Journal of Data Science 21 (4), 658–680
October 2023

DOI: 10.6339/22-JDS1069
Statistical Data Science

Linear Algorithms for Robust and Scalable Nonparametric Multiclass Probability Estimation

Liyun Zeng[1] and Hao Helen Zhang[1,2,*]

[1] *Statistics and Data Science GIDP, University of Arizona, Tucson, Arizona, USA*
[2] *Department of Mathematics, University of Arizona, Tucson, Arizona, USA*

Abstract

Multiclass probability estimation is the problem of estimating conditional probabilities of a data point belonging to a class given its covariate information. It has broad applications in statistical analysis and data science. Recently a class of weighted Support Vector Machines (wSVMs) has been developed to estimate class probabilities through ensemble learning for K-class problems (Wu et al., 2010; Wang et al., 2019), where K is the number of classes. The estimators are robust and achieve high accuracy for probability estimation, but their learning is implemented through pairwise coupling, which demands polynomial time in K. In this paper, we propose two new learning schemes, the baseline learning and the One-vs-All (OVA) learning, to further improve wSVMs in terms of computational efficiency and estimation accuracy. In particular, the baseline learning has optimal computational complexity in the sense that it is linear in K. Though not the most efficient in computation, the OVA is found to have the best estimation accuracy among all the procedures under comparison. The resulting estimators are distribution-free and shown to be consistent. We further conduct extensive numerical experiments to demonstrate their finite sample performance.

Keywords *linear time algorithm; multiclass classification; non-parametric; probability estimation; scalability; support vector machines*

1 Introduction

In machine learning and pattern recognition, the goal of multiclass classification is to assign a data point to one of K classes based on its input, where $K \geqslant 3$. In specific, we observe the pair $(\boldsymbol{X}, Y)$ from an unknown distribution $P(\boldsymbol{x}, y)$, where $\boldsymbol{X} \in \mathbb{R}^p$ is the input vector and $Y \in \mathcal{A} = \{1, 2, \dots, K\}$ indicates class membership, and the task is to learn a discriminant rule $\phi : \mathbb{R}^p \to \mathcal{A}$ for making predictions on future data. Define the conditional class probabilities as $p_j(\boldsymbol{x}) = P(y = j|\boldsymbol{X} = \boldsymbol{x})$ for $j \in \{1, \dots, K\}$. For any classifier ϕ, its prediction performance can be evaluated by its risk $R(\phi) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}[I(y \neq \phi(\boldsymbol{x}))] = P(y \neq \phi(\boldsymbol{x}))$, also called the expected prediction error (EPE). The optimal rule minimizing the risk is the *Bayes* classifier, given by:

$$\phi_B(\boldsymbol{x}) = \arg \max_{j=1,\dots,K} p_j(\boldsymbol{x}). \tag{1}$$

For classification purposes, although it is sufficient to learn the argmax rule only, the estimation of class probabilities is usually desired as they can quantify uncertainty about the

[*] Corresponding author. Email: hzhang@math.arizona.edu.

Received June 3, 2022; Accepted September 25, 2022

classification decision and provide additional confidence. Probability estimation is useful in a broad range of applications. For example, in medical decision making such as cancer diagnosis, class probabilities offer an informative measurement of the reliability of the disease classification (Hastie et al., 2009). In e-commerce advertising and recommendation systems, accurate estimation of class probabilities enables businesses to focus on the most promising alternatives and therefore optimize budget to maximize profit gain and minimize loss. Moreover, for unbalanced problems, class probabilities can help users construct optimal decision rules for cost-sensitive classification, where different types of misclassification errors may incur different costs (Herbei and Wegkamp, 2006).

In literature, there are a variety of methods for probability estimation, and commonly used methods include linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), and logistic regression. These methods typically make certain assumptions on statistic models or data distributions (McCullagh and Nelder, 1989). Recently, multiclass SVMs (Vapnik, 1998; Weston and Watkins, 1999; Crammer and Singer, 2001; Lee et al., 2004; Wang and Shen, 2007; Liu, 2007; Liu and Yuan, 2011; Zhang and Liu, 2013; Huang et al., 2013; Lei et al., 2015) have shown great advantage in many real-world applications, such as cancer diagnosis, hypothyroid detection, handwritten digit recognition, spam detection, speech recognition, and online news classification (Burges, 1998; Cristianini and Shawe-Taylor, 2000; Zhu et al., 2003; Chamasemani and Singh, 2011; Mezzoudj and Benyettou, 2012; Saigal and Khanna, 2020). However, standard SVMs can not estimate class probabilities. To tackle this limitation, the concept of binary weighted SVMs (wSVMs) is proposed by Wang et al. (2008) to learn class probabilities via ensemble learning. In particular, the binary wSVM methods assign different weights to data from two classes, train a series of classifiers from the weighted loss function, and aggregate multiple decision rules to construct the class probabilities. The wSVM methods are model-free, hence being both flexible and robust. Wu et al. (2010) extend the wSVM from $K = 2$ to $K \geqslant 3$ by learning multiple probability functions simultaneously, which has nice theoretical properties and empirical performance but demands computational time exponentially in K, making it impracticable in many real-world applications with $K \geqslant 4$. Recently, Wang et al. (2019) propose the divide-and-conquer learning approach via pairwise coupling to reduce the computational cost of Wu et al. (2010) from exponential to polynomial time in K, but it is still not optimal for complex problems with a large K, say, $K \geqslant 5$.

In this article, we propose new estimation schemes and computational algorithms to further improve Wang et al. (2019) in terms of its computation and estimation accuracy. The first algorithm uses the idea of "baseline learning", which can reduce the computational time of wSVMs from polynomial to linear complexity in K, and it is therefore scalable for learning massive data sets with a large number of classes K and sample points n. The second scheme employs the One-vs-All strategy to improve wSVMs in terms of probability estimation accuracy. Both methods can be formulated as Quadratic Programming (QP), and solved by popular optimization packages in R, Python, MATLAB, and C/C++ *etc.* We also provide rigorous analysis on their computational complexity. In addition, Their divide-and-conquer nature can further speed up their implementation via parallel computing by GPU and multi-core parallel computing, high-performance computing (HPC) clusters, and massively parallel computing (MPP).

The rest of the paper is organized as follows. Section 2 introduces the notations reviews the multiclass wSVMs framework. Section 3 proposes the baseline learning scheme and shows its computational and theoretical properties. Section 4 introduces the One-vs-All learning scheme and presents the computational algorithm and Fisher consistency results. Section 5 conducts a rigorous analysis of the computational complexity of the new learning schemes and algorithms.

We show that the baseline learning method enjoys the optimal complexity, with computational time linear in the number of classes K. Section 6 discusses the issue of hyperparameter tuning. Sections 7 and 8 are contributed to numerical studies by performing extensive simulated and real-world data analysis and comparison. Section 9 presents concluding remarks and discussions.

2 Notations and Review of Weighted SVMs

2.1 Notations

Denote the data points by $\{(\boldsymbol{x}_i, y_i), i = 1, \dots, n\}$, where $\boldsymbol{x}_i = (x_{i1}, \dots, x_{ip})^{\mathrm{T}} \in \mathbb{R}^p$, n is the sample size, and p is the dimension of input $\boldsymbol{x}$. There are K classes, with $K \geqslant 3$. For $j = 1, \dots, K$, denote the subset of data points belonging to class j by $\mathcal{S}_j = \{(\boldsymbol{x}_i, y_i) \mid y_i = j;\ i = 1, \dots, n\}$, and the corresponding sample size by $n_j = |\mathcal{S}_j|$. Denote the multiclass learner trained from the data by $\hat{\boldsymbol{f}} = \{\hat{f}_1, \dots, \hat{f}_K\}$, where $\hat{f}_j(\boldsymbol{x})$ represents evidence of a data point with input $\boldsymbol{x}$ belonging to class j. The classifier is constructed using the argmax rule, i.e., $\hat{\phi}(\boldsymbol{x}) = \arg\max_{1 \leqslant j \leqslant K} \hat{f}_j(\boldsymbol{x})$.

2.2 Binary Weighted SVMs

In binary class problems, the class label $y \in \{+1, -1\}$. Define the conditional class probabilities $p_{+1}(\boldsymbol{x}) = P(Y = +1 | \boldsymbol{X} = \boldsymbol{x})$ and $p_{-1}(\boldsymbol{x}) = P(Y = -1 | \boldsymbol{X} = \boldsymbol{x})$ for class $+1$ and -1, respectively. The standard SVM learns the decision function $f \in \mathcal{F}$ by solving the regularization problem:

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} L(y_i f(\boldsymbol{x}_i)) + \lambda J(f), \tag{2}$$

where $L(yf(\boldsymbol{x})) = [1 - yf(\boldsymbol{x})]_+ = \max\{0, 1 - yf(\boldsymbol{x})\}$ is the hinge loss, $\mathcal{F}$ is some functional space, the regularization term $J(f)$ controls model complexity, and the tuning parameter $\lambda > 0$ balances the bias–variance tradeoff (Hastie et al., 2009). For linear SVMs, $f(\boldsymbol{x}) = \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{x}$. For kernel SVMs, f employs a bivariate kernel representation form $f(\boldsymbol{x}) = d + \sum_{i=1}^{n} c_i \mathbf{K}(\boldsymbol{x}_i, \boldsymbol{x})$, according to the representer theorem (Kimeldorf and Wahba, 1971), where $\mathbf{K}(\cdot, \cdot)$ is a Mercer kernel, $\mathcal{F} = \mathcal{H}_{\mathbf{K}}$ is the reproducing kernel Hilbert space (RKHS, Wahba, 1990) induced by $\mathbf{K}(\cdot, \cdot)$, and $J(f) = \|\boldsymbol{f}\|^2_{\mathcal{H}_{\mathbf{K}}} = \sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j \mathbf{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Then the optimization problem (2) amounts to:

$$\min_{d, c_1, \dots, c_n} \frac{1}{n} \sum_{i=1}^{n} L(y_i f(\boldsymbol{x}_i)) + \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j \mathbf{K}(\boldsymbol{x}_i, \boldsymbol{x}_j), \quad \text{where } f(\boldsymbol{x}) = d + \sum_{i=1}^{n} c_i \mathbf{K}(\boldsymbol{x}_i, \boldsymbol{x}). \tag{3}$$

It is known that the minimizer of the expected hinge loss $\mathbb{E}_{(\boldsymbol{X},Y) \sim P(\boldsymbol{X},Y)}[1 - Yf(\boldsymbol{X})]_+$ has the same sign as the Bayes decision boundary $\mathrm{sign}[p_{+1}(\boldsymbol{x}) - \frac{1}{2}]$ (Lin, 2002), therefore binary SVMs directly target on the optimal Bayes classifier without estimating $p_{+1}(\boldsymbol{x})$.

Despite their success in many real-world classification problems, standard SVMs can not estimate the conditional class probabilities $p_{+1}(\boldsymbol{x})$ and $p_{-1}(\boldsymbol{x})$. To overcome this limitation, binary weighted SVMs (wSVMs, Wang et al., 2008) were proposed to learn a series of classifiers by minimizing the weighted hinge loss function, and construct probabilities by aggregating multiple decision rules. In particular, by assigning the weight π $(0 \leqslant \pi \leqslant 1)$ to data points from class -1 and assigning $1 - \pi$ to those from class $+1$, the wSVM solves the following optimization problem:

$$\min_{f \in \mathcal{H}_{\mathbf{K}}} \frac{1}{n} \Big[(1 - \pi) \sum_{y_i = 1} L(y_i f(\boldsymbol{x}_i)) + \pi \sum_{y_i = -1} L(y_i f(\boldsymbol{x}_i)) \Big] + \lambda J(f). \tag{4}$$

For any π, the minimizer of the expected weighted hinge loss $\mathbb{E}_{(\boldsymbol{X},Y)\sim P(\boldsymbol{X},Y)}\{W(Y)L[Yf(\boldsymbol{X})]\}$ has the same sign as $\text{sign}[p_{+1}(\boldsymbol{X})-\pi]$ (Wang et al., 2008). To estimate the class probabilities, we first obtain multiple classifiers $\hat{f}_{\pi_2}, \ldots, \hat{f}_{\pi_M}$ by solving (4) with a series of values $0=\pi_1<\pi_2<\cdots<\pi_M<\pi_{M+1}=1$. Given any point $\boldsymbol{x}$, the values $\hat{f}_\pi(\boldsymbol{x})$ is non-increasing in π, so there exists a unique $m^* \in \{1,\ldots,M\}$ such that $\pi_{m^*} < p_{+1}(\boldsymbol{x}) < \pi_{m^*+1}$. Consequently, we can construct a consistent probability estimator by $\hat{p}_{+1}(\boldsymbol{x}) = \frac{1}{2}(\pi_{m^*}+\pi_{m^*+1})$. The numerical precision level of $\hat{p}_{+1}(\boldsymbol{x})$ is determined by M, which controls the fineness of the grid points π's.

