

$$M_m^{\kappa}(\Phi(x_i)) = \frac{1}{\kappa} \sum_{j=1}^{\kappa} \Phi(x)_{NN}(j), \tag{3}$$

where, $\Phi(x_i)_{NN}(j)$ defines the j^{th} nearest neighbor from data point x_i of class m. κ is the free parameter which defines how many neighbors to consider. This parameter needs to be optimized for each dataset. Now, let us define two matrices $L_1(\Phi(x_i))$ and $L_2(\Phi(x_i))$. We will use the kernel trick to formulate these matrices. In that case, the matrices are calculated on a component by component basis, where, a component of $L_1(\Phi(x_i))$ is defined as:

$$\begin{aligned} &(L_1(\Phi(x_i)))_j = \mathcal{K}(x_j, x_i) - (M_2^{\kappa}(\Phi(x_i)))_j, \\ &\forall i \in \{1, 2, \ldots, N_1\}, \forall j \in \{1, 2, \ldots, N\}, \end{aligned} \tag{4}$$

and a component of $L_2(\Phi(x_i))$ is defined as

$$\begin{aligned} &(L_2(\Phi(x_i)))_j = \mathcal{K}(x_j, x_i) - (M_1^{\kappa}(\Phi(x_i)))_j, \\ &\forall i \in \{N_1 + 1, N_1 + 2, \ldots, N_1 + N_2\}, \forall j \in \{1, 2, \ldots, N\}. \end{aligned} \tag{5}$$

With these formulations, the between-class scatter matrix in the feature space can be defined as:

$$\nabla = \frac{1}{(N_1+N_2)} \sum_{i=1}^{N_1} \Psi_i L_1(\Phi(x_i)) L_1(\Phi(x_i))^T + \frac{1}{(N_1+N_2)} \sum_{i=N_1+1}^{N_1+N_2} \Psi_i L_2(\Phi(x_i)) L_2(\Phi(x_i))^T. \quad (6)$$

Here, Ψ_i are the weighting functions to nullify the effects of samples that are far from the boundary. It is defined as follows [4]:

$$\Psi_i = \frac{min\{d(\Phi(x_i), \Phi(xNN_{1i}^{\kappa}))^{\gamma}, d(\Phi(x_i), \Phi(xNN_{2i}^{\kappa}))^{\gamma}\}}{d(\Phi(x_i), \Phi(xNN_{1i}^{\kappa}))^{\gamma} + d(\Phi(x_i), \Phi(xNN_{2i}^{\kappa}))^{\gamma}}, \quad (7)$$

where γ is a control parameter which can range from zero to infinity, and $d(\Phi(x_i), \Phi(xNN_{ji}^{\kappa}))$ is the Euclidean distance from x_i to its $\kappa - NN$'s from class $\mathcal{X}_j$ in the kernel space. γ controls how rapidly the value of weighting function falls to zero as we move away from the classification boundary.

The motivation behind KNDA is the observation that essentially the nearest neighbors represent the classification structure in the best way. For small values of κ, the matrices in Equation (4) and (5) represent the direction of the gradients of the respective class density functions in the feature space. If the weighting functions are not used, samples with large gradients that are far from the boundary may pollute the necessary information. Hence, these gradients with combination of the weighting functions form the between-class scatter matrix ∇, which preserves the classification structure.

The KNDA does not make any modifications to the within-class scatter matrix. As a result, the formula for within-class scatter matrix Δ is similar to the KFD, and can be written as follows:

$$\Delta = \mathbf{K}_1(I - 1_{N_1})\mathbf{K}_1^T + \mathbf{K}_2(I - 1_{N_2})\mathbf{K}_2^T, \quad (8)$$

where $\mathbf{K}_1$ is a $N \times N_1$ Kernel matrix for the class $\mathcal{X}_1$ and $\mathbf{K}_2$ is a $N \times N_2$ Kernel matrix for the class $\mathcal{X}_2$. I is the identity matrix and 1_{N_1} and 1_{N_2} are the matrices with all entries $\frac{1}{N_1}$ and $\frac{1}{N_2}$, respectively. With these definitions of ∇ and Δ, the KNDA method proceeds by computing the eigenvectors and eigenvalues of $\Delta^{-1}\nabla$. Since the higher dimensional feature space $\mathcal{F}$ is of dimension N, the matrix Δ is needed to be regularized before calculating the inverse. This is achieved by adding a small multiple β of the identity matrix I. Hence, the eigenvectors and eigenvalues of $(\Delta + \beta I)^{-1}\nabla$ are computed, and the eigenvector corresponding to the largest eigenvalue forms the optimal decision hyperplane. We can exploit the fact that the matrix ∇ is only of rank one (i.e., $\alpha^T \nabla \alpha = \frac{1}{(N_1+N_2)} \sum_{i=1}^{N_1} \Psi_i \big(\alpha^T L_1(\Phi(x_i)) L_1(\Phi(x_i))^T\big) + \frac{1}{(N_1+N_2)} \sum_{i=N_1+1}^{N_1+N_2} \Psi_i \big(\alpha^T L_2(\Phi(x_i)) L_2(\Phi(x_i))^T\big)$). Thus, we can fix $\alpha^T \nabla \alpha$ to any non-zero value, for example 1 and minimize $\alpha^T \Delta \alpha$. This amounts to the following quadratic optimization problem:

