Maxi-Min Margin Machine: Learning Large Margin Classifiers Locally and Globally

Kaizhu Huang, Haiqin Yang,

Irwin King *IEEE Member*, and Michael R. Lyu *IEEE Fellow*

Department of Computer Science and Engineering

The Chinese University of Hong Kong

Shatin, New Territories, Hong Kong

{kzhuang, hqyang, king, lyu}@cse.cuhk.edu.hk

Abstract

We propose a novel large margin classifier, called the Maxi-Min Margin Machine (M^4). This model learns the decision boundary both locally and globally. In comparison, other large margin classifiers construct separating hyperplanes only either locally or globally. For example, a state-of-the-art large margin classifier, the Support Vector Machine (SVM), considers data only locally, while another significant model, the Minimax Probability Machine (MPM), focuses on building the decision hyperplane exclusively based on the global information. As a major contribution, we show that SVM yields the same solution as M^4 when data satisfy certain conditions, and MPM can be regarded as a relaxation model of M^4. Moreover, based on our proposed local and global view of data, another popular model, the Linear Discriminant Analysis, can easily be interpreted and extended as well. We describe the M^4 model definition, provide a geometrical interpretation, present theoretical justifications, and propose a practical sequential conic programming method to solve the optimization problem. We also show how to exploit Mercer kernels to extend M^4 for nonlinear classifications. Furthermore, we perform a series of evaluations on both synthetic data sets and real world benchmark data sets. Comparison with SVM and MPM demonstrates the advantages of our new model.

Index Terms

classification, large margin, kernel methods, second order cone programming, learning locally and globally

I. Introduction

Recently, large margin classifiers [18] have attracted much interest in the community of machine learning and pattern recognition. The Support Vector Machine (SVM) [25][21][10], the most famous of them, represents a state-of-the-art classifier. The essential point of SVM is to find a linear separating hyperplane which achieves the maximal margin among different classes of data. Furthermore, one can extend SVM to build nonlinear separating decision hyperplanes by exploiting kernelization techniques.

However, SVM obtains the decision hyperplane in a "local" way, i.e., the decision boundary is exclusively determined by a number of critical points, which are called *support vectors*, whereas all other points are irrelevant to this hyperplane. Although this scheme has been shown to be powerful both theoretically and empirically, it discards the global information in the data.

An illustration example can be seen in Figure 1. In this figure, the classification boundary is intuitively observed to be mainly determined by the dotted axis, i.e., the long axis of the Y data (represented by $\Box$'s) or the short axis of the X data (represented by $\circ$'s). Moreover, along this axis, the Y data are more likely to be scattered than the X data, since the Y data contain a relatively larger variance in this direction. Noting this "global" fact, a good decision hyperplane seems reasonable to lie closer to the X side (see the dash-dot line). However, SVM ignores this kind of "global" information, i.e., the statistical trend of data occurrence: the derived SVM decision hyperplane (the solid line) lies unbiasedly right in the middle of two "local" points (the support vectors).[1]

Aiming to construct classifiers both locally and globally, we propose the Maxi-Min Margin Machine (M^4) in this paper. As we show later, one key contribution of this novel model is that M^4 is closely related to SVM and an important model, the Minimax Probability Machine (MPM) [9]. More specifically, SVM yields the same solution as M^4 when data satisfy certain conditions, while MPM can be regarded as a relaxation method of our proposed model. Moreover, based on our proposed local and global view of data, another popular model, the Linear Discriminant Analysis (LDA) [4], can easily be interpreted and extended as well.

Another good feature of the M^4 model is that it can be cast as a sequential conic programming problem [17], or more specifically, a sequential Second Order Cone Programming (SOCP) problem [11], [14], which thus can be solved practically in polynomial time. In addition, with incorporating the global

[1]Note that two classes of data cannot easily be scaled to the same data trend simultaneously. The only viable approach is to scale one class of data first by using a certain transformation, while the other class of data needs to be scaled subsequently, based on the same transformation.

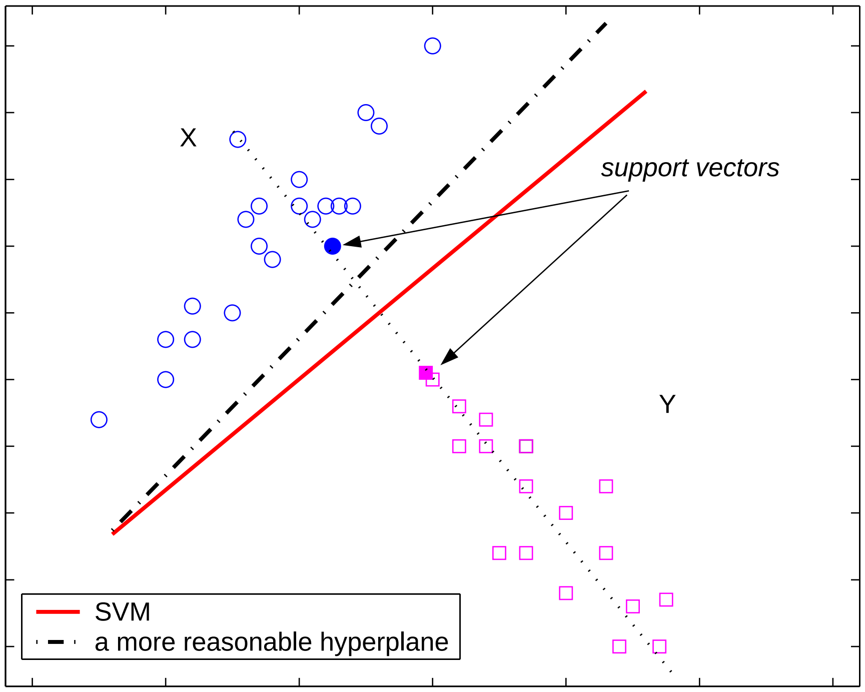


Fig. 1. A decision hyperplane with considerations of both local and global information.

information, a reduction method is proposed for decreasing the computational time of this new model.

The third important feature of our proposed model is that, the kernelization methodology is also applicable for this formulation. This thus generalizes the linear M^4 into a more powerful classification approach, which can derive nonlinear decision boundaries.

The rest of this paper is organized as follows. In the next section, we introduce the M^4 model in detail, including its model definition, the geometrical interpretation, connections with other models, and the associated solving methods. In Section III, we develop a reduction method to remove redundant points, in order to decrease the computational time. In Section IV, we exploit kernelization to extend M^4 to nonlinear classification tasks. In Section V, we evaluate this novel model on both synthetic data sets and real world benchmark data sets. In Section VI, we discuss the M^4 model and also present future work. Finally, we conclude this paper in Section VII. Some results of Section II, IV, and V of this paper have appeared earlier in [5], but they are expanded significantly both theoretically and experimentally in the current paper, while the remaining sections are new.

II. Maxi-Min Margin Machine

In the following, we first, for the purpose of clarity, divide M^4 into separable and nonseparable categories, and then introduce the corresponding hard-margin M^4 and soft-margin M^4 in turn. In this section, we will also establish the connections of the M^4 model with other large margin classifiers including SVM, MPM, LDA, and the Mininum Error Minimax Probability Machine (MEMPM) [6].

A. Separable Case

Assuming the classification samples are separable, we first introduce the model definition and the geometrical interpretation. We then transform the model optimization problem into a sequential SOCP problem and discuss the optimization method.

1) Problem Definition: Only two-category classification tasks are considered in this paper. Let a training data set contain two classes of samples X and Y, represented by $\mathbf{x}_i \in \mathbb{R}^n$ and $\mathbf{y}_j \in \mathbb{R}^n$ respectively, where $i = 1, 2, \ldots, N_{\mathbf{x}}$, $j = 1, 2, \ldots, N_{\mathbf{y}}$. Hereafter, we denote the total number of samples as $N = N_{\mathbf{x}} + N_{\mathbf{y}}$. The basic task here can be informally described as finding a suitable hyperplane $f(\mathbf{z}) = \mathbf{w}^T\mathbf{z} + b$ separating the two classes of data as robustly as possible ($\mathbf{w} \in \mathbb{R}^n \backslash \{\mathbf{0}\}$, $b \in \mathbb{R}$, and $\mathbf{w}^T$ is the transpose of $\mathbf{w}$). Future data points $\mathbf{z}$ for which $f(\mathbf{z}) \geq 0$ are then classified as class X; otherwise, they are classified as class Y.

The formulation for M^4 can be written as:

$$\max_{\rho, \mathbf{w} \neq \mathbf{0}, b} \quad \rho \quad s.t. \tag{1}$$

$$\frac{(\mathbf{w}^T\mathbf{x}_i + b)}{\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w}}} \geq \rho, \quad i = 1, 2, \ldots, N_{\mathbf{x}}\ , \tag{2}$$

$$\frac{-(\mathbf{w}^T\mathbf{y}_j + b)}{\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}} \geq \rho, \quad j = 1, 2, \ldots, N_{\mathbf{y}}\ , \tag{3}$$

where $\mathbf{\Sigma}_{\mathbf{x}}$ and $\mathbf{\Sigma}_{\mathbf{y}}$ refer to the covariance matrices of the X and the Y data, respectively.[2]

This model tries to *maximize* the margin defined as the *minimum* Mahalanobis distance for all training samples,[3] while simultaneously classifying all the data correctly. Compared to SVM, M^4 incorporates the data information in a global way; specifically, in our model, the covariance information of the data or the

[2]For simplicity, we assume $\mathbf{\Sigma}_x$ and $\mathbf{\Sigma}_y$ are always positive definite. In practice, this can be satisfied by adding a small positive amount into their diagonal elements, which is a widely used technique.

[3]This suggested the name of our model.

statistical trend of data occurrence is considered, while SVMs, including l_1-SVM [27] and l_2-SVM [24],[4] simply discard this information or use the same covariance for each class.