2.3 Multiclass Weighted SVMs

Assume $Y = y \in \{1,2,\ldots,K\}$ with $K \geqslant 3$. The goal is to estimate class probabilities $p_j(\boldsymbol{x}) = P(Y=j|\boldsymbol{X}=\boldsymbol{x})$ for $j \in \{1,\ldots,K\}$. Wu et al. (2010) extended wSVMs from $K=2$ to $K \geqslant 3$ via a joint estimation approach. However, its computational complexity is exponential in K, which makes the procedure infeasible for problems with a large K and a large n. Recently Wang et al. (2019) proposed the pairwise coupling method which decomposes the K-class problem into $\binom{K}{2}$ binary classification problems and enjoys a reduced computational complexity, polynomial in K. For each class pair (j,k) with $1 \leqslant j \neq k \leqslant K$, the pairwise coupling first learns the pairwise conditional probability $\hat{q}_{j|(j,k)}$ by training binary wSVMs, and then aggregates the decision rules from all pairs to construct the class probabilities estimator as:

$$\hat{p}_j(\boldsymbol{x}) = \frac{\hat{q}_{j|(j,k)}(\boldsymbol{x})/\hat{q}_{k|(j,k)}(\boldsymbol{x})}{\sum_{s=1}^{K}\hat{q}_{s|(s,k)}(\boldsymbol{x})/\hat{q}_{k|(s,k)}(\boldsymbol{x})}, \quad j=1,\ldots,K; \quad k\neq j. \tag{5}$$

Class prediction is done by either the argmax rule, i.e., $\hat{y} = \arg\max_{1\leqslant j\leqslant K}\hat{p}_j(\boldsymbol{x})$, or the max voting algorithm (Kallas et al., 2012; Tomar and Agarwal, 2015; Ding et al., 2019). Wang et al. (2019) showed that the wSVMs can outperform benchmark methods like kernel multi-category logistic regression (Zhu and Hastie, 2005), random forest, and classification trees.

3 New Learning Scheme: Baseline Learning

3.1 Methodology

The pairwise coupling method (Wang et al., 2019) reduces the computational complexity of Wu et al. (2010) from exponential to polynomial in K. The procedure works well for a moderate K, but not feasible for a much larger K. To speed up, we propose a more efficient learning scheme, the baseline learning. We show that the computational cost of the baseline learning is linear in K, making it scalable for classification problems with a large K.

In pairwise coupling, we optimize the algorithm (discussed in Section 7) to choose adaptively the best baseline class for each data point. It requires training all class pairs (j,k) with $j \neq k$ and fitting $\binom{K}{2}$ binary wSVMs problems. To speed up, we propose using a common baseline class k^* and training only $K-1$ binary problems for (j,k^*) with $j \neq k^*$, call the procedure as baseline learning. The rational behind the baseline learning is that, the final class probability estimator does not depend on the choice of the baseline class as shown in (5) (Wang et al., 2019). In Section 5, the baseline learning is shown to enjoy computational complexity linear in K, which is the lowest among all the wSVMs methods. After choosing k^*, we train $K-1$ binary wSVMs problems and obtain $K-1$ pairwise conditional probabilities $\hat{q}_{j|(j,k^*)}$, for $j \neq k^*$ and

$j = 1, \ldots, K$. Assuming $\hat{q}_{k^*|(k^*,k^*)} = 1$, the final class probability estimators are then given by:

$$\hat{p}_j^{bs}(\boldsymbol{x}) = \frac{\hat{q}_{j|(j,k^*)}(\boldsymbol{x})/\hat{q}_{k^*|(j,k^*)}(\boldsymbol{x})}{\sum_{s=1}^{K} \hat{q}_{s|(s,k^*)}(\boldsymbol{x})/\hat{q}_{k^*|(s,k^*)}(\boldsymbol{x})}, \quad j = 1, \ldots, K. \tag{6}$$

The choice of k^* is critical to optimize finite performance of the baseline learning. To assure numerical stability, we need to choose k^* such that the denominators $\hat{q}_{k^*|(j,k^*)}$ of the ratios in (6) are kept away from zero simultaneously for all the classes j with $j \neq k^*$. Towards this, we propose the following two methods of choosing k^* and then evaluate their finite sample performance in numerical studies.

Choice of k^* (Method 1) We propose choosing k^* to be the largest subclass in the training set, i.e., $k^* = \arg\max_{1\leqslant j\leqslant K} |\mathcal{S}_j|$. With this choice, the quantity $\hat{q}_{k^*|(j,k^*)}$ is relatively large among all possible choices and hence stablizes the ratios in (6). This method has been used in other contexts of one-class and multi-class SVMs classification tasks and found computationally robust (Krawczyk et al., 2014). We will refer to the baseline learning wSVMs with this choice as B1-SVM.

Choice of k^* (Method 2) Alternatively, we propose choosing k^* to be the class not too far from any other class, in order to prevent $\hat{q}_{k^*|(j,k^*)}$ in (6) from being too close to zero for any $j \neq k^*$. Towards this, we would look for the class which sits at the "middle" location of all the classes. For each class j, we compute its within-class compactness $\mathcal{D}_{cp}$ and between-class distances $\mathcal{D}_{bc}$ (Islam et al., 2016) to another class j', construct an "aggregated" distance $\mathcal{D}_{agg}(j)$ to measure its overall distance to all the remaining classes. The class k^* is chosen based on the median of $\mathcal{D}_{agg}(j)$'s.

For each class j, we compute its within-class compactness $\mathcal{D}_{cp}(j)$ using $\mathcal{S}_j$ as follows: (1) for each point $\boldsymbol{x}_t \in \mathcal{S}_j$, $1 \leqslant t \leqslant |\mathcal{S}_j|$, calculate the sum of its Euclidean distances to all other data points in class j, i.e. $d_t = \sum_{\forall \boldsymbol{x}_s \in \mathcal{S}_j \setminus \{\boldsymbol{x}_t\}} \|\boldsymbol{x}_s - \boldsymbol{x}_t\|_2$; (2) identify the median point $\tilde{\boldsymbol{x}}_m$, corresponding to the median of $\{d_1, \ldots, d_{|\mathcal{S}_j|}\}$; (3) calculate the maximum distance of the points in class j to the median, i.e., $\mathrm{dm}(j) = \max_{\boldsymbol{x}_s \in \mathcal{S}_j} \|\boldsymbol{x}_s - \tilde{\boldsymbol{x}}_m\|_2$; (4) define $\mathcal{D}_{cp}(j) = \mathrm{dm}(j)$. For each pair $(j, j'), 1 \leqslant j \neq j' \leqslant K$, their between-class distance $\mathcal{D}_{bc}(j, j')$ is the minimum distance between the boundary points of the two classes. In specific, it is equal to the minimum value of the Euclidean distances of data points from class j to those in class j', i.e., $\mathcal{D}_{bc}(j, j') = \min_{\forall \boldsymbol{x}_s \in \{\mathcal{S}_j\} \times \forall \boldsymbol{x}_t \in \{\mathcal{S}_{j'}\}} \|\boldsymbol{x}_s - \boldsymbol{x}_t\|_2$. Finally, the aggregated class distance $\mathcal{D}_{agg}(j)$ is defined as

$$\mathcal{D}_{agg}(j) = \frac{1}{K} \frac{\sum_{\{j'|1\leqslant j' \neq j \leqslant K\}} \mathcal{D}_{bc}(j, j')}{\mathcal{D}_{cp}(j)}. \tag{7}$$

Define $k^* = \arg \mathrm{median}(\{\mathcal{D}_{agg}(j) \mid 1 \leqslant j \leqslant K\})$. This wSVMs learner is referred to as B2-SVM.

3.2 Computational Algorithm

To implement the baseline learning, we carry out the following steps: (1) choose the common baseline class k^*; (2) decompose the K-class classification problem into $K - 1$ binary problems; (3) for each j, train the binary wSVMs for the pair (j, k^*) and obtain the pairwise conditional probability estimator $\hat{q}_{j|(j,k^*)}$; (4) compute $\hat{p}_j^{bs}(\boldsymbol{x})$ using (6) for each $j \neq k^*$. The following is the detailed algorithm:

Step 1: Choose the baseline class k^*, using either Method 1 (B1-SVM) or Method 2 (B2-SVM).

Step 2: For each $j \neq k^*$, define a function $R_{j,k^*}(y) = 1$ if $y = j$; $R_{j,k^*}(y) = -1$ if $y = k^*$. Fit the kernel binary wSVMs for the class pair (j, k^*) by solving the optimization problem:

$$\min_{f \in \mathcal{H}_K} \frac{1}{n}\Big[(1-\pi_m)\sum_{y_i=j} L(R_{j,k^*}(y_i)f(\boldsymbol{x}_i)) + \pi_m \sum_{y_i=k^*} L(R_{j,k^*}(y_i)f(\boldsymbol{x}_i))\Big] + \lambda J(f) \qquad (8)$$

over a series $0 < \pi_2 < \cdots < \pi_M < 1$. For each $m = 2, \ldots, M$, denote the optimized classifier of (8) as $\hat{f}_{j,k^*,\pi_m}(\cdot)$. Furthermore, we define $\pi_1 = 0$, $\pi_{M+1} = 1$, and assume $\hat{f}_{j,k^*,\pi_m}(\cdot) > 0$ if $m = 1$ and $\hat{f}_{j,k^*,\pi_m}(\cdot) < 0$ if $m = M+1$.

Step 3: For each class pair (j, k^*), calculate the pairwise conditional probability estimator as:

$$\hat{q}_{j|(j,k^*)}(\boldsymbol{x}) = \frac{1}{2}\left[\arg\min_{\pi_m}\{\hat{f}_{j,k^*,\pi_m}(\boldsymbol{x}) < 0\} + \arg\max_{\pi_m}\{\hat{f}_{j,k^*,\pi_m}(\boldsymbol{x}) > 0\}\right], \quad \forall \boldsymbol{x} \in \mathcal{S}. \qquad (9)$$

Step 4: Calculate the Bayesian posterior class probabilities estimator as:

$$\begin{aligned} \hat{p}_j^{bs}(\boldsymbol{x}) &= \frac{\hat{q}_{j|(j,k^*)}(\boldsymbol{x})/\hat{q}_{k^*|(j,k^*)}(\boldsymbol{x})}{\sum_{s=1}^{K}\hat{q}_{s|(s,k^*)}(\boldsymbol{x})/\hat{q}_{k^*|(s,k^*)}(\boldsymbol{x})}, \quad j = \{1, \ldots, K\} \setminus \{k^*\}, \\ \hat{p}_{k^*}^{bs}(\boldsymbol{x}) &= 1 - \sum_{\{j, 1 \leqslant j \neq k^* \leqslant K\}} \hat{p}_j^{bs}(\boldsymbol{x}). \end{aligned} \qquad (10)$$

Since the baseline learning trains only $K-1$ binary wSVMs, compared to $\binom{K}{2}$ wSVMs classifiers in the pairwise coupling method, its computational saving is substantial for a large K. In oder to obtain optimal result, for now, the regularization parameter $\lambda > 0$ needs to be tuned adaptively using the data at Step 2. In Section 6, we will discuss the parameter tuning in details.