$$\min_{\alpha \neq 0, \alpha_0} \quad \alpha^T \Delta \alpha, \tag{9}$$

$$s.t. \quad \alpha^T \nabla \alpha = 1. \tag{10}$$

3 The CKSVMNDA Model

In this section, we present our proposed model CKSVMNDA which combines the data spread information represented by the normal vectors to the decision surface for the KNDA (partially global information), and the support vectors for the KSVM (local information). Thus, the CKSVMNDA overcomes the drawbacks of KSVM by controlling the spread of the data, which is represented by the KNDA dominant normal directions to the decision boundary, while maximizing a relative margin separating the data classes. Moreover, the choice of KNDA κ-nearest neighbors ($\kappa - NN$'s) on the decision boundary is improved by the KSVM support vectors. Therefore, the CKSVMNDA objective function is a simple and rigorous summation of the KSVM and KNDA objective functions:

$$\min_{\alpha \neq 0, \alpha_0} \quad \Big\{ \frac{1}{2} \alpha^T \big[2\lambda \big(\Delta + \beta \nabla \big) + I \big] \alpha - \lambda \beta \tag{11}$$

$$+ C \sum_{i=1}^{N} max(0, 1 - t_i(\Phi^T(x_i)\alpha + \alpha_0)) \Big\}. \tag{12}$$

In theory, CKSVMNDA should outperform both KSVM and KNDA if the control parameter λ can be optimally chosen. In practice, the values of λ and β will be tuned via the cross validation technique, where data is divided into a number of subsets. Then, one subset is used for testing while the others are used for training. All the subsets are used for testing in turns and the average is taken into consideration to reduce variability. This whole process is repeated with different values of λ and β. The latter are assigned the values with the best performance.

3.1 Solving the Optimization Problem

Since our optimization problem is similar to the KSVM optimization problem, we can solve it in a similar way, i.e., by using Lagrange multipliers. However, obtaining the CKSVMNDA solution this way requires an entirely new implementation to test this method. The following lemma gives us an easier alternative to implement this method:

Lemma 1. *The CKSVMNDA method formulation is equivalent to:*

$$\min_{\hat{\mathbf{w}} \neq 0, w_0} \quad \Big\{ \frac{1}{2} \hat{\mathbf{w}}^T \hat{\mathbf{w}} + C \sum_{i=1}^{N} max(0, 1 - t_i(\hat{\Phi}^T(x_i)\hat{\mathbf{w}} + w_0)) \Big\}, \tag{13}$$

where

$$\hat{\mathbf{w}} = \Theta^{1/2}\mathbf{w}, \tag{14}$$

$$\hat{\Phi}(x_i) = \Theta^{-1/2}\Phi(x_i) \quad \forall i = 1, \ldots, N \tag{15}$$

and

$$\Theta = \eta\Delta(\nabla + \beta I)^{-1}\Delta + I. \tag{16}$$

Proof. Substituting Equations (14-16) into equation (13) we get the original CKSVMNDA problem (Equation (11)).

This lemma gives us a significant advantage from the implementation viewpoint. This essentially means that we can use the existing SVM implementations [8] provided we can calculate the terms $\Theta^{1/2}$ and $\Theta^{-1/2}$. The algorithm used to solve the optimization problem in this implementation is based on the interior-reflective Newton method described in [2].

4 Experimental Results

In this section we evaluate the proposed CKSVMNDA method against three other contemporary classifiers, namely, the KSVM, KNDA and the Kernel Fisher Discriminant (KFD). To strengthen the significance of our method, we provide results for both real-world datasets and a face recognition application.

For kernelization of the data, we use the Gaussian RBF Kernel $\mathcal{K}(x_i, x_j) = e^{-\|x_i - x_j\|^2/\sigma}$. This kernel is proven to be robust and flexible. Here, σ represents the positive "width" parameter. For KNDA and KFD, after finding the optimal eigenvector, Bayes classifier was used for conducting the final classification.