One important feature of M^4 is that its solution is invariant with respect to invertible linear transformations of data. This is verified as follows. Assume an invertible linear transformation is given as $Tr(\mathbf{x}_i) = A\mathbf{x}_i$, $Tr(\mathbf{y}_i) = A\mathbf{y}_i$, where A is an invertible matrix. By using $\mathbf{\Sigma}_{Tr(\mathbf{x})} = A\mathbf{\Sigma}_{\mathbf{x}}A^T$, $\mathbf{\Sigma}_{Tr(\mathbf{y})} = A\mathbf{\Sigma}_{\mathbf{y}}A^T$, we can formulate the optimization for the transformed data as follows:

$$\max_{\rho^*, \mathbf{w}^* \neq \mathbf{0}, b^*} \quad \rho \quad s.t. \tag{4}$$

$$\frac{(\mathbf{w}^{*T} A\mathbf{x}_i + b^*)}{\sqrt{\mathbf{w}^{*T} A\mathbf{\Sigma}_{\mathbf{x}} A^T \mathbf{w}^*}} \geq \rho^*, \quad i = 1, 2, \ldots, N_{\mathbf{x}} \ , \tag{5}$$

$$\frac{-(\mathbf{w}^{*T} A\mathbf{y}_j + b^*)}{\sqrt{\mathbf{w}^{*T} A\mathbf{\Sigma}_{\mathbf{y}} A^T \mathbf{w}^*}} \geq \rho*, \quad j = 1, 2, \ldots, N_{\mathbf{y}} \ . \tag{6}$$

In the above, $\{\rho^*, \mathbf{w}^*, b^*\}$ denotes the variables to be optimized in the transformed problem. As observed from the optimization, if $\mathbf{w}$ maximizes (1)-(3), $A^T\mathbf{w}^*$ should be equal to $\mathbf{w}$. Therefore we have the following:

$$A^T\mathbf{w}^* = \mathbf{w} \Rightarrow \mathbf{w}^* = (A^T)^{-1}\mathbf{w} \ . \tag{7}$$

For an input data point $\mathbf{z}$, we have $\mathbf{w}^{*T}Tr(\mathbf{z}) = \mathbf{w}^{*T}A\mathbf{z} = \mathbf{w}^T A^{-1} A\mathbf{z} = \mathbf{w}^T\mathbf{z}$. Hence, the derived decision boundary is invariant against invertible linear transformations.

2) Geometrical Interpretation: A geometrical interpretation of M^4 can be seen in Figure 2. In this figure, the X data are represented by the inner ellipsoid on the left side with its center at $\mathbf{x}_0$, while the Y data are represented by the inner ellipsoid on the right side with its center at $\mathbf{y}_0$. It is observed that these two ellipsoids contain unequal covariances or risks of data occurrence. However, SVM does not consider this global information: its decision hyperplane (the dotted blue line) locates unbiasedly midway between two support vectors (filled points). In contrast, M^4 defines the margin as a Maxi-Min Mahalanobis distance, which constructs a decision plane (the solid magenta line) taking account of both the local and the global information: the M^4 hyperplane corresponds to the tangent line of two dashed ellipsoids centered at the support vectors (the local information) and shaped by the corresponding covariances (the global information).

[4] l_p-SVM means the "p-norm" distance-based SVM.

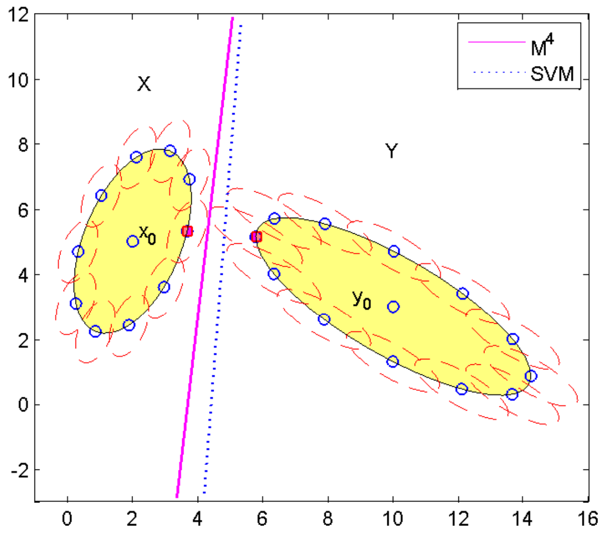


Fig. 2. A geometric interpretation of M^4. The M^4 hyperplane corresponds to the tangent line (the solid magenta line) of two small dashed ellipsoids centered at the support vectors (the local information) and shaped by the corresponding covariances (the global information). It is thus more reasonable than the hyperplane generated by SVM (the dotted line).

3) Optimization Method: In the following, we propose the optimization method for the M^4 model. We will demonstrate that the above problem can be cast as a sequential conic programming problem, or more specifically, a sequential SOCP problem.

Our strategy is based on the "Divide and Conquer" technique. One may note that in the optimization problem of M^4, if ρ is fixed to a constant ρ_n, the problem to "conquer" changes exactly into the problem of checking whether the constraints of (2) and (3) can be satisfied. Moreover, as will be demonstrated shortly, this "checking" procedure can be stated as an SOCP problem. Thus the problem now becomes one of determining how ρ is set, which we can use a "divide" strategy to handle: if the constraints are satisfied, we can increase ρ_n accordingly; otherwise, we decrease ρ_n.

We detail this solving technique in the following two steps:

1. **Divide:** Set $\rho_n = (\rho_0 + \rho_m)/2$, where ρ_0 is a feasible ρ, ρ_m is an infeasible ρ, and $\rho_0 \le \rho_m$.
2. **Conquer:** Call the Modified Second Order Cone Programming (MSOCP) procedure elaborated in the following discussion to check whether ρ_n is a feasible ρ. If yes, set $\rho_0 = \rho_n$; otherwise, set $\rho_m = \rho_n$.

In the above, if a ρ value satisfies the constraints of (2) and (3), we call it a feasible ρ; otherwise, we call it an infeasible ρ. These two steps are iterated until $|\rho_0 - \rho_m| \leq \varepsilon$, where ε is a small positive value.[5]

We propose the following Theorem showing that the MSOCP procedure, namely, the checking problem with ρ fixed to a constant ρ_n, is solvable by casting it as an SOCP problem.

Theorem 1: The problem of checking whether there exist $\mathbf{w}$ and b satisfying the following two sets of constraints (8) and (9) can be transformed into an SOCP problem, which can be solved in polynomial time:

$$(\mathbf{w}^T\mathbf{x}_i + b) \geq \rho_n\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}},\ i = 1,\ldots,N_{\mathbf{x}}\ , \tag{8}$$

$$-(\mathbf{w}^T\mathbf{y}_j + b) \geq \rho_n\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}},\ j = 1,\ldots,N_{\mathbf{y}}\ , \tag{9}$$

$$\mathbf{w} \neq 0\ .$$

Proof: Introducing dummy variables $\boldsymbol{\tau}$, we rewrite the above checking problem into an equivalent optimization problem:

$$\begin{aligned}
&\max_{\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\tau}} \quad \{\min_{k=1}^{N} \quad \boldsymbol{\tau}_k\} \quad s.t.\\
&(\mathbf{w}^T\mathbf{x}_i + b) \geq \rho_n\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}} - \boldsymbol{\tau}_i,\\
&-(\mathbf{w}^T\mathbf{y}_j + b) \geq \rho_n\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}} - \boldsymbol{\tau}_{j+N_{\mathbf{x}}}\ ,\\
&\mathbf{w} \neq 0\ ,
\end{aligned}$$

where $i = 1,\ldots,N_{\mathbf{x}}$ and $j = 1,\ldots,N_{\mathbf{y}}$.

By checking whether the minimum $\boldsymbol{\tau}_k$ at the optimum point is positive, we can know whether the constraints of (2) and (3) can be satisfied. If we go further, we can introduce another dummy variable and transform the above problem into an SOCP problem:

$$\begin{aligned}
&\max_{\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\tau},\eta} \quad \eta \quad s.t.\\
&(\mathbf{w}^T\mathbf{x}_i + b) \geq \rho_n\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}} - \boldsymbol{\tau}_i\ ,\\
&-(\mathbf{w}^T\mathbf{y}_j + b) \geq \rho_n\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}} - \boldsymbol{\tau}_{j+N_{\mathbf{x}}}\ ,\\
&\eta \leq \boldsymbol{\tau}_k\ ,\\
&\mathbf{w} \neq 0\ ,
\end{aligned}$$

where $i = 1,\ldots,N_{\mathbf{x}}$, $j = 1,\ldots,N_{\mathbf{y}}$, and $k = 1,\ldots,N$. By checking whether the optimal η is greater than 0, we can immediately know whether there exist $\mathbf{w}$ and b satisfying the constraints of (2) and (3).

[5]The proposed solving technique is also referred to as the bi-section search method.

Moreover, the above optimization is easily verified to be the standard SOCP form, since the optimization function is a linear form and the constraints are either linear or the typical second order conic constraints.

■

Remarks. In practice, many SOCP programs, e.g., Sedumi [19], provide schemes to directly handle the above checking procedure. Therefore, it may not be necessary to introduce dummy variables as we have done in the proof. Furthermore, ρ_0 can often be set to zero, while ρ_{max} can simply be set to a large value, e.g., 50 as used in our experiments.

We now analyze the time complexity of M^4. As indicated in [11], if the SOCP is solved based on interior-point methods, it contains a worst-case complexity of $O(n^3)$. If we denote the range of feasible ρ's as $L = \rho_{max} - \rho_{min}$ and the required precision as ε, then the number of iterations for M^4 is $\log(L/\varepsilon)$ in the worst case. Adding the cost of forming the system matrix (constraint matrix), which is $O(Nn^3)$ (N represents the number of training points), the total complexity would be $O(n^3 \log(L/\varepsilon) + Nn^3) \approx O(Nn^3)$, a polynomial time complexity.

B. Connections with Other Models

In this section, we establish connections between M^4 and other models. We show that our model can be changed to SVM and MPM when certain settings are used. Moreover, LDA can be interpreted and extended according to our local and global views of data.