3.3 Theory

We now study asymptotic properties of the conditional probability estimators given by the baseline learning (6) and establish its Fisher consistency. At Step 2, since the baseline learning solves the sub-problems of the pairwise coupling, Lemma 2 in Wang et al. (2019) holds in our case as well. Therefore, for any fixed j and π, the global minimizer of the weighted hinge loss corresponding to (8) for pair (j, k^*) is a consistent estimator of $q_{j|(j,k^*)}(\boldsymbol{X}) - \pi$. When the sample size $n \to \infty$ and the functional space $\mathcal{F}$ is sufficiently rich, with a properly chosen λ, the solution to (8), $\hat{q}_{j|(j,k^*)}(\boldsymbol{x})$, will converge to $q_{j|(j,k^*)}(\boldsymbol{x})$ asymptotically. Theorem 1 below shows that the estimator from the baseline learning wSVM, given by (10), is asymptotically consistent under certain regularity conditions. Proof of Theorem 1 is similar to those in Wang and Shen (2006) and Wang et al. (2008) and hence omitted.

Theorem 1. *Consider the K-class classification problem with kernel weighted SVMs. Denote the class probabilities estimator as $\hat{p}_j^{bs}(\boldsymbol{x})$, $j \in \{1, \ldots, K\} \setminus \{k^*\}$ and $\hat{p}_{k^*}^{bs}(\boldsymbol{x})$ from (10), where k^* is the predefined baseline class. Define the series of weights $0 = \pi_1 < \pi_2 < \cdots < \pi_M < \pi_{M+1} = 1$, and their maximum interval $\theta_\pi = \max_{\epsilon \in \{1,\ldots,M\}}\{\Delta\pi_\epsilon \mid \Delta\pi_\epsilon = \pi_{\epsilon+1} - \pi_\epsilon\}$. Then the estimators are consistent, i.e., $\hat{p}_j^{bs}(\boldsymbol{x}) \to p_j(\boldsymbol{x})$ for $j \in \{1, \ldots, K\}$ if $\lambda \to 0$, $\theta_\pi \to 0$, and $n \to \infty$.*

3.4 Enhanced Pairwise Coupling By Baseline Learning

Interestingly, the idea of baseline learning can be also used to enhance performance of the original pairwise coupling method in Wang et al. (2019) by improving its computation time and estimation accuracy. Recall that, the pairwise coupling requires training $\binom{K}{2}$ binary wSVMs to obtain $\hat{q}_{j|(j,j')}(\boldsymbol{x})$ for all possible (j, j') pairs, which is computationally expensive. In the following, we use the baseline learning solutions $\{\hat{q}_{j|(j,k^*)}(\boldsymbol{x}), j \neq k^*\}$ to mathematically derive $\hat{q}_{j|(j,j')}(\boldsymbol{x})$ without actually solving the optimization problems.

For any $\boldsymbol{x}$, we would like to calculate $\hat{q}_{j|(j,j')}(\boldsymbol{x})$ with $\{(j, j') \mid 1 \leqslant (j, j') \neq k^* \wedge (j < j') \leqslant K\}$. Based on the definition of pairwise conditional probability in Wang et al. (2019), we have:

$$\hat{q}_{j|(j,k^*)}(\boldsymbol{x}) = \frac{\hat{p}_j(\boldsymbol{x})}{\hat{p}_j(\boldsymbol{x}) + \hat{p}_{k^*}(\boldsymbol{x})}, \quad \hat{q}_{j'|(j',k^*)}(\boldsymbol{x}) = \frac{\hat{p}_{j'}(\boldsymbol{x})}{\hat{p}_{j'}(\boldsymbol{x}) + \hat{p}_{k^*}(\boldsymbol{x})}, \quad \hat{q}_{j|(j,j')}(\boldsymbol{x}) = \frac{\hat{p}_j(\boldsymbol{x})}{\hat{p}_j(\boldsymbol{x}) + \hat{p}_{j'}(\boldsymbol{x})}.$$

Therefore, from $\hat{q}_{j|(j,k^*)}(\boldsymbol{x})$ and $\hat{q}_{j'|(j',k^*)}(\boldsymbol{x})$, we have the following relationship

$$\hat{p}_{k^*}(\boldsymbol{x}) = \frac{\hat{p}_j(\boldsymbol{x})}{\hat{q}_{j|(j,k^*)}(\boldsymbol{x})} - \hat{p}_j(\boldsymbol{x}) = \frac{\hat{p}_{j'}(\boldsymbol{x})}{\hat{q}_{j'|(j',k^*)}(\boldsymbol{x})} - \hat{p}_{j'}(\boldsymbol{x}).$$

By applying simple algebraic transformation, we can estimate $q_{j|(j,j')}$ as follows:

$$\hat{q}_{j|(j,j')}(\boldsymbol{x}) = \frac{\hat{q}_{j|(j,k^*)}(\boldsymbol{x}) - \hat{q}_{j|(j,k^*)}(\boldsymbol{x})\hat{q}_{j'|(j',k^*)}(\boldsymbol{x})}{\hat{q}_{j|(j,k^*)}(\boldsymbol{x}) + \hat{q}_{j'|(j',k^*)}(\boldsymbol{x}) - 2\hat{q}_{j|(j,k^*)}(\boldsymbol{x})\hat{q}_{j'|(j',k^*)}(\boldsymbol{x})}. \tag{11}$$

After reconstructing the pairwise conditional probabilities, we can estimate the class probabilities by dynamically selecting the best baseline class k for each data point, by implementing the optimization algorithm of pairwise coupling method discussed in Section 7, which enjoys the linear time complexity. This new type of pairwise coupling enhanced by baseline learning, using k^* chosen either by Method 1 or Method 2, is denoted as BP1-SVM and BP2-SVM, respectively. In Sections 7 and 8, we will evaluate their performance and make comparisons with the enhanced implementation of the pairwise coupling and the baseline learning.

4 New Learning Scheme: One-vs-All Learning (OVA-SVM)

4.1 Methodology

Both of our enhanced implementation of the pairwise coupling in Wang et al. (2019) and the baseline learning require the choice of a baseline class k, either adaptively for each data point or using a common baseline class k^*. In this section, we develop another scheme which does not require choosing the baseline class at all. The new method employs the One-vs-All (OVA) idea, a simple yet effective divide-and-conquer technique for decomposing a large multiclass problem into multiple smaller and simpler binary problems. The OVA has shown competitive performance in real-world applications (Rifkin and Klautau, 2004).

The new procedure is proposed as follows. First, we decompose the multiclass problem into K binary problems by training trains class j vs class“not j”, where class “not j” combines all the data points not belonging to class j. Second, we fit a weighted SVMs binary classifier separately for the above binary problem for each $j = 1, \ldots, K$. Third, we aggregate all the decision rules to construct the class probability estimators for p_j’s. We refer this method as OVA-SVM, which is simple to implement and has robust performance. Since the OVA-SVM needs to train unbalanced binary problems, involved with the full data set, its computational cost is higher than the baseline learning. We will perform complexity analysis in Section 5.

4.2 Computation Algorithm

The following is the algorithm to implement the OVA-SVM:

Step 1: For each j, create class "not j" by combining all the data points not in class j; label the class as A_j.

Step 2: Define a function $R_{j,A_j}(y) = 1$ if $y = j$; and $R_{j,A_j}(y) = -1$ if $y = A_j$. Fit the kernel binary wSVMs for the class pair (j, A_j) by solving the optimization problem:

$$\min_{f \in \mathcal{H}_K} \frac{1}{n}\Big[(1-\pi_m)\sum_{y_i = j} L(R_{j,A_j}(y_i) f(\boldsymbol{x}_i)) + \pi_m \sum_{y_i = A_j} L(R_{j,A_j}(y_i) f(\boldsymbol{x}_i))\Big] + \lambda J(f) \quad (12)$$

over a series $0 < \pi_2 < \cdots < \pi_M < 1$. Denote the solution to (12) as $\hat{f}_{j,A_j,\pi_m}(\cdot)$ for each j and m. Furthermore, assume $\hat{f}_{j,k^*,\pi_1=0}(\cdot) > 0$ and $\hat{f}_{j,k^*,\pi_{M+1}=1}(\cdot) < 0$.

Step 3: For each class j, compute its estimated class probability as:

$$\hat{p}_j(\boldsymbol{x}) = \frac{1}{2}\left[\arg\min_{\pi_m}\{\hat{f}_{j,A_j,\pi_m}(\boldsymbol{x}) < 0\} + \arg\max_{\pi_m}\{\hat{f}_{j,A_j,\pi_m}(\boldsymbol{x}) > 0\}\right], \quad \forall \boldsymbol{x} \in \mathcal{S}. \quad (13)$$

Step 4: Repeat Steps 1–3 and obtain $\hat{p}_j(\boldsymbol{x})$ for $j = 1, \ldots, K$.

To predict the class label for any given input $\boldsymbol{x}$, we apply $\hat{y} = \arg\max_{1 \leqslant j \leqslant K} \hat{p}_j(\boldsymbol{x})$. At Step 2, λ needs to selected adaptively for optimal performance. We will discuss the parameter tuning strategy in Section 6.

4.3 Theory

Wang et al. (2008) show that the binary wSVMs probability estimator is asymptotically consistent, i.e., $\hat{p}_{+1}(\boldsymbol{x})$ asymptotically converges to $p_{+1}(\boldsymbol{x})$ as $n \to \infty$, when the functional space $\mathcal{F}$ is sufficiently rich and λ is properly chosen. For the OVA-SVM approach, the class probability estimator $\hat{p}_j(\boldsymbol{x})$ for each class $j \in \{1, \ldots, K\}$ is solely determined by training the classifier $\hat{f}_{j,A_j,\pi_m}(\cdot)$ by solving (12). Theorem 2 below shows the OVA-SVM probability estimators given by (13) are asymptotically consistent for the true class probabilities $p_j(\boldsymbol{x})$'s. The proof is similar to that of Theorems 2 and 3 in Wang et al. (2008) and thus omitted.

Theorem 2. *Consider the K-class classification problem using the kernel weighted SVM. Denote the class probabilities estimator from (13) as $\hat{p}_j(\boldsymbol{x})$ for $j \in \{1, \ldots, K\}$. Define the series of weights $0 = \pi_1 < \pi_2 < \cdots < \pi_M < \pi_{M+1} = 1$, and their maximum interval $\theta_\pi = \max_{\epsilon \in \{1,\ldots,M\}}\{\Delta\pi_\epsilon \mid \Delta\pi_\epsilon = \pi_{\epsilon+1} - \pi_\epsilon\}$. Then the estimators are consistent, i.e., $\hat{p}_j(\boldsymbol{x}) \to p_j(\boldsymbol{x})$ for $j = \{1, \ldots, K\}$ if $\lambda \to 0$, $\theta_\pi \to 0$, and the sample size $n \to \infty$.*

5 Complexity Analysis

We perform rigorous analysis on the computational complexity of a variety of divide-and-conquer learning schemes for multiclass wSVMs, including the pairwise coupling (Wang et al., 2019), the proposed baseline learning in Section 3, and the proposed OVA-SVM in Section 4.

Without loss of generality, assume the K classes are balanced, i.e., $n_j = n$ for any $1 \leqslant j \leqslant K$. When we randomly split the data as the training and tuning sets, we also keep the class size balanced. For each class pair $\{(j, j') \mid 1 \leqslant j \neq j' \leqslant K\}$, denote the training set and the tuning set as $\mathbb{S}_{train}$ and $\mathbb{S}_{tune}$, and their corresponding sample size as n_{train} and n_{tune}, respectively.