The involved parameters were optimized using exhaustive search to try all possible combinations. Although the parameter optimization is a lengthy process, this needs to be done only once for a new dataset, and, hence, does not contribute to the actual classification performance. If the optimization needs to be faster, efficient methods like coordinate descent technique can be used at the cost of a small degradation in accuracy values.

The number of parameters to tune for the CKSVMNDA method is 4, while it is 2 for KSVM and KNDA. It might seem that an accurate fit of the parameter values is necessary for CKSVMNDA to perform well, specially if we have a small training dataset. But as we will see from the results, CKSVMNDA performs better compared to other methods by tuning over a limited range of parameter values we have used (e.g. we use a set of only 20 κ values and 20 η values for parameter tuning to obtain the results of Table 1). Since this is a combination of KSVM and KNDA, the parameters compensate each other, and the fit doesn't necessarily have to be perfect. Also, for small training set, we tackle the problem of poor performance due to inaccuracies in matrix inversion by adding a regularization term before inverting.

4.1 Experiments on Real and Artificial Datasets

We have applied the classification algorithms on 11 real-world and artificial datasets.The datasets are obtained from the Benchmark Repository used in [9]. Namely, the datasets are: Flare-Sonar, Breast-Cancer, German, Heart, Banana, Diabetes, Ringnorm, Thyroid, Twonorm, Waveform and Splice. These datasets are obtained from the UCI, DELVE and STATLOG repositories. Some of these datasets are originally multi-class. In such cases, some of the classes were (randomly) merged to convert it into a two-class classification problem. 100 partitions are then generated for each dataset, where about 60% data is used for training and the rest for testing [9]. For our experimental results, we randomly picked 5 out of these 100 partitions (5 partitions each for training and 5 each for testing). Additionally, we repeated this random picking process 5 times to achieve the average result. This randomness was introduced to ensure that no method has a coincidental advantage over the others. For parameter tuning, 5-fold cross validation on the training dataset was performed for each model (i.e. 4 out of the 5 picked training partitions were used for training and the remaining one for validation at each stage of cross validation).

4.2 Interpretation of the Results

Accuracy

Table 1 contains the average accuracy values and the standard deviations obtained over all the runs. We see that the CKSVMNDA method outperforms the KSVM, KNDA and the KFD in almost all cases. Since the CKSVMNDA combines the global and near-global variations provided by the KSVM and the KNDA, respectively, it can classify the relatively difficult test samples. Also, being a variation of the KSVM and KNDA, this method is free from any underlying distribution assumption, and, hence, can provide better results. Concerning the parameters λ and β, in order to reduce the time of optimization, we had to restrict ourselves to only a few values. Still, as we can see, these limited values are good enough for almost all the datasets. This establishes the fact that our method can be used in practical applications. To measure the statistical significance of the results, we paired up the CKSVMNDA method with the other methods and performed paired t-tests on the accuracy values. The paired t-test determines whether or not two paired sets of measured values are significantly different. The last row of Table 1 provides the confidence intervals (in %) obtained from the performed t-tests. This confidence interval quantifies the probability of the paired distributions being the same. The higher the confidence interval, the lower is the probability that the underlying distributions are statistically indifferent. As we can see, all the confidence intervals are almost 100%, which proves that the CKSVMNDA method indeed provides statistically significant accuracy improvements.

If we compare the results between the KNDA and KFD, we see that in some cases, the KFD provides better classification results than the KNDA. This is due to the fact that the optimal nearest neighbor parameter for the KNDA (the

$\kappa - NN$'s) is not always easy to find. But since our method combines the KNDA with KSVM, the optimality of this parameter is not as crucial as it is in the KNDA.

Computational Complexity Analysis

The computational complexity of the KSVM scales with $\mathcal{O}(N^2)$ for one iteration. The KNDA and KFD scale with a computational complexity of $\mathcal{O}(N^3)$ (dominated by the inversion of the within-class scatter matrix). Each of the KNDA and KFD methods requires only one run as there is no iterative process involved.

In the CKSVMNDA, the complexity for the inversion of Θ scales with $\mathcal{O}(N^3)$. However, this inversion process can be considered to be part of pre-processing, as it is needed to be done only once before start of the training. Therefore, the computational complexity of our proposed CKSVMNDA can be considered similar to that of the KSVM, i.e., $\mathcal{O}(N^2)$ per iteration. This can also be seen from the obtained results (second last row of Table 1), where we see that the average computational time of our method is on par with that of KSVM.