1) Connection with Minimax Probability Machine: If one performs constraint relaxation, i.e., expands the constraints of (2) and adds all of them together, one can immediately obtain the following:

$$\begin{aligned} & \mathbf{w}^T \sum_{i=1}^{N_{\mathbf{x}}} \mathbf{x}_i + N_{\mathbf{x}} b \geq N_{\mathbf{x}} \rho \sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{x}} \mathbf{w}} \ , \\ \Rightarrow \quad & \mathbf{w}^T \overline{\mathbf{x}} + b \geq \rho \sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{x}} \mathbf{w}} \ , \end{aligned} \tag{10}$$

where $\overline{\mathbf{x}}$ denotes the mean of the X training data.

Similarly, from (3) one can obtain:

$$\begin{aligned} & -(\mathbf{w}^T \sum_{j=1}^{N_{\mathbf{y}}} \mathbf{y}_j + N_{\mathbf{y}} b) \geq N_{\mathbf{y}} \rho \sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{y}} \mathbf{w}} \ , \\ \Rightarrow \quad & -(\mathbf{w}^T \overline{\mathbf{y}} + b) \geq \rho \sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{y}} \mathbf{w}} \ , \end{aligned} \tag{11}$$

where $\overline{\mathbf{y}}$ denotes the mean of the Y training data.

Adding (10) and (11), one can obtain:

$$\max_{\rho, \mathbf{w}\neq\mathbf{0}} \quad \rho \quad s.t.$$

$$\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho(\sqrt{\mathbf{w}^T\mathbf{\Sigma_x w}} + \sqrt{\mathbf{w}^T\mathbf{\Sigma_y w}}) \ . \tag{12}$$

The above optimization is exactly the MPM optimization [9]. Note, however, that the above procedure cannot be reversed. This means that MPM is looser than M^4. In another word, MPM is actually a relaxation model of M^4.

Remarks. In MPM, since the decision is completely determined by the global information, namely, the mean and covariance matrices [9],[6] the estimates of mean and covariance matrices need to be reliable in order to ensure accurate performance. However, this may not always be the case in real world tasks. On the other hand, M^4 seems to solve this problem in a natural way, because the impact caused by inaccurately estimated mean and covariance matrices can be partly neutralized by utilizing the local information, namely by satisfying the constraints of (2) and (3) for each local data point. This will be demonstrated later in the experiments.

2) Connection with Support Vector Machine: If one assumes $\mathbf{\Sigma_x} = \mathbf{\Sigma_y} = \mathbf{\Sigma}$, the optimization of M^4 can be changed to:

$$\max_{\rho, \mathbf{w}\neq\mathbf{0}, b} \quad \rho \quad s.t.$$

$$(\mathbf{w}^T\mathbf{x}_i + b) \geq \rho\sqrt{\mathbf{w}^T\mathbf{\Sigma w}} \ ,$$

$$-(\mathbf{w}^T\mathbf{y}_j + b) \geq \rho\sqrt{\mathbf{w}^T\mathbf{\Sigma w}} \ ,$$

where $i = 1, \ldots, N_\mathbf{x}$ and $j = 1, \ldots, N_\mathbf{y}$.

Observing that the magnitude of $\mathbf{w}$ will not influence the optimization, without loss of generality, one can further assume $\rho\sqrt{\mathbf{w}^T\mathbf{\Sigma w}} = 1$. Therefore the optimization can be changed to:

$$\min_{\mathbf{w}\neq\mathbf{0}, b} \quad \mathbf{w}^T\mathbf{\Sigma w} \quad s.t. \tag{13}$$

$$(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \ , \tag{14}$$

$$-(\mathbf{w}^T\mathbf{y}_j + b) \geq 1 \ , \tag{15}$$

where $i = 1, \ldots, N_\mathbf{x}$ and $j = 1, \ldots, N_\mathbf{y}$.[7]

[6]This can be directly observed from (12).

[7]The optimization of (13)-(15) is actually a standard SVM formulation with the linear kernel $K(\mathbf{z}_1, \mathbf{z}_2) = \mathbf{z}_1^T\mathbf{\Sigma}^{-1}\mathbf{z}_2$.

A special case of the above with $\mathbf{\Sigma} = \mathbf{I}$ is precisely the optimization of SVM, where $\mathbf{I}$ is the unit matrix.

Remarks. In the above, SVM is equivalent to M^4 when making two assumptions: one is the assumption about data "orientation" or data shape, i.e., $\mathbf{\Sigma}_{\mathbf{x}} = \mathbf{\Sigma}_{\mathbf{y}} = \mathbf{\Sigma}$, and the other is the assumption about data "scattering magnitude" or data compactness, i.e., $\mathbf{\Sigma} = \mathbf{I}$. However, these two assumptions are inappropriate. We demonstrate this in Figure 3 (a) and Figure 3 (b). We assume the orientation and the magnitude of each ellipsoid represent the data shape and compactness, respectively, in these figures.

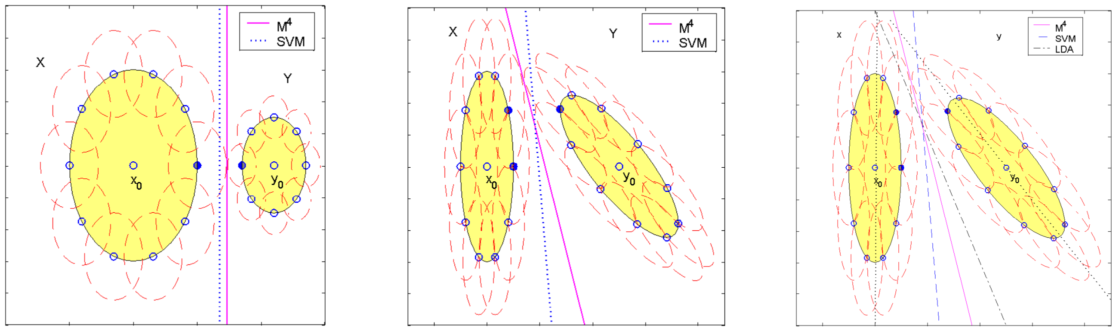
Figure 3 (a) plots two types of data with the same data orientations but different data scattering magnitudes. It is obvious that, by ignoring data scattering, SVM inappropriately locates itself unbiasedly midway between the support vectors (filled points), since $\mathbf{x}$ is likely to be scattered in the horizontal axis. Instead, M^4 is more reasonable (see the solid line in this figure). Furthermore, Figure 3 (b) plots the case with the same data scattering magnitudes but different data orientations. Similarly, SVM does not capture the orientation information. In contrast, M^4 employs this information and demonstrates a more suitable decision plane: M^4 represents the tangent line between two small dashed ellipsoids centered at the support vectors (filled points). Note that SVM and M^4 do not necessarily achieve the same support vectors. In Figure 3 (b), M^4 uses the two filled points mentioned above as support vectors, whereas SVM uses all three of the filled points as support vectors. It is also interesting that when $\mathbf{\Sigma}_{\mathbf{x}} = \mathbf{\Sigma}_{\mathbf{y}}$, SVM can produce the same results as M^4 with less computational time cost. However, it proves to be tough for SVM to consider different covariance matrices. Note that again, generally speaking, no single normalization can make two classes of data contain the same data trend simultaneously.

3) Link with Linear Discriminant Analysis: LDA, an important and popular method, is used widely in constructing decision hyperplanes [13][15] and reducing the feature dimensionality [12]. In the following discussion, we mainly consider its application as a classifier. LDA involves solving the following optimization problem:

$$\max_{\mathbf{w}\neq\mathbf{0}} \frac{|\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}})|}{\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w} + \mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}}.$$

Like MPM, LDA also focuses on using the global information rather than considering data both locally and globally. We now show that LDA can be modified to consider data both locally and globally.

If one changes the denominators in (2) and (3) to $\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w} + \mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}$, the optimization can be



(a) (b) (c)

Fig. 3. An illustration on the connections between SVM, LDA and M^4. (a) demonstrates SVM omits the data compactness information. (b) demonstrates SVM discards the data orientation information. (c) demonstrates LDA partly yet incompletely considers the data orientation.

changed to:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad \rho \quad s.t. \tag{16}$$

$$\frac{(\mathbf{w}^T\mathbf{x}_i + b)}{\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w} + \mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}} \geq \rho \, , \tag{17}$$

$$\frac{-(\mathbf{w}^T\mathbf{y}_j + b)}{\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w} + \mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}} \geq \rho \, , \tag{18}$$

where $i = 1, \ldots, N_{\mathbf{x}}$ and $j = 1, \ldots, N_{\mathbf{y}}$. The above optimization is a variant of LDA, which considers data locally and globally. This is verified as follows.

If one performs the procedure similar to that of Section II-B.1, the above optimization problem is easily verified to be the following optimization problem:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b} \quad \rho \quad s.t.$$

$$\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w} + \mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}. \tag{19}$$

One can change (19) to: $\rho \leq \frac{|\mathbf{w}^T(\overline{\mathbf{x}}-\overline{\mathbf{y}})|}{\sqrt{\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{x}}\mathbf{w}+\mathbf{w}^T\mathbf{\Sigma}_{\mathbf{y}}\mathbf{w}}}$, which is exactly the optimization of the LDA (note that $\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}})$ must have a positive value, from (17) and (18)).

Remarks. The extended LDA optimization focuses on considering the data orientation, while omitting the data scattering magnitude information. This can be shown from an analysis similar to that of Section II-B.2. Its decision hyperplane in the example of Figure 3 (a) coincides with that of SVM. With respect to the data orientation, it uses the average of the covariances for the two types of data. As illustrated in Figure 3 (c), the extended LDA corresponds to the line lying exactly in the middle of the long axes of the X and Y data. This shows that the extended LDA considers the data orientation partially yet incompletely.