For each binary wSVMs classifier, the optimization problem is solved by Quadratic Programming (QP) using the training set $\mathbb{S}_{train}$. In R, we employ the `quadprog` package, which uses the active set method and has the polynomial time complexity $\sim O(n_{train}^3)$ (Ye and Tse, 1989; Cairano et al., 2013). The time complexity for label prediction on the tuning set of size n_{tune} by the weighted SVMs classifier is $O(n_{tune}n_{train})$. To train $O(m)$ sequence of wSVMs for pairwise class probability estimation with n_{tune} data points, the time complexity is: $O(m)[O(n_{train}^3) + O(n_{tune}n_{train})] + n_{tune}O(m\log m)$. During the parameter tuning step, let n_{par} be the number of tuning parameters and $\{\gamma_p \mid p \in \{1,\dots,n_{par}\}\}$ be the number of grid points. Then the time complexity for computing the pairwise conditional probability estimators is:

$$\left(\prod_{p=1}^{n_{par}} \gamma_p\right)\left[O(m)(O(n_{train}^3) + O(n_{train}n_{tune})) + n_{tune}O(m\log m)\right]. \tag{14}$$

In the following, we focus on the RBF kernel with two hyperparameters $\{\lambda,\sigma\}$, and the number of grid points are $\gamma_\lambda = \gamma_1$ and $\gamma_\sigma = \gamma_2$. We have m as constant and $n_{train} = n_{tune} = n$. Then (14) is reduced as $O(\gamma_1\gamma_2 n^3)$. For the pairwise coupling method, we need to compute $O(K^2)$ pairwise conditional probability estimators, so the total time complexity is $O(\gamma_1\gamma_2K^2n^3)$.

For the baseline learning, we first choose the baseline class k^*, then compute $O(K)$ pairwise conditional class probabilities. The B1-SVM method chooses k^* as the largest class, and its time complexity is $O(Kn) + O(\gamma_1\gamma_2Kn^3) = O(\gamma_1\gamma_2Kn^3)$. The B2-SVM method selects k^* by measuring the median class distance. Since the time complexity of computing K-class $\mathcal{D}_{cp}$ and $\mathcal{D}_{bc}$ is $O(Kn^2)$, we need $O(Kn^2)$ to obtain k^*. Then the time complexity for B2-SVM is $O(Kn^2)+ O(\gamma_1\gamma_2Kn^3) = O(\gamma_1\gamma_2Kn^3)$, which is the same as B1-SVM.

Now we consider the time complexity of the One-vs-All method (OVA-SVM), where we calculate $O(K)$ pairwise conditional class probabilities. Assume that $n_{train} = n_{tune} = 0.5Kn$. From (14), we have $O(\gamma_1\gamma_2K^3n^3)$, and so the total time complexity for OVA-SVM is $O(\gamma_1\gamma_2K^4n^3)$.

In summary, for K-class problems with the wSVMs, for fixed γ_1, γ_2, m, and n, the time complexity for the baseline learning methods (B1-SVM and B2-SVM) is $\propto O(K)$, for the pairwise coupling is $\propto O(K^2)$, and for OVA-SVM is $\propto O(K^4)$. The proposed baseline method is linear in K, which has the lowest computational cost among all of the divide-and-conquer wSVM methods. The above analysis is based on a balanced scenario. For the unbalanced case, the analysis is more complex mathematically, but the conclusion remains the same.

6 Parameter Tuning

The proposed kernel learning method involves two tuning parameters, $\lambda > 0$ and $\sigma \in I\!R$ (if an RBF kernel is used). Their values are critical to assure the optimal finite sample performance of the wSVMs. To choose proper values for these tuning parameters, we randomly split the data into two equal parts: the training set $\mathbb{S}_{train}$ used to train the wSVM at (8) or (12) for a given weight π and fixed (λ,σ), and the tuning set used to select the optimal values of $\{\lambda,\sigma\}$.

For a fixed $\{\lambda,\sigma\}$, we train $M-1$ classifiers $\hat{f}_{\pi_\epsilon}^{\lambda,\sigma}(\boldsymbol{x})$ for $\pi_\epsilon \in \{\frac{j-1}{M} \mid j = 2,\dots,M\}$ with $\mathbb{S}_{train}$ to obtain the pairwise conditional probability estimators $\hat{q}_{j|(j,j')}^{\lambda,\sigma}$, where j' is the common baseline class k^* in the baseline learning and A_j for the OVA-SVM. We propose choosing the tuning parameters using a grid search, based on the prediction performance of $\hat{q}^{\lambda,\sigma}$'s on the tuning set $\mathbb{S}_{tune}$. Assume the true pairwise conditional probabilities are $q_{j|(j,j')}(\boldsymbol{x})$. To quantify the estimation accuracy of $\hat{q}$'s, we use the generalized Kullback–Leibler (GKL) loss function:

$$\begin{aligned}
&\mathrm{GKL}(q_{j|(j,j')}, \hat{q}^{\lambda,\sigma}_{j|j,j')}) \qquad (15)\\
= &\mathop{\mathbb{E}}_{X\sim\mathcal{P}}\left[q_{j|(j,j')}(\boldsymbol{X})\log\frac{q_{j|(j,j')}(\boldsymbol{X})}{\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{X})} + (1-q_{j|(j,j')}(\boldsymbol{X}))\log\frac{1-q_{j|(j,j')}(\boldsymbol{X})}{1-\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{X})}\right]\\
= &\mathop{\mathbb{E}}_{X\sim\mathcal{P}}\left[q_{j|(j,j')}(\boldsymbol{X})\log q_{j|(j,j')}(\boldsymbol{X}) + (1-q_{j|(j,j')}(\boldsymbol{X}))\log(1-q_{j|(j,j')}(\boldsymbol{X}))\right] \qquad (16)\\
&-\mathop{\mathbb{E}}_{X\sim\mathcal{P}}\left[q_{j|(j,j')}(\boldsymbol{X})\log\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{X}) + (1-q_{j|(j,j')}(\boldsymbol{X}))\log(1-\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{X}))\right]. \qquad (17)
\end{aligned}$$

The term (16) is unrelated to $\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{X})$, so only the term (17) is essential. Since the true values of $q_{j|(j,j')}(\boldsymbol{X})$ are unknown in practice, we need to approximate GKL empirically based on the tuning data. Set a function $R_{j,j'}(Y) = 1$ for $y_i = j$; $= -1$ for $y_i = j'$, which satisfies $\mathbb{E}\left[\frac{1}{2}(R_{j,j'}(Y)+1)|\boldsymbol{X}, Y \in \{j, j'\}\right] = q_{j|(j,j')}(\boldsymbol{X})$, so a computable proxy of GKL is given by

$$\begin{aligned}
&\mathrm{EGKL}(\hat{q}^{\lambda,\sigma}_{j|j,j')}) \qquad (18)\\
= &-\frac{1}{2n_{jj'}}\sum_{\forall i: y_i = j\vee j'}\left[(1+R_{j,j'}(y_i))\log\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{x}_i) + (1-R_{j,j'}(y_i))\log(1-\hat{q}^{\lambda,\sigma}_{j|(j,j')}(\boldsymbol{x}_i))\right].
\end{aligned}$$

The optimal values of $\{\lambda, \sigma\}$ are chosen by minimizing the EGKL.

7 Numerical Studies

7.1 Experiment Setup

We evaluate the performance of the baseline learning (denoted by B-SVM) and the OVA-SVM (denoted by A-SVM) schemes, and the pairwise coupling enhanced by the baseline learning (denoted by BP-SVM). We compare them with our enhanced implementation of the pairwise coupling algorithm in Wang et al. (2019) to dynamically chooses the best baseline class for each data point to be the class k that has the maximum voting for larger pairwise conditional probabilities (denoted by P-SVM). All of these methods are implemented in R, and the tuning parameters are chosen using EGKL based on a tuning set. In addition, we include five popular benchmark methods in the literature: multinomial logistic regression (MLG), multiclass linear discriminant analysis (MLDA), classification tree (TREE, Breiman et al., 1984), random forest (RF, Ho, 1995), and XGBoost (XGB, Chen and Guestrin, 2016), which are implemented in R with hyperparameters tuned by cross-validation. All of the numerical experiments are performed on the High Performance Computing (HPC) cluster at the University of Arizona, with Lenovo's Nextscale M5 Intel Haswell V3 28 core processors and 18GB memory.

In all the simulation examples, we set $|\mathbb{S}_{train}| = |\mathbb{S}_{tune}| = n$. The weights are specified as $\pi_j = (j-1)/M$, $j = \{1, \ldots, M+1\}$, where M is a preset value to control estimation precision. Following Wang et al. (2008), we set $M = \lfloor\sqrt{n}\rfloor$. We use the RBF kernel and select the optimal values of $\{\lambda, \sigma\}$ using GKL or EGKL based on a grid search. The range of σ is $\{\sigma_M/4, 2\sigma_M/4, 3\sigma_M/4, 4\sigma_M/4, 5\sigma_M/4, 6\sigma_M/4\}$, where $\sigma_M = \mathrm{median}_{y_s\neq y_t}\|\boldsymbol{x}_s - \boldsymbol{x}_t\|_2$ is the median pairwise Euclidean distance between K classes (Wu et al., 2010), and $\lambda \in \{5.5\times10^{-8}, 1\times10^{-8}, 5.5\times10^{-7}, \ldots, 5.5\times10^{+7}, 1\times10^{+8}\}$. We report the results of both GKL and EGKL, showing that they give a comparable performance. In real applications, since GKL is not available, we will use EGKL for tuning.

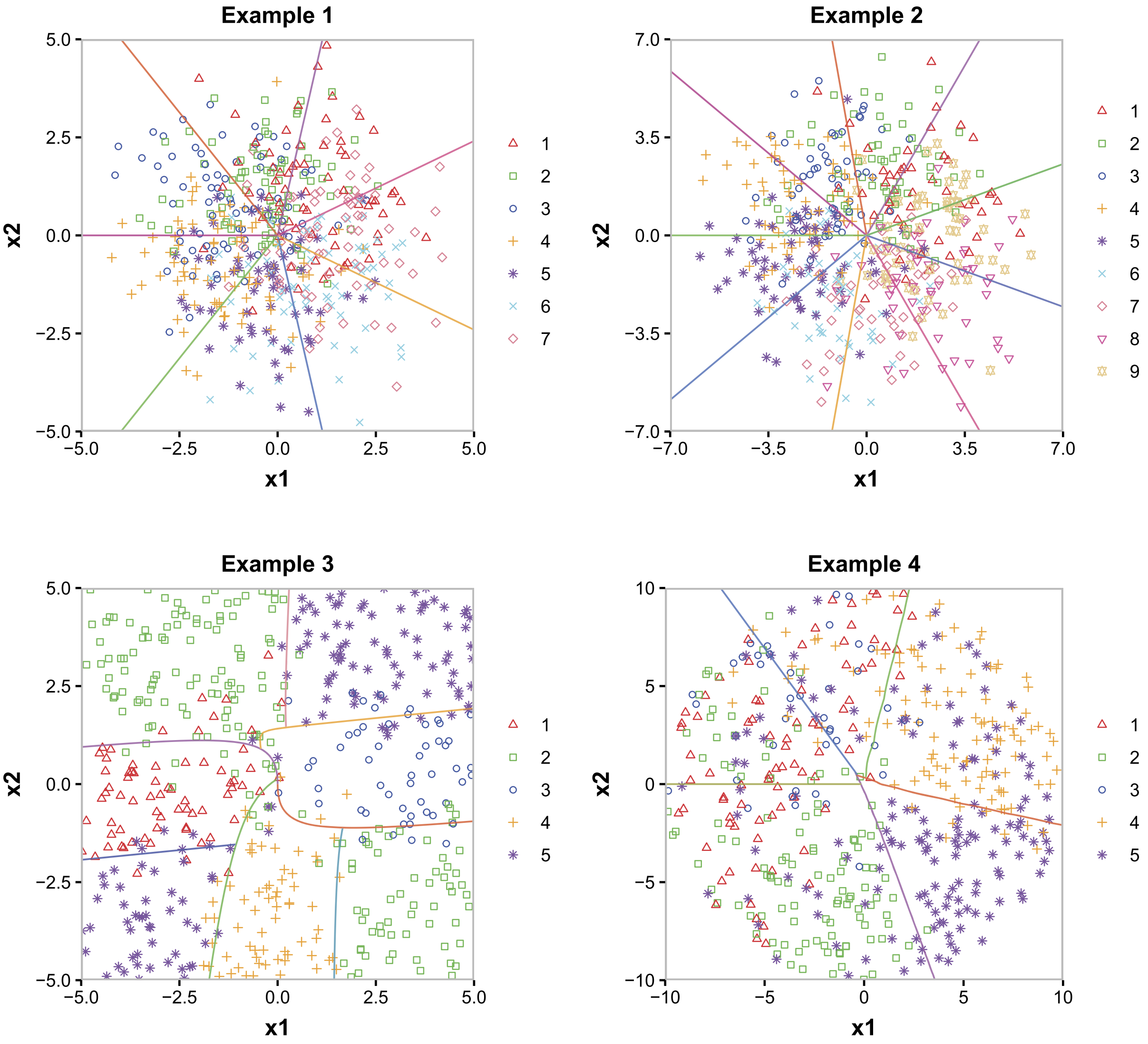

Figure 1: The scatter plots of the training data for Examples 1–4. Different symbols are for different classes, and the optimal decision boundaries (Bayes rules) are solid lines.