Dataset	CKSVMNDA	KSVM	KND	KFD
Flare-Sonar	**67.7** (0.47)	66.9 (0.41)	*67.1* (0.65)	66 (0.40)
Breast-Cancer	*78.9* (1.96)	77.4 (2.1)	**79.3** (2.00)	77 (2.37)
German	**78.1** (0.40)	*77* (0.38)	76.3 (0.68)	75.7 (0.51)
Heart	**86.5** (2.21)	*85.4* (2.3)	81.7 (1.58)	82.9 (2.13)
Banana	**89.8** (0.25)	*89.6* (0.29)	*89.6* (0.22)	89.5 (0.20)
Diabetes	**78.6** (0.50)	*77.7* (0.69)	75.7 (0.90)	77.3 (1.03)
Ringnorm	**98.5** (0.04)	*98.4* (0.04)	98.3 (0.03)	97.4 (0.07)
Thyroid	**97.3** (0.6)	96.5 (1.02)	*97.1* (0.64)	96.8 (0.49)
Twonorm	**97.7** (0.04)	*97.6* (0.05)	96.5 (0.32)	96.9 (0.08)
Waveform	**90.7** (0.15)	*90.5* (0.15)	89.3 (0.19)	90 (0.12)
Splice	**88.9** (0.41)	*88.7* (0.38)	88.5 (0.41)	88.4 (0.36)
Avg. time	4.04	4.05	2.95	2.92
Confidence	-	99.8	97.6	99.9

Table 1. Average percentage classification accuracy and standard deviation (in parentheses) of each method for the 11 data sets (best method in **bold**, second best *emphasized*). The last two rows contain the average cpu time for each method (in *seconds*) and the t-test confidence interval, respectively.

5 Conclusion

In this paper, we have proposed a novel classification method named CKSVMNDA. The CKSVMNDA method incorporates the global variational information from the KSVM and the near-global information from the KNDA. Being a combination of these two methods, CKSVMNDA is a robust classifier, free from any

underlying assumption regarding class distribution. Our method is also capable of tackling the small sample size problem. Being a convex optimization problem, our method provides a global optimum solution and can be solved efficiently by using numerical methods. Besides, we have shown that our method can be reduced to the classical KSVM model so that existing KSVM implementations can be used. The experimental results on some contemporary datasets verifies the superiority of our method, where we compare CKSVMNDA with the KSVM, KND and KFD. In future, we plan to build a multi-class classifier based on the principles of the CKSVMNDA method.

References

1. Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. On Manifold Regularization. In *Proceedings of the Artificial Intelligence and Statistics*, 2005.
2. T.F. Coleman and Y. Li. A reflective newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996.
3. Koby Crammer, Mark Dredze, and Fernando Pereira. Exact Convex Confidence-Weighted Learning. *Advances in Neural Information Processing Systems 21*, 2009.
4. K. Fukunaga. *Introduction to Statistical Pattern Recognition, second ed.* Academic Press, 2000.
5. S.S. Keerthi. Efficient Tuning of SVM Hyperparameters Using Radius/Margin Bound and Iterative Algorithms. *IEEE Transactions on Neural Networks*, 13(5):1225–1229, Sep 2002.
6. N.M. Khan, R. Ksantini, I. Ahmad, and B. Boufama. A novel SVM+NDA model for classification with an application to face recognition. *Pattern Recognition*, 45(1):66–79, 2012.
7. R. Ksantini and B. Boufama. Combining partially global and local characteristics for improved classification. *Int. J. Machine Learning & Cybernetics*, 3(2):119–131, 2012.
8. MATLAB Bioinformatics Toolbox. The mathworks^{TM}, 2011.
9. G. Ratsch, T. Onoda, and K.R. Muller. Soft Margins for Adaboost. *Machine Learning*, 42(3):287–320, 2000.
10. B. Scholkopf and A. Smola. *Learning With Kernels-Support Vector Machines, Regularization, Optimization and Beyond.* MA: MIT Press, Cambridge, 2001.
11. P.L. Shivaswamy and T. Jebara. Elliposoidal Kernel Machines. In *Proceedings of the Artificial Intelligence and Statistics*, 2007.
12. P.L. Shivaswamy and T. Jebara. Maximum relative margin and data-dependent regularization. *Journal of Machine Learning Research*, 11:747–788, 2010.
13. V.N. Vapnik. *Statistical Learning Theory.* John Wiley & Sons, New York, USA, 1998.
14. J. Weston, R. Collobert, F. H. Sinz, L. Bottou, and V. Vapnik. Inference with the universum. In *Proceedings of the International Conference on Machine Learning*, pages 1009–1016, 2006.
15. Baochang Zhang, Xilin Chen, Shiguang Shan, and Wen Gao. Nonlinear face recognition based on maximum average margin criterion. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 554 – 559, 2005.