C. Nonseparable Case

In this section, we modify the M^4 model to handle the nonseparable case. We need to introduce slack variables in this case. The optimization of M^4 is changed to:

$$\max_{\rho, \mathbf{w} \neq \mathbf{0}, b, \boldsymbol{\xi}} \quad \rho - C \sum_{k=1}^{N} \boldsymbol{\xi}_k \quad s.t. \tag{20}$$

$$(\mathbf{w}^T \mathbf{x}_i + b) \geq \rho \sqrt{\mathbf{w}^T \boldsymbol{\Sigma}_{\mathbf{x}} \mathbf{w}} - \boldsymbol{\xi}_i \ , \tag{21}$$

$$-(\mathbf{w}^T \mathbf{y}_j + b) \geq \rho \sqrt{\mathbf{w}^T \boldsymbol{\Sigma}_{\mathbf{y}} \mathbf{w}} - \boldsymbol{\xi}_{j+N_{\mathbf{x}}} \ , \tag{22}$$

$$\xi_k \geq 0 \ ,$$

where $i = 1, \ldots, N_{\mathbf{x}}$, $j = 1, \ldots, N_{\mathbf{y}}$, and $k = 1, \ldots, N$. C is the positive penalty parameter and $\boldsymbol{\xi}_k$ is the slack variable, which can be considered as the extent how the training point $\mathbf{z}_k$ disobeys the ρ margin ($\mathbf{z}_k = \mathbf{x}_k$ when $1 \leq k \leq N_{\mathbf{x}}$; $\mathbf{z}_k = \mathbf{y}_{k-N_{\mathbf{y}}}$ when $N_{\mathbf{x}} + 1 \leq k \leq N$). Thus $\sum_{k=1}^{N} \boldsymbol{\xi}_k$ can be conceptually regarded as the training error or the empirical error. In other words, the above optimization successfully maximizes the minimum margin while minimizing the total training error.

1) Method of Solution: As clearly observed, when ρ is fixed, the optimization is equivalent to minimizing $\sum_{k=1}^{N} \boldsymbol{\xi}_k$ under the same constraints. This is once again an SOCP problem and thus can be solved in polynomial time. We can then update ρ according to certain rules and repeat the whole process until an optimal ρ is found. This is also known as the so-called line search problem. More precisely, if we denote the value of optimization as a function $f(\rho) = \rho - C \sum_{k=1}^{N} \xi_k^{\rho}$, (where ξ_k^{ρ} is the associated optimal value for ξ_k when a specific ρ is set), the above procedure corresponds to finding an optimal ρ to maximize $f(\rho)$. Instead of using an explicit function as in traditional line search problems, the value of the function here is implicitly given by an SOCP procedure.

The line search problem can be solved by many methods. In this paper, we adopt the Quadratic Interpolation (QI) method, which converges superlinearly to a local optimum. Figure 4 illustrates the QI algorithm. QI searches for the maximum point by updating a three-point pattern (ρ_1, ρ_2, ρ_3) repeatedly. The new ρ denoted as ρ_{new} is given by the quadratic interpolation from the three-point pattern. Then a new three-point pattern is constructed by ρ_{new} and two of ρ_1, ρ_2, ρ_3.

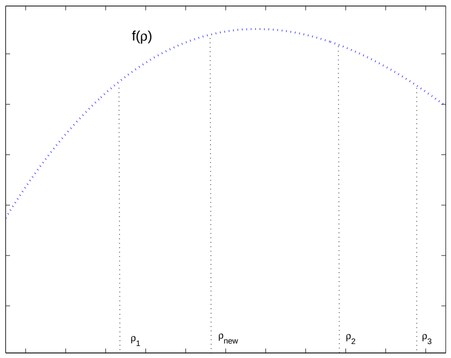


Fig. 4. A three-point pattern and Quadratic Line search method. A ρ_{new} is obtained and a new three-point pattern is constructed by ρ_{new} and two of ρ_1, ρ_2 and ρ_3.

In summary, we iterate the following two steps to solve the modified optimization.

Step 1.Generate a new ρ_n from three previous ρ_1, ρ_2, ρ_3 by using the Quadratic Interpolation method.

Step 2.Fixing $\rho = \rho_n$, perform the optimization based on SOCP algorithms. Then update ρ_1, ρ_2, ρ_3.

Remarks. Note that the above solution method is non-convex. The non-convexity is introduced by the coupled ρ and $\mathbf{w}$ in the term $\rho\sqrt{\mathbf{w}^T\mathbf{\Sigma_x}\mathbf{w}}$ and $\rho\sqrt{\mathbf{w}^T\mathbf{\Sigma_y}\mathbf{w}}$. Our proposed method can de-couple ρ and $\mathbf{w}$ by a sequential step successfully, and therefore provides a practical way to solve the optimization. Moreover, the final optimization of $f(\rho)$ is a one-dimensional line search problem with respect to ρ and can be globally optimized. Later experimental results on both toy and real data sets also demonstrate the effectiveness of this solving method.

D. Further Connection with Minimum Error Minimax Probability Machine

In this section, we show how the M^4 can be connected with the Minimum Error Minimax Probability Machine [6], which is a worst-case Bayes optimal classifier as well as a superset of MPM.

A careful consideration of the optimization of nonseparable M^4 shows that a more precise form is the one replacing $\boldsymbol{\xi}_k$ with $\boldsymbol{\xi}_k\sqrt{\mathbf{w}^T\mathbf{\Sigma_x}\mathbf{w}}$ in (21) and $\boldsymbol{\xi}_k\sqrt{\mathbf{w}^T\mathbf{\Sigma_y}\mathbf{w}}$ in (22). However, this optimization may prove to be a difficult problem, since slack variables ξ_i $(i = 1, 2, \ldots, N)$ are coupled with $\mathbf{w}^T$. Nevertheless, we can start from this precise form and derive the connection of M^4 with MEMPM.

We reformulate the optimization of (21)-(22) as their precise forms as follows:

$$\max_{\rho,\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\xi}} \quad \rho - C\sum_{k=1}^{N}\boldsymbol{\xi}_k \quad s.t. \tag{23}$$

$$\frac{\mathbf{w}^T\mathbf{x}_i + b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} \geq \rho - \boldsymbol{\xi}_i \;, \tag{24}$$

$$-\frac{\mathbf{w}^T\mathbf{y}_j + b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}}} \geq \rho - \boldsymbol{\xi}_{j+N_{\mathbf{x}}} \;, \tag{25}$$

$$\boldsymbol{\xi}_k \geq 0 \;, \tag{26}$$

where $i = 1,\ldots,N_{\mathbf{x}}$, $j = 1,\ldots,N_{\mathbf{y}}$, and $k = 1,\ldots,N$.

Maximizing (23) implies a similar meaning as minimizing $\frac{1}{\rho^2} + B\sum_{k=1}^{N}\boldsymbol{\xi}_k$ (B is a positive parameter) in the sense that they both attempt to maximize the margin ρ and minimize the error rate. If we consider $\sum_{k=1}^{N}\boldsymbol{\xi}_k$ as the residue and regard $\frac{1}{\rho^2}$ as the regularization term, the optimization can be cast into the framework of solving ill-posed problems.[8]

According to [24], [26], the above optimization, known as Tikhonov's Variation Method [22], is equivalent to the optimization below referred to Ivannov's Quasi-Solution Method [7], in the sense that if one of the methods for a given value of the parameter (say C) produces a solution $\{\mathbf{w}, b\}$, then the other method can derive the same solution by adapting its corresponding parameter (say A).

$$\min_{\rho,\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\xi}} \quad \sum_{k=1}^{N}\boldsymbol{\xi}_k \quad s.t.$$

$$\frac{\mathbf{w}^T\mathbf{x}_i + b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} \geq \rho - \boldsymbol{\xi}_i \;, \tag{27}$$

$$-\frac{\mathbf{w}^T\mathbf{y}_j + b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}}} \geq \rho - \boldsymbol{\xi}_{j+N_{\mathbf{x}}} \;, \tag{28}$$

$$\rho \geq A \;, \boldsymbol{\xi}_k \geq 0 \;, \tag{29}$$

where A is a positive constant parameter.

Now if we expand (27) for each i and add them all together, we can obtain:

$$N_{\mathbf{x}}\frac{\mathbf{w}^T\overline{\mathbf{x}} + b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} \geq N_{\mathbf{x}}\rho - \sum_{i=1}^{N_{\mathbf{x}}}\boldsymbol{\xi}_i \;. \tag{30}$$

This equation can easily be changed to:

$$\sum_{i=1}^{N_{\mathbf{x}}}\boldsymbol{\xi}_i \geq N_{\mathbf{x}}\rho - N_{\mathbf{x}}\frac{\mathbf{w}^T\overline{\mathbf{x}} + b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} \;. \tag{31}$$

[8]A trick can be made by assuming $\frac{1}{\rho}$ as a new variable and thus the condition that the regularization is convex can be satisfied.

Similarly, if we expand (28) for each j and add them all together, we obtain:

$$\sum_{j=1}^{N_{\mathbf{y}}} \boldsymbol{\xi}_{j+N_{\mathbf{x}}} \geq N_{\mathbf{y}}\rho + N_{\mathbf{y}}\frac{\mathbf{w}^T\overline{\mathbf{y}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}}} \; . \tag{32}$$

By adding (31) and (32), we can change the optimization to:

$$\min_{\rho,\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\xi}} \quad \sum_{k=1}^{N}\boldsymbol{\xi}_k \quad s.t.$$

$$\sum_{k=1}^{N}\boldsymbol{\xi}_k \geq N\rho - (N_{\mathbf{x}}\frac{\mathbf{w}^T\overline{\mathbf{x}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} - N_{\mathbf{y}}\frac{\mathbf{w}^T\overline{\mathbf{y}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}}}) \; . \tag{33}$$

To achieve the minimum residue, namely, $\min_{\rho,\mathbf{w}\neq\mathbf{0},b,\boldsymbol{\xi}} \quad \sum_{k=1}^{N}\boldsymbol{\xi}_k$, we may minimize the right hand side of (33).[9] Hence in this case ρ should attain its lower bound A, while the second part should be as large as possible, i.e.,

$$\max_{\mathbf{w}\neq\mathbf{0},b} \theta\frac{\mathbf{w}^T\overline{\mathbf{x}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} - (1-\theta)\frac{\mathbf{w}^T\overline{\mathbf{y}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}}} \; , \tag{34}$$

where θ is defined as $\frac{N_{\mathbf{x}}}{N}$ and thus $1-\theta$ denotes $\frac{N_{\mathbf{y}}}{N}$. If one further transforms the above to:

$$\max_{\mathbf{w}\neq\mathbf{0},b} \quad \theta t + (1-\theta)s \quad s.t. \tag{35}$$

$$\frac{\mathbf{w}^T\overline{\mathbf{x}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{x}}\mathbf{w}}} \geq t \; , \tag{36}$$

$$-\frac{\mathbf{w}^T\overline{\mathbf{y}}+b}{\sqrt{\mathbf{w}^T\boldsymbol{\Sigma}_{\mathbf{y}}\mathbf{w}}} \geq s \; , \tag{37}$$

one can see that the above optimization has a very similar form to the MEMPM model except that (35) changes to $\min_{\mathbf{w}\neq\mathbf{0},b}\theta\frac{t^2}{1+t^2}+(1-\theta)\frac{s^2}{1+s^2}$ [6]. In MEMPM, $\frac{t^2}{1+t^2}$ $(\frac{s^2}{1+s^2})$ (denoted as α (β)) represents the worst-case accuracy for the classification of future $X(Y)$ data. Thus MEMPM maximizes the weighted accuracy of the future data. In M^4, s and t represent the corresponding margin, which is defined as the distance from the hyperplane to the class center. Therefore, it represents the weighted maximum margin machine in this sense. Moreover, since the conversion function of $g(u)=\frac{u^2}{1+u^2}$ increases monotonically with u, maximizing the above formulae contains a physical meaning similar to the optimization of MEMPM in some sense.