To evaluate the estimation accuracy of the probability estimator $\hat{p}_j$'s, we further generate a test set $\mathbb{S}_{test} = \{(\tilde{\boldsymbol{x}}_i, \tilde{y}_i), i = 1, \ldots, \tilde{n}\}$, of the size $|\mathbb{S}_{test}| = \tilde{n}$. The following metrics are used to quantify the performance of different procedures: (i) 1-norm error $\frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \sum_{j=1}^{K} |\hat{p}_j(\tilde{\boldsymbol{x}}_i) - p_j(\tilde{\boldsymbol{x}}_i)|$; (ii) 2-norm error $\frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \sum_{j=1}^{K} (\hat{p}_j(\tilde{\boldsymbol{x}}_i) - p_j(\tilde{\boldsymbol{x}}_i))^2$; (iii) the EGKL loss $\frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \sum_{j=1}^{K} p_j(\tilde{\boldsymbol{x}}_i) \log \frac{p_j(\tilde{\boldsymbol{x}}_i)}{\hat{p}_j(\tilde{\boldsymbol{x}}_i)}$; and (iv) the GKL loss $\frac{1}{\tilde{n}} \sum_{i=1}^{\tilde{n}} \sum_{j=1}^{K} \left[p_j(\tilde{\boldsymbol{x}}_i) \log \frac{p_j(\tilde{\boldsymbol{x}}_i)}{\hat{p}_j(\tilde{\boldsymbol{x}}_i)} + (1 - p_j(\tilde{\boldsymbol{x}}_i)) \log \frac{(1-p_j(\tilde{\boldsymbol{x}}_i))}{(1-\hat{p}_j(\tilde{\boldsymbol{x}}_i))} \right]$. To evaluate the classification accuracy of the learning methods, we report two misclassification rates based on the maximum probability and max voting algorithms.

We consider four examples, including two linear cases (Examples 1–2) and two nonlinear cases (Examples 3–4). The number of classes $K \in \{5, 7, 9\}$. MLG and MLDA are oracles methods for Examples 1–2, but not for Examples 3–4.

Figure 1 illustrates the scatter plots of the training data points for Examples 1–4, along with the optimal (Bayes) decision boundary shown in solid lines. For all the examples, $n = 500$ and $\tilde{n} = 10,000$. For each example, we run $N_r = 100$ Monte Carlo simulations and report the average performance measurement with standard error.

7.2 Linear Examples

Example 1 ($K = 7$, linear)**.** The data are generated as follows: (i) Generate Y uniformly from $\{1, 2, 3, 4, 5, 6, 7\}$; (ii) Given $Y = y$, generate $\boldsymbol{X}$ from the bivariate normal distribution $N(\boldsymbol{\mu}(y), \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}(y) = (1.5\cos(2y\pi/7), 1.5\sin(2y\pi/7))^{\mathrm{T}}$, $\boldsymbol{\Sigma} = 1.2^2\mathbf{I}_2$.

Example 2 ($K = 9$, linear)**.** The data are generated as follows: (i) Generate Y uniformly from $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$; (ii) Given $Y = y$, generate $\boldsymbol{X}$ from the bivariate normal distribution $N(\boldsymbol{\mu}(y), \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}(y) = (2.5\cos(2y\pi/9), 2.5\sin(2y\pi/9))^{\mathrm{T}}$, $\boldsymbol{\Sigma} = 1.5^2\mathbf{I}_2$.

Examples 1–2 are two linear examples used to show that the baseline methods can handle a large class size $K \geqslant 5$ with competitive performance and the best running time. Table 1 summarizes the performance of the baseline learning methods (B1-SVM, B2-SVM), the pairwise coupling enhanced by the baseline learning (BP1-SVM, BP2-SVM), respectively, based on two methods of choosing the baseline class k^*, the OVA-SVM method (A-SVM), the pairwise coupling with dynamic baseline class choosing for each data point (P-SVM), and five benchmark methods: MLG (Oracle-1), MLDA (Oracle-2), TREE, RF and XGBoost. We consider 2 different class sizes and report seven performance measures: running time, 1-norm error, 2-norm error, EGKL loss, GKL loss, and the misclassification rate for classification. The standard error (SE), as calculated by $\sigma_{\bar{v}} = \sigma_{\boldsymbol{v}}/\sqrt{N_r}$, is in the parenthesis.

We make the following observations. In general, SVM-based methods outperform other methods and are closest to the oracle methods in all cases. Two B-SVM estimators have similar probability estimation performance as the pairwise coupling methods, and they outperform all three benchmark methods besides the oracle procedures (MLG and MLDA), showing a significant gain in computation efficiency when K is large. The BP-SVM methods have no additional running cost since they don't involve additional training, showing better classification accuracy in some cases. Based on data distribution, two B-SVM methods may select different baseline class k^*, which may give slightly different yet similar results. The OVA-SVM consistently gives the best probability estimation and classification in all methods, however with the most expensive computational cost, which agrees with the theoretic complexity analysis. The tuning criteria GKL and EGKL give similar performance, showing that EGKL is a good approximation as GKL for parameter tuning. Noted, the max voting algorithm works only with the full pairwise probability table, so it is not applicable for OVA-SVM and B-SVM, as denoted as NA. The tree based methods, such as classification trees (TREE) and random forest (RF), may incur 0 in class probability estimation for some $\boldsymbol{x}$ and classes, resulting "Inf" in Tables 1 and 2 for both EGKL and GKL, thus the SEs are unavailable and output NaN ("Not A Number") in R.

7.3 Nonlinear Examples

Here we show that the proposed methods can handle complex non-linear problems and offer competitive performance in both probability estimation and classification accuracy. In these examples, the linear methods MLG and MLDA are not the oracle anymore.

Table 1: Performance measure on the test set for Example 1 and Example 2.

Example 1 (Seven-class Linear Example, MLG and MLDA as Oracle, denoted as Oracle-1 and Oracle-2)											
EGKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	Oracle-1	Oracle-2	TREE	RF	XGB
Time	0.9 (0.0)	0.9 (0.0)	0.8 (0.0)	0.8 (0.0)	30.7 (0.1)	2.8 (0.0)	0.0 (0.0)	0.0 (0.0)	0.4 (0.0)	1.8 (0.0)	1.8 (1.1)
L_1	47.9 (0.2)	47.9 (0.2)	48.0 (0.3)	48.0 (0.3)	41.8 (0.2)	46.5 (0.1)	36.5 (0.2)	36.0 (0.1)	56.3 (0.2)	72.0 (0.1)	95.2 (0.2)
L_2	9.5 (0.1)	9.5 (0.1)	9.8 (0.1)	9.8 (0.1)	8.5 (0.1)	9.1 (0.0)	8.2 (0.1)	8.1 (0.0)	11.7 (0.1)	19.4 (0.1)	35.6 (0.1)
EGKL	31.7 (1.9)	31.7 (1.9)	33.3 (1.9)	33.3 (1.9)	26.7 (1.3)	30.2 (1.2)	22.5 (1.2)	22.1 (1.2)	Inf (NaN)	Inf (NaN)	124.6 (2.6)
GKL	25.1 (0.7)	25.1 (0.7)	26.9 (1.2)	26.9 (1.2)	15.1 (0.4)	24.6 (0.4)	14.8 (0.4)	14.4 (0.3)	Inf (NaN)	Inf (NaN)	162.0 (1.4)
TE1	60.7 (0.2)	60.5 (0.2)	60.9 (0.2)	60.7 (0.2)	60.0 (0.1)	60.3 (0.1)	57.6 (0.1)	57.7 (0.1)	61.9 (0.1)	64.7 (0.1)	65.7 (0.1)
TE2	NA (NA)	60.3 (0.2)	NA (NA)	60.5 (0.2)	NA (NA)	60.3 (0.1)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	1	1	5	5	NA	NA	NA	NA	NA	NA	NA
GKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	Oracle-1	Oracle-2	TREE	RF	XGB
Time	0.9 (0.0)	0.9 (0.0)	0.8 (0.0)	0.8 (0.0)	30.8 (0.1)	2.8 (0.0)	0.0 (0.0)	0.0 (0.0)	0.4 (0.0)	1.8 (0.0)	1.8 (1.1)
L_1	48.4 (0.3)	48.4 (0.3)	48.5 (0.3)	48.5 (0.3)	42.2 (0.2)	46.9 (0.2)	36.5 (0.2)	36.0 (0.1)	56.3 (0.2)	72.0 (0.1)	95.2 (0.2)
L_2	10.4 (0.1)	10.4 (0.1)	10.3 (0.1)	10.3 (0.1)	9.2 (0.0)	9.8 (0.1)	8.2 (0.1)	8.1 (0.0)	11.7 (0.1)	19.4 (0.1)	35.6 (0.1)
EGKL	31.8 (2.4)	31.8 (2.4)	33.9 (2.6)	33.9 (2.6)	26.8 (1.3)	30.3 (1.2)	22.5 (1.2)	22.1 (1.2)	Inf (NaN)	Inf (NaN)	124.6 (2.6)
GKL	25.3 (1.1)	25.3 (1.1)	26.8 (0.9)	26.8 (0.9)	15.2 (0.4)	24.6 (0.4)	14.8 (0.4)	14.4 (0.3)	Inf (NaN)	Inf (NaN)	162.0 (1.4)
TE1	60.8 (0.2)	60.6 (0.2)	61.2 (0.2)	60.9 (0.2)	59.9 (0.1)	60.3 (0.2)	57.6 (0.1)	57.7 (0.1)	61.9 (0.1)	64.7 (0.1)	65.7 (0.1)
TE2	NA (NA)	60.4 (0.2)	NA (NA)	60.8 (0.3)	NA (NA)	60.3 (0.2)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	1	1	5	5	NA	NA	NA	NA	NA	NA	NA
Example 2 (Nine-class Linear Example, MLG and MLDA as Oracle, denoted as Oracle-1 and Oracle-2)											
EGKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	Oracle-1	Oracle-2	TREE	RF	XGB
Time	0.8 (0.0)	0.8 (0.0)	0.8 (0.0)	0.8 (0.0)	31.6 (0.2)	3.4 (0.0)	0.0 (0.0)	0.0 (0.0)	0.4 (0.0)	1.8 (0.0)	7.2 (2.8)
L_1	46.1 (0.3)	46.1 (0.3)	47.0 (0.3)	47.0 (0.3)	43.5 (0.1)	45.2 (0.1)	43.5 (0.2)	42.7 (0.2)	64.0 (0.2)	75.2 (0.1)	99.8 (0.1)
L_2	12.4 (0.1)	12.4 (0.1)	12.3 (0.1)	12.3 (0.1)	11.1 (0.1)	12.0 (0.0)	10.6 (0.1)	10.4 (0.1)	14.5 (0.1)	21.7 (0.1)	40.0 (0.1)
EGKL	32.0 (2.4)	32.0 (2.4)	32.2 (2.2)	32.2 (2.2)	27.0 (1.6)	31.6 (1.2)	26.9 (1.4)	25.9 (1.3)	Inf (NaN)	Inf (NaN)	145.2 (2.8)
GKL	22.6 (1.1)	22.6 (1.1)	22.5 (1.2)	22.5 (1.2)	18.5 (0.6)	21.7 (0.5)	18.1 (0.4)	16.9 (0.4)	Inf (NaN)	Inf (NaN)	187.1 (1.8)
TE1	61.2 (0.2)	60.4 (0.2)	61.6 (0.3)	60.9 (0.3)	60.1 (0.1)	60.9 (0.1)	57.4 (0.1)	57.3 (0.1)	62.5 (0.1)	64.2 (0.1)	65.3 (0.1)
TE2	NA (NA)	60.3 (0.2)	NA (NA)	60.6 (0.3)	NA (NA)	60.9 (0.1)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	6	6	8	8	NA	NA	NA	NA	NA	NA	NA
GKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	Oracle-1	Oracle-2	TREE	RF	XGB
Time	0.8 (0.0)	0.8 (0.0)	0.8 (0.0)	0.8 (0.0)	31.7 (0.2)	3.4 (0.0)	0.0 (0.0)	0.0 (0.0)	0.4 (0.0)	1.8 (0.0)	7.2 (2.8)
L_1	46.1 (0.5)	46.1 (0.5)	47.1 (0.4)	47.1 (0.4)	43.1 (0.1)	45.2 (0.3)	43.5 (0.2)	42.7 (0.2)	64.0 (0.2)	75.2 (0.1)	99.8 (0.1)
L_2	12.6 (0.2)	12.6 (0.2)	12.8 (0.1)	12.8 (0.1)	11.6 (0.1)	12.5 (0.1)	10.6 (0.1)	10.4 (0.1)	14.5 (0.1)	21.7 (0.1)	40.0 (0.1)
EGKL	32.0 (3.5)	32.0 (3.5)	32.9 (3.1)	32.9 (3.1)	27.1 (1.6)	31.6 (1.4)	26.9 (1.4)	25.9 (1.3)	Inf (NaN)	Inf (NaN)	145.2 (2.8)
GKL	22.5 (3.2)	22.5 (3.2)	22.3 (1.9)	22.3 (1.9)	18.1 (0.6)	21.5 (1.0)	18.1 (0.4)	16.9 (0.4)	Inf (NaN)	Inf (NaN)	187.1 (1.8)
TE1	62.0 (0.3)	61.9 (0.3)	62.0 (0.3)	61.8 (0.3)	60.7 (0.1)	61.3 (0.3)	57.4 (0.1)	57.3 (0.1)	62.5 (0.1)	64.2 (0.1)	65.3 (0.1)
TE2	NA (NA)	61.3 (0.3)	NA (NA)	61.3 (0.3)	NA (NA)	61.3 (0.2)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	6	6	8	8	NA	NA	NA	NA	NA	NA	NA