[9] In fact, when the optimum is achieved, (33) will change to an equality in this optimization; otherwise, we can maintain the current optimum, say $\mathbf{w}_*$, b_*, ρ_* unchanged and decrease $\boldsymbol{\xi}_*$. The new solution will satisfy the constraints and will be a better solution. Therefore, this is contradictory with the statement that $\{\mathbf{w}_*, b_*, \rho_*, \boldsymbol{\xi}_*\}$ is the optimal solution.

III. Reduction

The variable in previous sections is $[\mathbf{w}, b, \xi_1, \ldots, \xi_{N_{\mathbf{x}}}, \ldots, \xi_N]$, whose dimension is $n+1+N$. The number of the second order conic constraints is easily verified to be N. This size of the generated constraint matrix will be large and we may thus encounter problems typically in solving large scale classification tasks. Therefore, we should reduce both the number of constraints and the number of variables.

Since this problem is caused by the number of the data points, we consider removing some redundant points to reduce both the space and time complexity. The reduction rule is introduced as follows:

Reduction Rule. Set a threshold $\nu \in [0,\ 1)$. In each class, calculate the Mahalanobis distance, d_i, of each point to its corresponding class center. If $d_i^2/(1+d_i^2)$, denoted as ν_i, is greater than ν, namely, $\nu_i \geq \nu$, keep this point; otherwise, remove this point.

The intuition underlying this rule is that, in general, the more discriminant information the point contains, the further it is from its center (unless it is a noise point). The inner justification under this rule is from [9]: $d^2/(1+d^2)$, is the worst-case classification accuracy for future data, where d is the minimax Mahalanobis distance from the class center to the decision hyperplane. Thus removing those points with small ν's, namely, $d_i^2/(1+d_i^2)$ will not affect the worst-case classification accuracy and will not greatly reduce the overall performance.

Nevertheless, to cancel the negative impact caused by removing those points, we add the following global constraint:

$$\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho(\sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{x}} \mathbf{w}} + \sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{y}} \mathbf{w}}). \tag{38}$$

This global constraint can be obtained by using the process described in Section II-B.1.

Integrating the above, we formulate the modified model as follows:

$$\begin{aligned}
&\max_{\rho, \mathbf{w} \neq \mathbf{0}, b, \xi} \{\rho - C(\sum_{k=1}^{r_{\mathbf{x}}+r_{\mathbf{y}}} \xi_k + (N - r_{\mathbf{x}} - r_{\mathbf{y}})\xi_m)\} \quad s.t. \\
&(\mathbf{w}^T \mathbf{x}_i + b) \geq \rho(\sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{x}} \mathbf{w}}) - \xi_i,\ i = 1, \ldots, r_{\mathbf{x}}\ , \\
&-(\mathbf{w}^T \mathbf{y}_j + b) \geq \rho(\sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{y}} \mathbf{w}}) - \xi_{j+r_{\mathbf{x}}},\ j = 1, \ldots, r_{\mathbf{y}}\ , \\
&\mathbf{w}^T(\overline{\mathbf{x}} - \overline{\mathbf{y}}) \geq \rho(\sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{x}} \mathbf{w}} + \sqrt{\mathbf{w}^T \mathbf{\Sigma}_{\mathbf{y}} \mathbf{w}}) - \xi_m\ , \\
&\xi_m \geq 0, \quad \xi_k \geq 0, k = 1, \ldots, r_{\mathbf{x}} + r_{\mathbf{y}}\ ,
\end{aligned} \tag{39}$$

where ξ_m is the slack variable for the global constraint (38), ξ_k are modified slack variables for the remaining data points, $r_{\mathbf{x}}$ is the number of the remaining points in X, and $r_{\mathbf{y}}$ is the number of the remaining points in Y. Note that, in the aforementioned optimization, since (38) is used to cancel negative impact caused by removing $N - r_{\mathbf{x}} - r_{\mathbf{y}}$ points, the slack variable ξ_m should be weighted by $N - r_{\mathbf{x}} - r_{\mathbf{y}}$ as indicated in (39).

Remarks. Two issues deserve our attentions. First, an interesting observation from the above is that, when we set the reduction threshold ν to a larger value, or simply to the maximum value 1, the M^4 optimization degrades to the standard MPM optimization. This would imply that the above modified M^4 model contains the worst-case performance of MPM, if the incorporated local information is useful. Second, although the worst-case probability is relatively safe for reducing data points, it is highly related to the Mahalanobis distance. When used in real applications especially in non-linear cases, the Mahalanobis distance measure in the input space cannot tightly represent the true probability belonging to the corresponding class [6]. Therefore, we may need to set the threshold ν carefully in this case.

IV. Kernelization

In the above discussion, the classifier derived from M^4 is provided in a linear configuration. In order to handle nonlinear classification problems, we seek to use the kernelization trick [18] to map the n-dimensional data points into a high-dimensional feature space $\mathbb{R}^f$, where a linear classifier corresponds to a nonlinear hyperplane in the original space. The implementation of this for our model is described below.

The kernel mapping can be formulated as: $\mathbf{x}_i \to \varphi(\mathbf{x}_i)$, $\mathbf{y}_j \to \varphi(\mathbf{y}_j)$, where $i = 1, \ldots, N_{\mathbf{x}}$, $j = 1, \ldots, N_{\mathbf{y}}$, and $\varphi : \mathbb{R}^n \to \mathbb{R}^f$ is a mapping function. The corresponding linear classifier in $\mathbb{R}^f$ is $\boldsymbol{\gamma}^T \varphi(\mathbf{z}) = b$, where $\boldsymbol{\gamma}$, $\varphi(\mathbf{z}) \in \mathbb{R}^f$, and $b \in \mathbb{R}$.

The optimization of M^4 in the feature space can be written as:

$$\max_{\rho, \boldsymbol{\gamma} \neq \mathbf{0}, b} \quad \rho \quad s.t. \tag{40}$$

$$\frac{(\boldsymbol{\gamma}^T \varphi(\mathbf{x}_i) + b)}{\sqrt{\boldsymbol{\gamma}^T \boldsymbol{\Sigma}_{\varphi(\mathbf{x})} \boldsymbol{\gamma}}} \geq \rho, \quad i = 1, 2, \ldots, N_{\mathbf{x}} \,, \tag{41}$$

$$\frac{-(\boldsymbol{\gamma}^T \varphi(\mathbf{y}_j) + b)}{\sqrt{\boldsymbol{\gamma}^T \boldsymbol{\Sigma}_{\varphi(\mathbf{y})} \boldsymbol{\gamma}}} \geq \rho, \quad j = 1, 2, \ldots, N_{\mathbf{y}} \,. \tag{42}$$

However, to make the kernel work, we need to represent the optimization and the final decision hyperplane into a kernel form, $\mathbf{K}(\mathbf{z}_1, \mathbf{z}_2) = \varphi(\mathbf{z}_1)^T \varphi(\mathbf{z}_2)$, namely, an inner product form of the mapping data points.

A. Foundation of Kernelization for M^4

In the following, we demonstrate that the kernelization trick works in M^4, provided suitable estimates of means and covariance matrices are applied therein.

Proposition 1: If the estimates of means and covariance matrices are given in M^4 as the following estimates:

$$\overline{\varphi(\mathbf{x})} = \sum_{i=1}^{N_{\mathbf{x}}} \lambda_i \varphi(\mathbf{x}_i), \quad \overline{\varphi(\mathbf{y})} = \sum_{j=1}^{N_{\mathbf{y}}} \omega_j \varphi(\mathbf{y}_j) \ ,$$

$$\Sigma_{\varphi(\mathbf{x})} = \rho_{\mathbf{x}} \mathbf{I}_n + \sum_{i=1}^{N_{\mathbf{x}}} \Lambda_i (\varphi(\mathbf{x}_i) - \overline{\varphi(\mathbf{x})})(\varphi(\mathbf{x}_i) - \overline{\varphi(\mathbf{x})})^T \ ,$$

$$\Sigma_{\varphi(\mathbf{y})} = \rho_{\mathbf{y}} \mathbf{I}_n + \sum_{j=1}^{N_{\mathbf{y}}} \Omega_j (\varphi(\mathbf{y}_j) - \overline{\varphi(\mathbf{y})})(\varphi(\mathbf{y}_j) - \overline{\varphi(\mathbf{y})})^T \ ,$$

then the optimal $\boldsymbol{\gamma}$ in (40)-(42) lies in the space spanned by the training points. In the above, positive constants $\rho_{\mathbf{x}}$ and $\rho_{\mathbf{y}}$ can be regarded as the regularization terms for the covariance matrices; $\mathbf{I}_n$ is the unit matrix of dimension n (n is the dimension of the feature space); Λ_i and Ω_j are the normalized weights for the sample $\mathbf{x}_i$ ($i = 1, 2, \ldots, N_{\mathbf{x}}$) and $\mathbf{y}_j$ ($j = 1, 2, \ldots, N_{\mathbf{y}}$) respectively.

Proof: We write $\boldsymbol{\gamma} = \boldsymbol{\gamma}_p + \boldsymbol{\gamma}_d$, where $\boldsymbol{\gamma}_p$ is the projection of $\boldsymbol{\gamma}$ in the vector space spanned by all the training data points and $\boldsymbol{\gamma}_d$ is the orthogonal component to this span space. By using $\boldsymbol{\gamma}_d^T \varphi(\mathbf{x}_i) = 0$ and $\boldsymbol{\gamma}_d^T \varphi(\mathbf{y}_j) = 0$, one can easily verify that the optimization (40)-(42) changes to:

$$\max_{\rho, \{\boldsymbol{\gamma}_p, \boldsymbol{\gamma}_d\} \neq \mathbf{0}, b} \quad \rho \quad \text{s.t.}$$

$$\frac{-(\boldsymbol{\gamma}_p^T \varphi(\mathbf{x}_i) + b)}{\sqrt{\boldsymbol{\gamma}_p^T \sum_{i=1}^{N_{\mathbf{x}}} \Lambda_i (\varphi(\mathbf{x}_j) - \overline{\varphi(\mathbf{x})})(\varphi(\mathbf{x}_i) - \overline{\varphi(\mathbf{x})})^T \boldsymbol{\gamma}_p + \rho_{\mathbf{x}} (\boldsymbol{\gamma}_p^T \boldsymbol{\gamma}_p + \boldsymbol{\gamma}_d^T \boldsymbol{\gamma}_d)}} \geq \rho \ ,$$

$$\frac{-(\boldsymbol{\gamma}_p^T \varphi(\mathbf{y}_j) + b)}{\sqrt{\boldsymbol{\gamma}_p^T \sum_{j=1}^{N_{\mathbf{y}}} \Omega_j (\varphi(\mathbf{y}_j) - \overline{\varphi(\mathbf{y})})(\varphi(\mathbf{y}_j) - \overline{\varphi(\mathbf{y})})^T \boldsymbol{\gamma}_p + \rho_{\mathbf{y}} (\boldsymbol{\gamma}_p^T \boldsymbol{\gamma}_p + \boldsymbol{\gamma}_d^T \boldsymbol{\gamma}_d)}} \geq \rho \ ,$$

where $i = 1, \ldots, N_{\mathbf{x}}$, $j = 1, \ldots, N_{\mathbf{y}}$. Since we intend to maximize the margin ρ, the denominators in the above two constraints need to be as small as possible. This would lead to $\boldsymbol{\gamma}_d = \mathbf{0}$. In other words, the optimal $\boldsymbol{\gamma}$ lies in the vector space spanned by all the training data points. Note that the above discussion refers to the feature space. ■

According to Proposition 1, if we use the plug-in estimates to approximate the means and covariance matrices, we can write $\boldsymbol{\gamma}$ as a linear combination form of training data points:

$$\boldsymbol{\gamma} = \sum_{i=1}^{N_{\mathbf{x}}} \mu_i \varphi(\mathbf{x}_i) + \sum_{j=1}^{N_{\mathbf{y}}} \upsilon_j \varphi(\mathbf{y}_j) \ , \tag{43}$$

where the coefficients $\mu_i,\, \upsilon_j \in \mathbb{R}, i = 1, \dots, N_{\mathbf{x}},\, j = 1, \dots, N_{\mathbf{y}}$.

B. The Kernelization Result

Before we present the main kernelization result, we first introduce the notations. Let $\{\mathbf{z}\}_{i=1}^{N}$ denote all $N = N$ data points in the training set, where

$$\mathbf{z}_i = \mathbf{x}_i \qquad\qquad i = 1, 2, \dots, N_{\mathbf{x}}\ ,$$

$$\mathbf{z}_i = \mathbf{y}_{i-N_{\mathbf{x}}} \qquad i = N_{\mathbf{x}} + 1, N_{\mathbf{x}} + 2, \dots, N.$$

The element of the Gram matrix $\mathbf{K}$ in the position of (i, j) is defined as $\mathbf{K}_{i,j} = \varphi(\mathbf{z}_i)^T \varphi(\mathbf{z}_j)$ for $i, j = 1, 2, \dots, N$. We define $\mathbf{K}_i$ as the i-th column vector of the $\mathbf{K}$. We further define $\mathbf{K}_{\mathbf{x}}$ and $\mathbf{K}_{\mathbf{y}}$ as the matrices formed by the first $N_{\mathbf{x}}$ rows and the last $N_{\mathbf{y}}$ rows of $\mathbf{K}$, respectively. In other words,

$$\mathbf{K} := \begin{pmatrix} \mathbf{K}_{\mathbf{x}} \\ \mathbf{K}_{\mathbf{y}} \end{pmatrix}.$$

By setting the row average of the $\mathbf{K}_{\mathbf{x}}$ block and the $\mathbf{K}_{\mathbf{y}}$ block to zero, the block-row-averaged Gram matrix $\tilde{\mathbf{K}}$ is thus obtained:

$$\mathbf{K} := \begin{pmatrix} \tilde{\mathbf{K}}_{\mathbf{x}} \\ \tilde{\mathbf{K}}_{\mathbf{y}} \end{pmatrix} = \begin{pmatrix} \mathbf{K}_{\mathbf{x}} - \mathbf{1}_{N_{\mathbf{x}}} \tilde{\mathbf{k}}_{\mathbf{x}}^T \\ \mathbf{K}_{\mathbf{y}} - \mathbf{1}_{N_{\mathbf{y}}} \tilde{\mathbf{k}}_{\mathbf{y}}^T \end{pmatrix}\ ,$$

where $\tilde{\mathbf{k}}_{\mathbf{x}}, \tilde{\mathbf{k}}_{\mathbf{y}} \in \mathbb{R}^N$ are defined as:

$$[\tilde{\mathbf{k}}_{\mathbf{x}}]_i := \frac{1}{N_{\mathbf{x}}} \sum_{j=1}^{N_{\mathbf{x}}} \mathbf{K}(\mathbf{x}_j, \mathbf{z}_i)\ ,$$

$$[\tilde{\mathbf{k}}_{\mathbf{y}}]_i := \frac{1}{N_{\mathbf{y}}} \sum_{j=1}^{N_{\mathbf{y}}} \mathbf{K}(\mathbf{y}_j, \mathbf{z}_i)\ .$$

In the above, $\mathbf{1}_{N_{\mathbf{x}}} \in \mathbb{R}^{N_{\mathbf{x}}}$ and $\mathbf{1}_{N_{\mathbf{y}}} \in \mathbb{R}^{N_{\mathbf{y}}}$, which are defined as:

$$\mathbf{1}_i = 1, \quad i = 1, 2, \dots N_{\mathbf{x}}\ ,$$

$$\mathbf{1}_j = 1, \quad j = 1, 2, \dots N_{\mathbf{y}}\ .$$

Finally, we define the vector formed by the coefficients of $\boldsymbol{\gamma}$ as

$$\boldsymbol{\eta} = [\mu_1, \mu_2, \dots, \mu_{N_{\mathbf{x}}}, \upsilon_1, \upsilon_2, \dots, \upsilon_{N_{\mathbf{y}}}]^T. \tag{44}$$

We present the kernelization result as the following theorem.

Kernelization Theorem of M^4 *The optimal decision hyperplane for* M^4 *involves solving the following optimization problem:*

$$\max_{\rho, \boldsymbol{\eta} \neq \mathbf{0}, b} \quad \rho \quad s.t.$$

$$\frac{(\boldsymbol{\eta}^T \mathbf{K}_i + b)}{\sqrt{\frac{1}{N_{\mathbf{x}}} \boldsymbol{\eta}^T \tilde{\mathbf{K}}_{\mathbf{x}}^T \tilde{\mathbf{K}}_{\mathbf{x}} \boldsymbol{\eta}}} \geq \rho, \quad i = 1, 2, \ldots, N_{\mathbf{x}} ,$$

$$\frac{-(\boldsymbol{\eta}^T \mathbf{K}_{j+N_{\mathbf{x}}} + b)}{\sqrt{\frac{1}{N_{\mathbf{y}}} \boldsymbol{\eta}^T \tilde{\mathbf{K}}_{\mathbf{y}}^T \tilde{\mathbf{K}}_{\mathbf{y}} \boldsymbol{\eta}}} \geq \rho, \quad j = 1, 2, \ldots, N_{\mathbf{y}} .$$

Proof: The theorem can easily be proved by simply substituting the plug-in estimates of the means and covariance matrices and (43) into (40-42). ■

The optimal decision hyperplane can be represented as a linear form in the kernel space

$$f(\mathbf{z}) = \sum_{i=1}^{N_{\mathbf{x}}} \boldsymbol{\eta}_{*i} \mathbf{K}(\mathbf{z}, \mathbf{x}_i) + \sum_{i=1}^{N_{\mathbf{y}}} \boldsymbol{\eta}_{*N_{\mathbf{x}}+i} \mathbf{K}(\mathbf{z}, \mathbf{y}_i) + b_* ,$$

where $\boldsymbol{\eta}_*$ and b_* are the optimal parameters obtained by the optimization procedure.

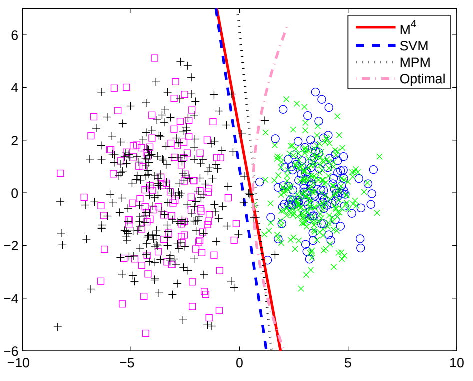
V. EXPERIMENTS

In this section, we present the evaluation results of M^4 in comparison with SVM and MPM on both synthetic toy data sets and real world benchmark data sets. SOCP problems are solved based on the general software named Sedumi [19], [20]. The covariance matrices are given by the plug-in estimates.