TABLE NOTE: The running time is measured in minutes. L_1 (the 1-norm error), L_2 (the 2-norm error), EGKL (the EGKL loss), GKL (the GKL loss), TE1 (the misclassification rate based on max probability), TE2 (the misclassification rate based on max voting), multiplied by 100 for both mean and standard derivation in parenthesis. Baseline class k^* in baseline methods are calculated by statistical mode. MLG: Multinomial logistic regression; MLDA: Multiclass linear discriminant analysis; TREE: Classification tree; RF: Random forest; XGB: XGBoost. The same explanation applies to the results of all the other examples.

Table 2: Performance measure on the test set for Example 3 and Example 4.

Example 3 (Five-class Non-linear Example, MLG and MLDA are not Oracle)

EGKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	MLG	MLDA	TREE	RF	XGB
Time	1.5 (0.0)	1.5 (0.0)	0.7 (0.0)	0.7 (0.0)	11.7 (0.0)	2.4 (0.0)	0.0 (0.0)	0.0 (0.0)	0.3 (0.0)	1.1 (0.0)	0.4 (0.0)
L_1	19.2 (0.3)	19.2 (0.3)	21.1 (1.2)	21.1 (1.2)	15.2 (0.1)	18.9 (0.1)	115.9 (0.1)	116.3 (0.1)	27.9 (0.2)	22.4 (0.1)	27.9 (0.1)
L_2	4.1 (0.2)	4.1 (0.2)	4.3 (0.7)	4.3 (0.7)	3.3 (0.1)	4.0 (0.1)	50.7 (0.1)	50.4 (0.1)	7.0 (0.1)	4.4 (0.1)	9.4 (0.1)
EGKL	12.5 (0.9)	12.5 (0.9)	13.7 (1.9)	13.7 (1.9)	9.2 (0.6)	12.2 (0.5)	60.7 (0.8)	60.8 (0.8)	Inf (NaN)	Inf (NaN)	28.6 (1.2)
GKL	22.6 (0.6)	22.6 (0.6)	26.9 (2.3)	26.9 (2.3)	15.6 (0.3)	20.8 (0.3)	88.5 (0.2)	89.7 (0.2)	Inf (NaN)	Inf (NaN)	49.7 (0.7)
TE1	16.6 (0.3)	16.6 (0.3)	16.7 (0.8)	16.7 (0.8)	16.4 (0.1)	16.5 (0.1)	62.1 (0.4)	61.8 (0.4)	17.5 (0.1)	17.2 (0.1)	18.2 (0.1)
TE2	NA (NA)	16.5 (0.3)	NA (NA)	16.6 (0.8)	NA (NA)	16.6 (0.1)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	2	2	4	4	NA	NA	NA	NA	NA	NA	NA
GKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	MLG	MLDA	Tree	RF	XGB
Time	1.5 (0.0)	1.5 (0.0)	0.7 (0.0)	0.7 (0.0)	11.7 (0.0)	2.4 (0.0)	0.0 (0.0)	0.0 (0.0)	0.3 (0.0)	1.1 (0.0)	0.4 (0.0)
L_1	19.0 (0.6)	19.0 (0.6)	21.4 (1.3)	21.4 (1.3)	15.9 (0.1)	18.7 (0.2)	115.9 (0.1)	116.3 (0.1)	27.9 (0.2)	22.4 (0.1)	27.9 (0.1)
L_2	4.2 (0.4)	4.2 (0.4)	4.4 (0.8)	4.4 (0.8)	3.2 (0.0)	4.1 (0.2)	50.7 (0.1)	50.4 (0.1)	7.0 (0.1)	4.4 (0.1)	9.4 (0.1)
EGKL	12.9 (1.2)	12.9 (1.2)	13.7 (1.6)	13.7 (1.6)	8.9 (0.5)	11.3 (0.5)	60.7 (0.8)	60.8 (0.8)	Inf (NaN)	Inf (NaN)	28.6 (1.2)
GKL	22.6 (0.9)	22.6 (0.9)	26.3 (1.5)	26.3 (1.5)	15.3 (0.3)	20.9 (0.3)	88.5 (0.2)	89.7 (0.2)	Inf (NaN)	Inf (NaN)	49.7 (0.7)
TE1	16.7 (0.4)	16.7 (0.4)	16.7 (1.0)	16.7 (1.0)	16.4 (0.1)	16.6 (0.1)	62.1 (0.4)	61.8 (0.4)	17.5 (0.1)	17.2 (0.1)	18.2 (0.1)
TE2	NA (NA)	16.6 (0.4)	NA (NA)	16.6 (1.0)	NA (NA)	16.5 (0.1)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	2	2	4	4	NA	NA	NA	NA	NA	NA	NA

Example 4 (Five-class Non-linear Example, MLG and MLDA are not Oracle)

EGKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	MLG	MLDA	TREE	RF	XGB
Time	1.0 (0.0)	1.0 (0.0)	0.8 (0.0)	0.8 (0.0)	7.2 (0.1)	2.1 (0.0)	0.0 (0.0)	0.0 (0.0)	0.3 (0.0)	1.5 (0.0)	0.5 (0.0)
L_1	23.3 (0.3)	23.3 (0.3)	24.4 (0.7)	24.4 (0.7)	20.1 (0.2)	22.6 (0.2)	41.2 (0.1)	41.2 (0.1)	31.9 (0.2)	44.0 (0.1)	64.1 (0.2)
L_2	5.3 (0.1)	5.3 (0.1)	5.6 (0.3)	5.6 (0.3)	4.6 (0.1)	5.2 (0.1)	7.1 (0.0)	7.2 (0.0)	6.1 (0.1)	9.9 (0.1)	21.6 (0.1)
EGKL	12.0 (1.8)	12.0 (1.8)	12.5 (1.7)	12.5 (1.7)	10.1 (1.2)	11.9 (1.0)	18.0 (1.0)	18.6 (1.0)	Inf (NaN)	Inf (NaN)	65.1 (1.9)
GKL	16.6 (0.7)	16.6 (0.7)	17.2 (1.0)	17.2 (1.0)	14.9 (0.5)	16.5 (0.4)	21.2 (0.2)	22.7 (0.3)	Inf (NaN)	Inf (NaN)	105.7 (1.2)
TE1	41.5 (0.1)	41.4(0.1)	42.5 (0.4)	42.5 (0.4)	40.8 (0.1)	41.5 (0.1)	45.0 (0.1)	44.6 (0.1)	43.7 (0.2)	46.3 (0.1)	47.9 (0.1)
TE2	NA (NA)	41.3(0.1)	NA (NA)	42.3 (0.4)	NA (NA)	41.1 (0.1)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	5	5	1	1	NA	NA	NA	NA	NA	NA	NA
GKL	B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	MLG	MLDA	TREE	RF	XGB
Time	1.0 (0.0)	1.0 (0.0)	0.8 (0.0)	0.8 (0.0)	7.2 (0.1)	2.1 (0.0)	0.0 (0.0)	0.0 (0.0)	0.3 (0.0)	1.5 (0.0)	0.5 (0.0)
L_1	22.9 (0.3)	22.9 (0.3)	24.7 (0.7)	24.7 (0.7)	19.5 (0.2)	22.0 (0.2)	41.2 (0.1)	41.2 (0.1)	31.9 (0.2)	44.0 (0.1)	64.1 (0.2)
L_2	5.2 (0.1)	5.2 (0.1)	5.5 (0.2)	5.5 (0.2)	4.5 (0.1)	5.0 (0.1)	7.1 (0.0)	7.2 (0.0)	6.1 (0.1)	9.9 (0.1)	21.6 (0.1)
EGKL	11.8 (1.8)	11.8 (1.8)	12.3 (1.7)	12.3 (1.7)	9.8 (1.2)	11.1 (1.0)	18.0 (1.0)	18.6 (1.0)	Inf (NaN)	Inf (NaN)	65.1 (1.9)
GKL	16.9 (0.6)	16.9 (0.6)	17.5 (0.9)	17.5 (0.9)	14.2 (0.4)	16.5 (0.3)	21.2 (0.2)	22.7 (0.3)	Inf (NaN)	Inf (NaN)	105.7 (1.2)
TE1	41.5 (0.1)	41.4 (0.1)	42.6 (0.4)	42.6 (0.4)	40.9 (0.1)	41.4 (0.1)	45.0 (0.1)	44.6 (0.1)	43.7 (0.2)	46.3 (0.1)	47.9 (0.1)
TE2	NA (NA)	41.3 (0.1)	NA (NA)	42.5 (0.4)	NA (NA)	41.1 (0.1)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
k^*	5	5	1	1	NA	NA	NA	NA	NA	NA	NA

Example 3 ($K = 5$, nonlinear)**.** For any $\boldsymbol{x} = (x_1, x_2)^{\mathrm{T}}$, define

$$\begin{aligned}
f_1(\boldsymbol{x}) &= -1.5x_1 + 0.2x_1^2 - 0.1x_2^2 + 0.2 \\
f_2(\boldsymbol{x}) &= 0.3x_1^2 + 0.2x_2^2 - x_1x_2 + 0.2 \\
f_3(\boldsymbol{x}) &= 1.5x_1 + 0.2x_1^2 - 0.1x_2^2 + 0.2 \\
f_4(\boldsymbol{x}) &= -0.1x_1^2 + 0.2x_2^2 - 1.5x_2 + x_1 + 0.1x_1x_2 \\
f_5(\boldsymbol{x}) &= 0.1x_1^2 + 0.1x_2^2 + x_1x_2 - 0.2
\end{aligned}$$

and $p_j(\boldsymbol{x}) = e^{f_j(\boldsymbol{x})}/(\sum_{l=1}^{5} e^{f_l(\boldsymbol{x})})$ for $j = 1, 2, 3, 4, 5$. The data are generated as follows: (i) Generate independent X_1 and X_2 uniformly from $[-5, 5]$ and $[-5, 5]$, respectively; (ii) Given $\boldsymbol{X} = \boldsymbol{x}$, the label Y takes the value j with the probability $p_j(\boldsymbol{x})$ for $j = 1, 2, 3, 4, 5$.