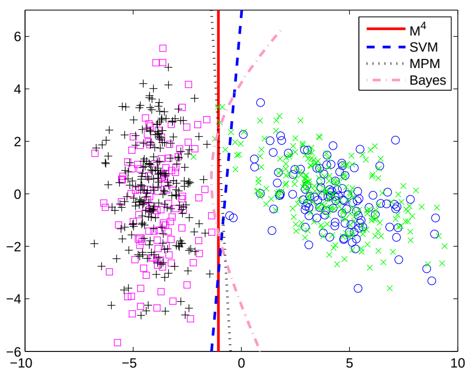
A. Evaluations on Four Synthetic Toy Data Sets

We demonstrate the advantages of our approach in comparison with SVM and MPM in the following synthetic toy data sets first.

As illustrated in Figure 5, we generate two types of data with the same data orientations but different data magnitudes in Figure 5 (a), and another two types of data with the same data magnitudes but different data orientations in Figure 5 (b). In (a), the X data are randomly sampled from the Gaussian distribution with the mean as $[3.5,\ 0]^T$ and a covariance of $[1,\ 0; 0,\ 1.5]$, while the Y data are randomly sampled from another Gaussian distribution with the mean and the covariance as $[-3.5,\ 0]^T$ and $[3,\ 0; 0,\ 4.5]$ respectively. In (b), the $\mathbf{x}$ data are randomly sampled from the Gaussian distribution with the mean as $[4,\ 0]^T$ and the covariance as $[1,\ 0; 0,\ 5]$, while the Y data are randomly sampled from another distribution with the mean and the covariance as $[-4,\ 0]^T$ and $[1,\ 0; 0,\ 5]$ respectively. Moreover, to

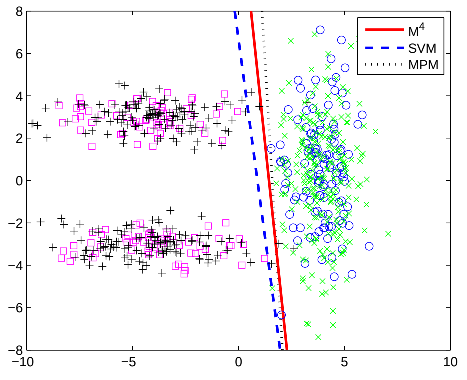


(a)

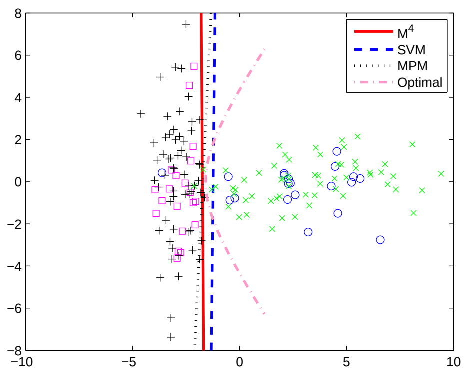


(b)

Fig. 5. The first two synthetic toy examples to illustrate M^4. Training (test) data, consisting of 120 (250) data points for each class are presented as $\circ$'s($\times$'s) and $\square$'s (+'s) for $\mathbf{x}$ and $\mathbf{y}$ respectively. Subfigure (a) demonstrates that SVM omits the data compactness information and (b) demonstrates that SVM discards the data orientation information, while the decision boundary of M^4 considers data both locally and globally.



(a)



(b)

Fig. 6. The third and fourth synthetic toy examples to illustrate M^4. Training (test) data, consisting of 120 (250) data points in (a) and 20 (60) in (b) for each class are presented as $\circ$'s($\times$'s) and $\square$'s (+'s) for $\mathbf{x}$ and $\mathbf{y}$ respectively. Subfigure (a) demonstrates M^4 performs better in a mixture of Gaussian data than SVM, while (b) shows that the M^4 achieves the suitable decision boundary, which considers data both locally and globally.

generate different data orientations, in this figure, the X data are rotated anti-clockwise at the angle of $-\frac{3}{8}\pi$. In both (a) and (b), training (test) data, consisting of 120 (250) data points for each class, are presented as $\circ$'s($\times$'s) and $\Box$'s (+'s) for X and Y respectively. Observed from Figure 5, M^4 demonstrates its advantages over SVM. More specifically, in Figure 5 (a), SVM discards the information of the data magnitudes, whose decision hyperplane lies basically in the middle of boundary points of two types of data, while M^4 successfully utilizes this information, i.e., its decision hyperplane lies closer to the compact class (X data), which is more reasonable. Similarly, in Figure 5 (b), M^4 takes advantage of the information of the data orientation, while SVM simply overlooks this information, resulting in a misclassification of many points.

In Figure 6 (a), we further generate two types of data. The X data are randomly sampled from the Gaussian distribution with the mean as $[4,\ 0]$ and the covariance as $[1,\ 0; 0,\ 8]$; the Y data are described by a mixture of two Gaussian data with the means as $[-4, -3]^T$ and $[-4,\ 3]^T$ respectively and both covariances as $[5,\ 0; 0,\ 0.4]^T$. Training (testing) data are 120 (250) random samples for each category resented as $\circ$'s ($\times$'s) and $\Box$'s (+'s) for X and Y respectively. As seen in this figure, the derived decision boundary of M^4 once again demonstrates a more appropriate separating plane than the SVM.

When MPM is compared with M^4, they achieve similar performance. This is because the global information, i.e., the mean and the covariance, can be reliably estimated from the data in the two data sets. To see the difference between M^4 and MPM, we generate another data set as illustrated in Figure 6 (b), where we intentionally produce a very small number of training data, i.e., only 20 training points. Similarly, the data are generated under two Gaussian distributions: The X data are randomly sampled from the Gaussian distribution with the mean as $[4,\ 0]^T$ and the covariance as $[6,\ 0; 0,\ 1]$, while the Y data are randomly sampled from another distribution with the mean and the covariance as $[-3,\ 0]^T$ and $[0.5,\ 0; 0,\ 8]$ respectively. Training data and test data are represented using symbols similar to Figure 6 (a). From Figure 6 (b), once again M^4 achieves a suitable decision boundary, which considers data both locally and globally; in contrast, SVM obtains the local boundary that is simply midway between the support vectors, thus discarding the global information, namely the statistical "trend" of data occurrence. The decision hyperplane of MPM is exclusively dependent on the mean and covariance matrices. Thus we can see that this hyperplane coincides with the data shape, i.e., the long axis of training data of Y is nearly in the same direction as the MPM decision hyperplane. However, the estimated mean and covariance is inaccurate due to the small number of traning data points. This results in a relatively lower test accuracy as illustrated in Figure 6 (b). In comparison, M^4 incorporates the information of the local points to neutralize the effect caused by inaccurate estimation.

In the above, we also plot the optimal decision boundaries by directly using the Gaussian distribution functions as seen in Figure 5 and Figure 6 (b). As observed, our M^4 boundaries are closer to the optimal lines than those of SVM. The test accuracies for the above three toy data sets listed in Table I further demonstrate the advantages of M^4.

Dataset	M^4	SVM	MPM
I(%)	**98.8**	96.8	**98.8**
II(%)	**98.8**	97.2	**98.8**
III(%)	**98.8**	97.2	**98.8**
IV(%)	**98.3**	97.5	95.8

TABLE I

COMPARISONS OF CLASSIFICATION ACCURACIES BETWEEN M^4, SVM, AND MPM ON THE TOY DATA SETS.

B. Evaluations on Benchmark Data Sets

We perform evaluations on seven standard data sets. Data for the Twonorm problem were synthetically generated according to [3]. The remaining six data sets were real world data obtained from the UCI machine learning repository [2]. We compared M^4 with SVM and MPM, engaging both the linear and Gaussian kernels. The parameter C for both M^4 and SVM was tuned via cross validation in the inner loop of 10-fold cross validation [8], as was the width parameter in the Gaussian kernel for all three models. The final performance were given with the $10-$fold cross validation results. Table II summarizes the evaluation results.

From the results, we observe that M^4 achieves the best overall performance. In comparison with SVM and MPM, M^4 wins five cases in the linear kernel and four in the Gaussian kernel. The evaluations on these standard benchmark data sets demonstrate that it is worth considering data both locally and globally, which is emphasized in M^4. Inspecting the differences between M^4 with SVM, the kernelized M^4 appears marginally better than the kernelized SVM, while the linear M^4 demonstrates a distinct advantage over the linear SVM. Moreover, we conduct the t-test at the significance level 0.05. The results show that M^4 is significantly better than SVM and MPM in Breast, Ionosphere, Sonar, Vote, and Heart-disease in the linear case, while it is only significant better than SVM and MPM in Breast, Ionosphere, Vote, and Heart-disease in the Gaussian kernel case. On the one hand, this can be explained from the fact that the data points are very sparse in the kernelized space or feature space (compared with