Example 4 ($K = 5$, nonlinear)**.** Generate $\boldsymbol{X} = (X_1, X_2)^{\mathrm{T}}$ uniformly from a disc $\{\boldsymbol{x} : x_1^2 + x_2^2 \leqslant 100\}$. Define $h_1(\boldsymbol{x}) = -3x_1\sqrt{5} + 3x_2$, $h_2(\boldsymbol{x}) = -3x_1\sqrt{5} - 3x_2$, $h_3(\boldsymbol{x}) = x_2\sqrt{3} - 1.2x_1$, $h_4(\boldsymbol{x}) = 2x_2\sqrt{3} + 1.2x_1$, and $h_5(\boldsymbol{x}) = \sqrt{|x_1x_2| + 1}$. Then consider the non-linear transformation $f_j(\boldsymbol{x}) = \Phi^{-1}(T_2(h_j(\boldsymbol{x})))$ for $j = 1, 2, 3, 4, 5$, where $\Phi(\cdot)$ is the cumulative distribution function(cdf) of the standard normal distribution and $T_2(\cdot)$ is the cdf of t_2 distribution. Then we set the probability functions as $p_j(\boldsymbol{x}) = \exp f_j(\boldsymbol{x})/(\sum_{l=1}^{5} \exp(f_l(\boldsymbol{x})))$ for $j = 1, 2, 3, 4, 5$.

Table 2 summarizes the performance of the proposed methods and their comparison with the pairwise coupling and five benchmark methods. Similar to the linear cases, the proposed OVA-SVM has the best performance among all the methods. The B-SVMs have close performance as pairwise coupling and outperform all the benchmark methods.

8 Benchmark Data Analysis

We use five real data sets, including three low-dimensional cases and two high-dimensional cases, to compare the performance of all the methods.

8.1 Low-dimensional Cases ($p < n$)

We first use three low-dimensional benchmark data sets, including pen-based handwritten digits, *E.coli* and yeast protein localization data, to evaluate the wSVMs. The data information and performance measurements are shown in Table 3.

The pen-based handwritten digits data set, abbreviated as *Pendigits*, is obtained from Dua and Graff (2019). It consists of 7,494 training data points and 3,498 test data points, based on collecting 250 random digit samples from 44 writers (Alimoglu and Alpaydin, 1997). Since the *Pendigits* is a large data set with a total of $K = 10$ classes, we start with a small classification problem by discriminating $K = 4$ classes among digits 1, 3, 6, and 9; then we use the full data set to evaluate the methods for a $K = 10$ class problem. The *E.coli* data set is obtained from Dua and Graff (2019), which includes $n = 336$ samples and $p = 7$ predictor variables. The goal is to predict the *E.coli* cellular protein localization sites (Horton and Nakai, 1996). Based on the biological functional similarity and locations proximity, we create a $K = 4$ class problem, including class “inner membrane” with 116 observations, class “outer membrane” with 25 observations, class “cytoplasm” with 143 observations, and class “periplasm” with 52 observations. The yeast data set is obtained from Dua and Graff (2019), and the goal is to predict yeast cellular protein localization sites (Horton and Nakai, 1996) using 8 predictor variables. The total sample size is

Table 3: Performance measures for benchmark data sets.

Data Sets	Data Info			Evals	Methods										
	K	***n***	***ñ***		B1-SVM	BP1-SVM	B2-SVM	BP2-SVM	A-SVM	P-SVM	MLG	MLDA	TREE	RF	XGB
E.coli	4	226	110	Time	0.1 (0.0)	0.1 (0.0)	0.1 (0.0)	0.1 (0.0)	0.3 (0.0)	0.2 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.1 (0.0)	1.3 (0.3)
				TE1	7.3 (0.5)	7.1 (0.4)	6.0 (0.4)	5.8 (0.3)	4.6 (0.4)	5.6 (0.3)	9.0 (1.3)	6.7 (0.7)	11.5 (0.7)	6.4 (0.6)	9.2 (1.2)
				TE2	NA (NA)	6.9 (0.3)	NA (NA)	5.6 (0.3)	NA (NA)	5.5 (0.3)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
				k^*	4	4	2	2	NA	NA	NA	NA	NA	NA	NA
Yeast	5	990	494	Time	1.1 (0.0)	1.1 (0.0)	0.9 (0.1)	0.9 (0.1)	13.4 (0.0)	1.9 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.4 (0.0)	2.4 (0.2)
				TE1	39.3 (0.5)	38.1 (0.5)	41.7 (1.3)	41.5 (1.4)	39.0 (0.6)	39.4 (0.6)	40.0 (0.3)	40.5 (0.4)	43.0 (0.6)	37.9 (0.5)	42.4 (0.5)
				TE2	NA (NA)	37.9 (0.6)	NA (NA)	41.8 (1.4)	NA (NA)	39.7 (0.4)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
				k^*	1	1	2	2	NA	NA	NA	NA	NA	NA	NA
Pendigits (1369)	4	2937	1372	Time	12.7 (0.1)	12.7 (0.1)	12.9 (0.3)	12.9 (0.3)	136.7 (0.9)	24.4 (0.3)	0.0 (0.0)	0.0 (0.0)	0.1 (0.0)	1.1 (0.0)	140.3 (4.7)
				TE1	1.0 (0.0)	0.9 (0.0)	1.0 (0.0)	1.0 (0.0)	0.8 (0.0)	0.9 (0.0)	3.3 (0.3)	6.8 (0.1)	7.1 (0.6)	1.3 (0.0)	2.0 (0.0)
				TE2	NA (NA)	0.9 (0.0)	NA (NA)	1.0 (0.0)	NA (NA)	0.9 (0.0)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
				k^*	1	1	1	1	NA	NA	NA	NA	NA	NA	NA
Pendigits (Full)	10	7494	3498	Time	43.4 (0.9)	43.5 (0.9)	53.1 (1.6)	53.1 (1.6)	1536.7 (9.4)	192.4 (1.9)	0.0 (0.0)	0.0 (0.0)	0.2 (0.0)	4.4 (0.1)	382.4 (5.4)
				TE1	4.1 (0.0)	3.9 (0.0)	4.5 (0.1)	4.4 (0.1)	3.7 (0.1)	4.0 (0.0)	10.6 (0.5)	17.1 (0.1)	25.9 (0.4)	5.3 (0.1)	5.2 (0.1)
				TE2	NA (NA)	3.9 (0.0)	NA (NA)	4.4 (0.1)	NA (NA)	4.0 (0.0)	NA (NA)	NA (NA)	NA (NA)	NA (NA)	NA (NA)
				k^*	2	2	1	1	NA	NA	NA	NA	NA	NA	NA

TABLE NOTE: The first column gives the names of the four benchmark data sets; the next 3 columns include the data set information: K is the number of classes, n is the training set size, and $\tilde{n}$ is the test set size. The remaining columns compare the performance of the proposed methods with benchmarks.

$n = 1{,}484$ with 10 different localization sites. Based on the biological functional similarity and locations proximity, we formulate a $K = 5$ class problem, including class "membrane proteins" with 258 observations, "CYT" with 463 observations, "NUC" with 429 observations, "MIT" with 244 observations, and class "other" with 90 observations.

For the pen-based handwritten digits data set, we randomly split the training and tuning sets of equal sizes 10 times and report the average misclassification rate evaluated on the test set along with the standard error. For *E.coli* and yeast data sets, since the class sizes are highly unbalanced, we first randomly split the data into the training, tuning, and test sets of equal size for each class separately, and then combine them across classes to form the final training, tuning, and test sets, for 10 times. We report the average test error rates along with the standard errors.

Table 3 suggests that the OVA-SVM overall is the best classifier among all cases, but its computational cost drastically increases with n and K. The two linear time methods consistently give similar performance as pairwise coupling in terms of error rates, but with a significantly reduced computation time for large n and K, and they outperform the benchmark methods. In some examples, the linear time method along with pairwise reconstruction may outperform pairwise coupling and One-vs-All methods (e.g. BP1-SVM on yeast and *Pendigits* data sets). In practice, when n and K are small, One-vs-All should be used for its best performance (e.g. *E.coli* data set); when K and n are large, linear methods have the best time complexity with similar performance as the pairwise and One-vs-All methods and are hence suitable for scalable applications.

The results suggest that the proposed One-vs-All method (OVA-SVM) consistently outperforms all of the methods in all the data sets, but has a high computational cost; the proposed baseline learning methods (B-SVM and BP-SVM) have a close performance to the pairwise coupling method and consistently outperform the benchmark methods, with a significant reduction in computational time. Similar observations are made for these four studies.

8.2 High-Dimensional Cases ($p > n$)

Two high-dimensional data sets are downloaded from the site: https://schlieplab.org/Static/Supplements/CompCancer/datasets.htm. The first data set consists of microarray gene expressions for the three major adult lymphoid malignancies of B-cell lymphoma: Diffuse large B-cell lymphoma (DLBCL), follicular lymphoma (FL), and chronic lymphocytic leukemia (CLL), as discussed in Alizadeh et al. (2000). There are a total of 2,093 expression profiles from 62 subjects. The second data set is the oligonucleotide microarrays gene expression profile for the treatment of pediatric acute lymphoblastic leukemia (ALL) based on the patient's risk of relapse, with six prognostically leukemia subtypes, including T-ALL, E2A-PBX1, BCR-ABL, TEL-AML1, MLL rearrangement, and hyperdiploid > 50 chromosomes. We combine the TEL-AML1, BCR-ABL and hyperdiploid > 50 chromosomes as class T+B+H for their similar gene functional clustering as discussed in Yeoh et al. (2002) and make it a $K = 4$ class problem, with $n = 248$ and $p = 2{,}526$. For each class, we randomly sample 75% of the data points and then split them equally into the training and tuning sets, and the remaining 25% of the data points are used as the test set. These three sets are combined across all the classes to form the final training, tuning, and test sets. For the B-cell lymphoma data set, we have the training and tuning sets of size 48 (containing 32 DLBCL, 7 FL, 9 CLL) and the test set of size 14 (containing 10 DLBCL, 2 FL, 2 CLL). For the acute lymphoblastic leukemia data set, we have the training and tuning set of size 188 (containing 33 T-ALL, 119 T+B+H, 15 MLL, 21 E2A-PBX1) and the test set of size 60 (containing 10 T-ALL, 39 T+B+H, 5 MLL, 6 E2A-PBX1).