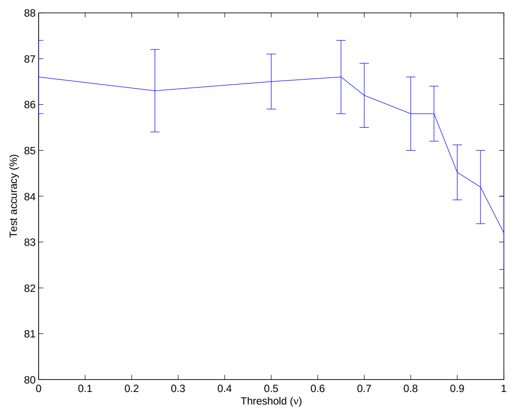
Data set	Linear kernel			Gaussian kernel		
	M^4	SVM	MPM	M^4	SVM	MPM
Twonorm(%)	96.5 ± 0.6	95.1 ± 0.7	$\mathbf{97.6 \pm 0.5}$	96.5 ± 0.7	96.1 ± 0.4	$\mathbf{97.6 \pm 0.5}$
Breast(%)	$\mathbf{97.5 \pm 0.7}$	96.6 ± 0.5	96.9 ± 0.8	$\mathbf{97.5 \pm 0.6}$	96.7 ± 0.4	96.9 ± 0.8
Ionosphere(%)	$\mathbf{87.7 \pm 0.8}$	86.9 ± 0.6	84.8 ± 0.8	$\mathbf{94.5 \pm 0.4}$	94.2 ± 0.3	92.3 ± 0.6
Pima(%)	77.7 ± 0.9	$\mathbf{77.9 \pm 0.7}$	76.1 ± 1.2	77.6 ± 0.8	$\mathbf{78.0 \pm 0.5}$	76.2 ± 1.2
Sonar(%)	$\mathbf{77.6 \pm 1.2}$	76.2 ± 1.1	75.5 ± 1.1	84.9 ± 1.2	86.5 ± 1.1	$\mathbf{87.3 \pm 0.8}$
Vote(%)	$\mathbf{96.1 \pm 0.5}$	95.1 ± 0.4	94.8 ± 0.4	$\mathbf{96.2 \pm 0.5}$	95.9 ± 0.6	94.6 ± 0.4
Heart-disease(%)	$\mathbf{86.6 \pm 0.8}$	84.1 ± 0.7	83.2 ± 0.8	$\mathbf{86.2 \pm 0.8}$	83.8 ± 0.5	83.1 ± 1.0

TABLE II

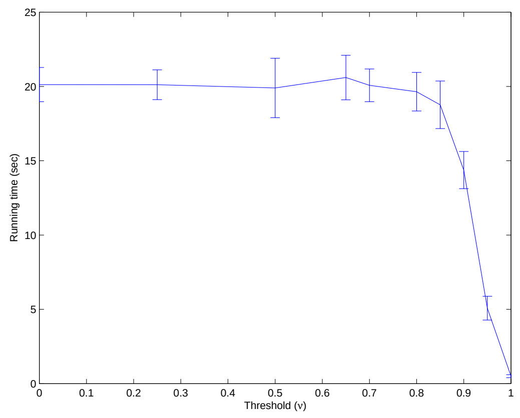
COMPARISONS OF CLASSIFICATION ACCURACIES AMONG M^4, SVM, AND MPM.

the huge dimensionality in the Gaussian kernel). Thus the plug-in estimates of the covariance matrices may not accurately represent the data information in this case. On the other hand, it is still uncertain whether the kernelization will not precisely keep the structure information in the feature space. One possible consequence is that maximizing the margin in the feature space does not necessarily maximize the margin in the original space [23]. Therefore, unless some connections are built between the original space and the feature space, utilizing the structure information, e.g., covariance matrices in the feature space, does not seem to help in this sense. By examining these two viewpoints, one interesting topic for future study is to consider forcing constraints on the mapping function so as to maintain the data topology in the kernelization process.

In the above experiments, we do not perform reduction on these data sets. To illustrate how well the reduction algorithm works at decreasing the computational time while maintaining the test accuracy, we implement it on the Heart-disease data set. We perform the reduction on training sets and keep the test sets unchanged. We repeat this process for different thresholds ν. We then plot the curve of the cross-validation accuracy against the threshold ν. Moreover, we also plot the curve of the computational time against the threshold. This can be seen in Figure 7. From this figure, we can see both the computational time and the test accuracy show little dependency on ν when ν is set to small values, e.g., $\nu \leq 0.7$. If looking at the Heart-disease data set, we find that most data points are far from their corresponding class center in terms of the Mahalanobis distance. Thus, setting small values of ν does not eliminate many data points. This generates a relatively flat curve with respect to both the test accuracy and the computational time in this range. As ν increases, the computational time decreases fast as more and more data points are removed, while the test accuracy goes down slowly. When the threshold is set to 1, the M^4 degrades



(a)



(b)

Fig. 7. Reduction on the Heart-disease data set.This figures demonstrates how the proposed reduction algorithms can decrease the computational time while maintaining good performance.

to the MPM model, yielding the same test accuracy between M^4 and MPM. This demonstrates how the proposed reduction algorithms can decrease the computational time while maintaining good performance. When applied in practice, the threshold can be set according to the required response time.

VI. Future Work and Open Problems

We will discuss several important issues in this section. First, one important future direction is to further develop special, and efficient optimization methods for the proposed model. Although M^4 can be solved in polynomial time $O(Nn^3)$, the computational time overhead is still its main shortcoming. In particular, when compared with the time complexity of $O(n^3 + Nn^2)$ in solving MPM or the quadratic programs of SVM, the computational cost may present a big challenge for large-scale data sets. Moreover, although we have proposed a reduction algorithm in this paper, removing points will inevitably discard useful information. In this sense, it is crucial to develop some special algorithms for M^4. Due to the sparsity of M^4 (which also contains support vectors), it is therefore very interesting to investigate whether decomposable methods, or an analogy to the Sequential Minimal Optimization [16] designed for SVM, can also be applied in training M^4. We believe there is much to gain from such explorations.

Second, since both SVM and MPM come with a generalization error bound, it is interesting whether a bound can be derived from their related model M^4? This interesting subject deserves future explorations.

Third, currently, the covariances used in this model are given by the plug-in estimations. When the number of the input data points are small, plug-in estimations may not be reliable or accurate. This would influence the performance of the proposed model. Therefore how to estimate the covariances robustly and reliably presents an important research topic.

Finally, since in this paper we mainly discuss M^4 for two-category classifications, determining how to extend its application to multi-way classifications is also an important future topic.

VII. Conclusion

Large margin machines have demonstrated their advantages in machine learning and pattern recognition. Many types of these machines learn their decision boundaries based on only either a global or a local view of data. For example, the most popular large margin classifier, the Support Vector Machine, obtains the decision hyperplane by focusing on considering some critical local points called support vectors, while discarding all other points; on the other hand, another significant model, the Minimax Probability Machine, uses only the global information, i.e., the mean and covariance information from data, while ignoring all individual local points. As a distinct contribution, our proposed model is constructed based on both a local and a global view of data. This new model is theoretically important in the sense that SVM and MPM show close relationship with it. Furthermore, the optimization of M^4 can be cast as a sequential conic programming problem, which can be solved in polynomial time.

We have provided a clear geometrical interpretation, and established detailed connections among our model and other models such as the Support Vector Machine, the Minimax Probability Machine, the Linear Discriminant Analysis, and the Minimum Error Minimax Probability Machine. We have also extended our model by exploiting Mercer kernels in building up nonlinear decision boundaries. In addition, we have proposed a reduction method to decrease the computational time. Experimental results on both synthetic data sets and real world benchmark data sets have demonstrated the advantages of M^4 over the Support Vector Machine and the Minimax Probability Machine.

Acknowledgements

The work described in this paper was fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project CUHK4205/04E and CUHK4235/04E). The authors are grateful to the anonymous reviewers for their value comments.

REFERENCES

[1] D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, Massachusetts, 2nd edition, 1999.

[2] C.L. Blake and C.J. Merz. UCI repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences, 1998. URL: http://www.ics.uci.edu/∼mlearn/MLRepository.html.

[3] L. Breiman. Arcing classifiers. Technical Report 460, Statistics Department, University of California, 1997.

[4] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, San Diego, 2nd edition, 1990.

[5] K. Huang, H. Yang, I. King, and M. R. Lyu. Learning large margin classifiers locally and globally. In Russ Greiner and Dale Schuurmans, editors, *The Twenty-first International Conference on Machine Learning (ICML-2004)*, pages 401–408, 2004.

[6] K. Huang, H. Yang, I. King, M. R. Lyu, and L. Chan. The minimum error minimax probability machine. *Journal of Machine Learning Research*, 5:1253–1286, 2004.

[7] V. V. Ivannov. On linear problems which are not well-posed. *Soviet Math. Docl.*, 3(4):981–983, 1962.

[8] R. Kohavi. A study of cross validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the fourteenth International Joint Conference on Artificial Intelligence (IJCAI-1995)*, pages 338–345. San Francisco, CA: Morgan Kaufmann, 1995.

[9] G. R. G. Lanckriet, L. E. Ghaoui, C. Bhattacharyya, and M. I. Jordan. A robust minimax approach to classification. *Journal of Machine Learning Research*, 3:555–582, 2002.

[10] Chun-Fu Lin and Sheng-De Wang. Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13:464–471, 2002.

[11] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[12] R. Lotlikar and R. Kothari. Fractional-step dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:623–627, 2000.

[13] J. Lu, K.N. Plataniotis, and A.N. Venetsanopoulos. Face recognition using LDA-based algorithms. *IEEE Transactions on Neural Networks*, 14:195– 200, 2003.

[14] Y. Nesterov and A. Nemirovsky. *Interior point polynomial methods in convex programming: Theory and applications*. Studies in Applied Mathematics. Philadelphia, 1994.

[15] J. Peng, D.R. Heisterkamp, and H.K. Dai. Ldasvm driven nearest neighbor classification. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 58–63, 2001.

[16] J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. *Technical Report MSR-TR-98-14*, 1998.

[17] A. Pruessner. Conic programming in GAMS. In *Optimization Software - The State of the Art*. INFORMS Atlanta, http://www.gamsworld.org/cone/links.htm, 2003.

[18] B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[19] J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11:625–653, 1999.

[20] J. F. Sturm. Central region method. In J. B. G. Frenk, C. Roos, T. Terlaky, and S. Zhang, editors, *High Performance Optimization*, pages 157–194. Kluwer Academic Publishers, 2000.

[21] J.A.K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor. A support vector machine formulation to pca analysis and its kernel version. *IEEE Transactions on Neural Networks*, 14:447– 450, 2003.

[22] A. N Tikhonov. On solving ill-posed problem and method of regularization. *Doklady Akademii Nauk USSR*, 153:501–504, 1963.

[23] S. Tong and D. Koller. Restricted bayes optimal classifiers. In *Proceedings of the seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, pages 658–664, 2000.

[24] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, 1998.

[25] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 2nd edition, 1999.

[26] V. V. Vasin. Relationship of several variational methods for approximate solutions of ill-posed problems. *Math. Notes*, 7:161–166, 1970.

[27] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Advances in Neural Information Processing Systems (NIPS 16)*, 2003.