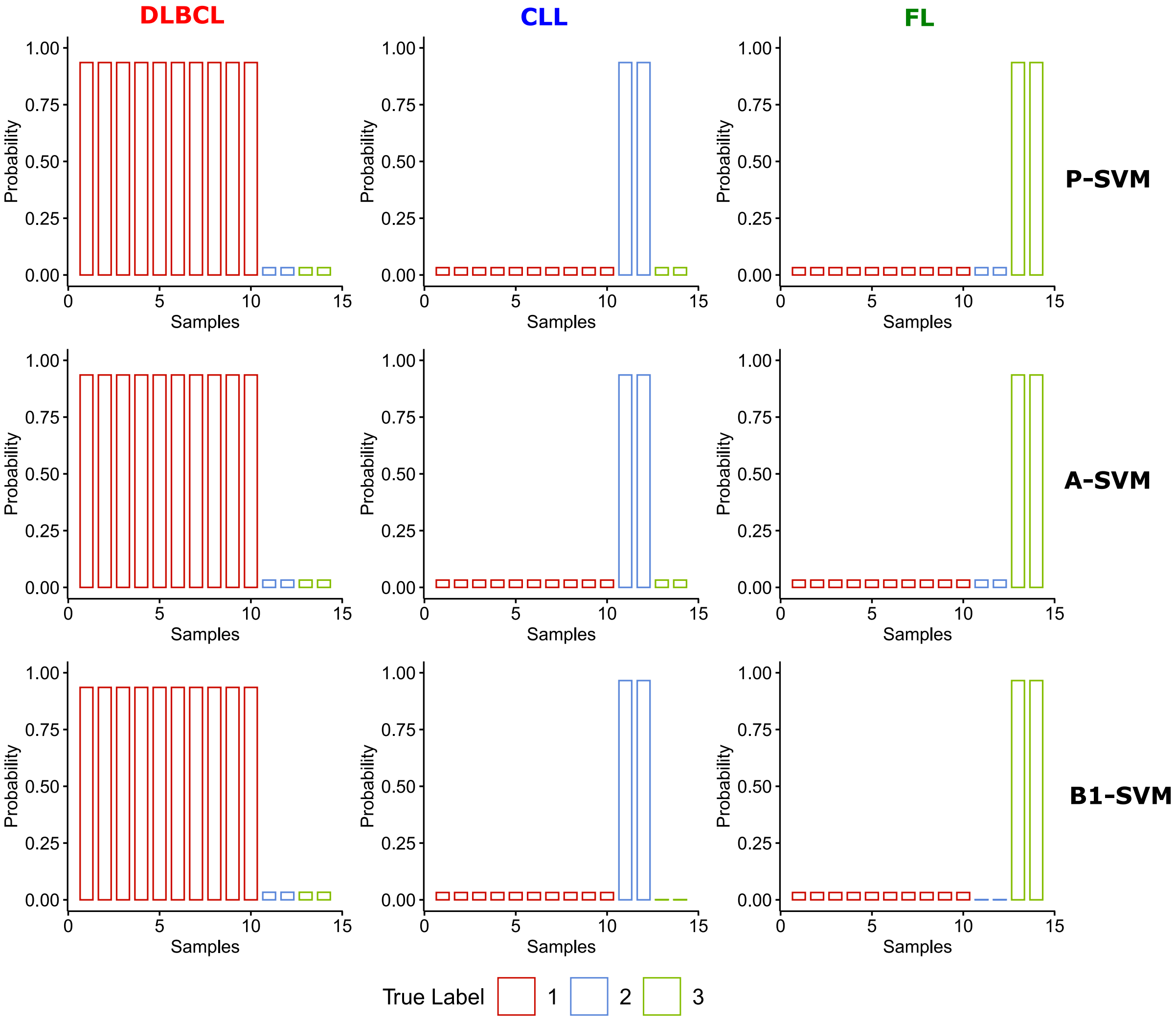

Figure 2: This figure plots the estimated probabilities (shown as the bar heights) for the B-cell lymphoma data set's test set for 14 data points with 3 methods. From the top to the bottom row are the evaluation of the pairwise coupling (P-SVM), OVA-SVM (A-SVM), and baseline (B1-SVM) methods. From the left to the right columns are estimated class probabilities of one subclass of test samples: DLBCL, CLL, and FL, labeled as 1, 2, 3, respectively, by the three methods. Three colors (red, azure, green) represent the estimated probabilities of test data points belonging to class DLBCL, CLL, and FL.

For both data sets, we normalize the gene expression values, obtain their Z-scores, and rank the genes by their BW criterion of the marginal relevance to the class label, calculated as the ratio of each gene's between-group to within-group sums of squares, as specified in Dudoit et al. (2002). We select top 5 genes as predictor variables based on the order of the relevance measures of the BW score. Figure 2 plots the estimated class probability for B-cell lymphoma data set on 14 test data points with the baseline method 1 (B1-SVM), OVA-SVM (A-SVM), and pairwise coupling (P-SVM), whose running time of 1 sec, 3 secs, and 2 secs, respectively.

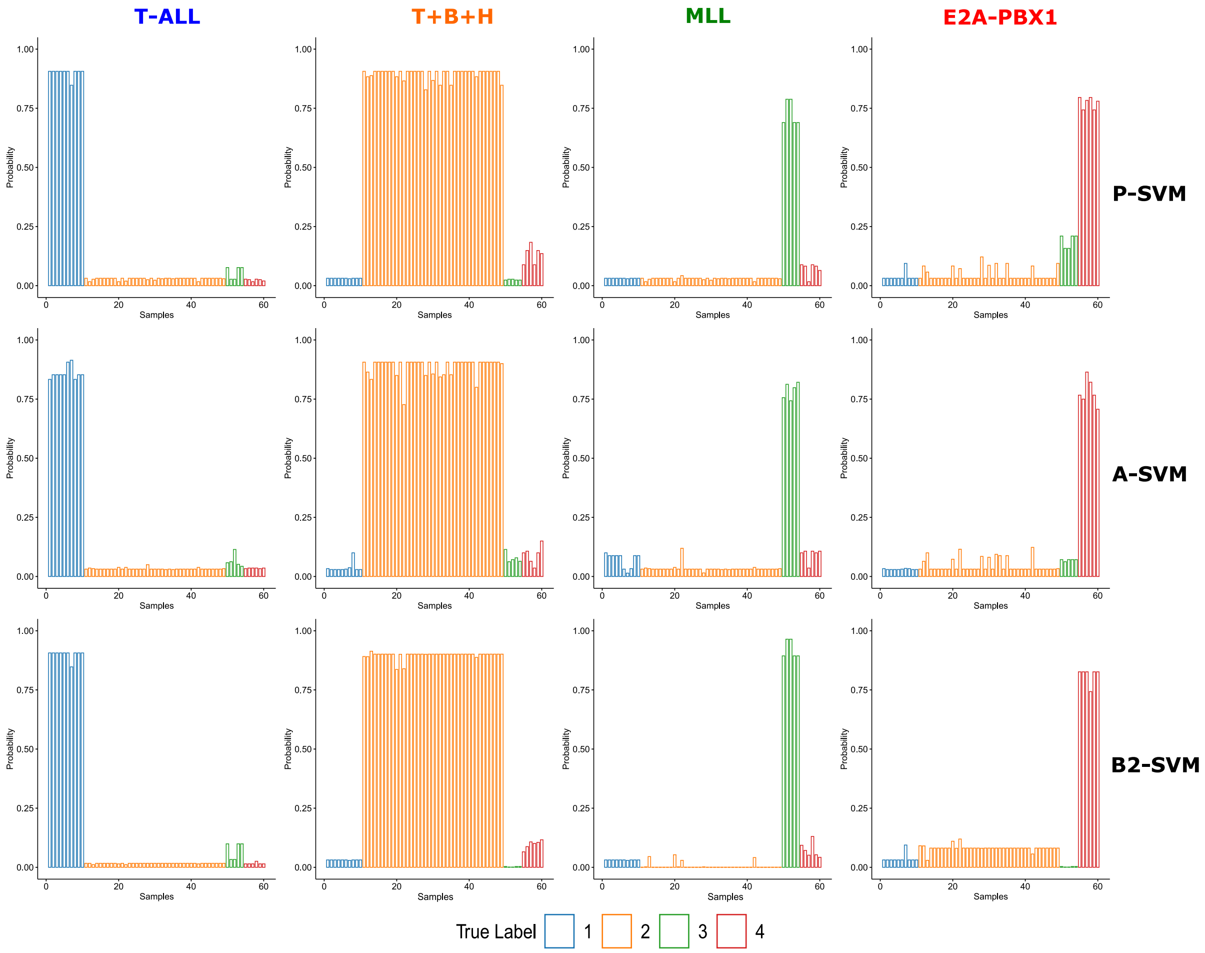

Figure 3: This figure plots the estimated probabilities (shown as the bar heights) for the acute lymphoblastic leukemia test set for 60 data points with 3 methods. From the top to the bottom row are the pairwise coupling (P-SVM), OVA-SVM (A-SVM), and the baseline (B2-SVM) methods. From the left to the right are estimated class probabilities of one subclass of test samples: T-ALL, T+B+H, MLL, and E2A-PBX1, labeled as 1,2,3,4 respectively, by the three methods. Four colors (blue, orange, green, garnet) represent the estimated probabilities of test data belonging to class T-ALL, T+B+H, MLL, and E2A-PBX1.

Figure 3 plots the estimated class probability for acute lymphoblastic leukemia data set on 60 test data points with baseline method 2 (B2-SVM), OVA-SVM (A-SVM), and pairwise coupling (P-SVM), whose running time of 5 secs, 18 secs, and 13 secs. For both data sets, the two baseline methods and OVA-SVM have similar class probability estimations for each test data point as pairwise coupling, with no misclassification for any classes (we illustrated one baseline model for each data set). However, the baseline methods have the fastest computational time. With the appropriate variable selection method, our proposed linear algorithms and One-vs-All methods can work with scalable high-dimensional data with high accuracy on multiclass probability estimation and classification.

9 Concluding Remarks

In this article, we propose two new learning schemes, the baseline learning and the One-vs-All learning, to improve the computation time and estimation accuracy of the wSVMs methods. The One-vs-All method shows consistent best performance in probability estimation and classification with the most expensive computational cost, which is good for a small sample size with a class size $K \leqslant 5$. The two baseline methods have comparable performance to pairwise coupling method with great value in reducing the time complexity especially when class size K and sample size n is big, which makes it a favorable approach for multiclass probability estimation in real-world applications for both low-dimensional and high-dimensional data analysis.

We also establish the consistency theory for the baseline models and One-vs-All multiclass probability estimators, which is similar to the pairwise coupling method. The nature of the baseline method for training $K-1$ binary wSVMs, makes it enjoys the advantage of parallel computing and adapts the general-purpose computing on graphics processing units (GPGPU), which could further reduce the computational time and optimized for complex multiclass probability estimation problems. One interesting question is how to use our proposed wSVMs framework for image classification, such as brain tumor MRI images, for fast screening. Since the widely used image classification approaches based on Convolutional Neural Networks (CNNs) lack the accurate class probability estimation (Guo et al., 2017; Minderer et al., 2021), which is important for pre-cancer diagnosis to establish the strength of confidence for classification.

Currently, we only evaluate the dense feature set. The class probability estimation performance would decrease significantly if the feature sets are sparse with highly correlated predictor variables and abundant noisy features (as shown in the high-dimensional data sets in 8.2). Since all the methods discussed in this article are based on the L_2-norm regularization of the binary wSVMs optimization problem, we don't have the automatic variable selection power with the current framework. One of the important directions for future research is how to incorporate variable selection in the proposed class probability estimation framework for wSVMs, which is an important problem for data with sparse signals and highly correlated features. In many real applications, abundant features are collected but only a few of them have predictive power. We will this topic a more thorough treatment in future work.

Supplementary Material

We developed the R Package `MPEwSVMs` for multiclass probability estimation based on the proposed methods and have submitted it to the GitHub repository at https://github.com/zly555/MPEwSVMs. Detailed instructions on how to use the package and as well as a numerical simulation example are shown in the `README.md` file.

References

Alimoglu F, Alpaydin E (1997). Combining multiple representations and classifiers for pen-based handwritten digit recognition. In: *Proceedings of the Fourth International Conference on Document Analysis and Recognition* (J Schürmann, ed.), volume 2, 637–640. Ulm, Germany.

Alizadeh AA, Eisen MB, Davis RE, Ma C, Lossos IS, Rosenwald A, et al. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769): 503–511.

Breiman L, Friedman J H, Olshen R A, Stone C J (1984). *Classification and Regression Trees*. Wadsworth Publishing Company, Belmont, California, USA.

Burges C (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2: 121–167.

Cairano SD, Brand M, Bortoff SA (2013). Projection-free parallel quadratic programming for linear model predictive control. *International Journal of Control*, 86(8): 1367–1385.

Chamasemani FF, Singh YP (2011). Multi-class support vector machine (SVM) classifiers – an application in hypothyroid detection and classification. In: *Proceedings of the Sixth International Conference on Bio-Inspired Computing: Theories and Applications* (R Abdullah, ed.), 351–356. Penang, Malaysia.

Chen T, Guestrin C (2016). XGBoost: A scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (A Smola, C Aggarwal, D Shen, R Rastogi, eds.), In: *KDD '16*, 785–794. ACM, New York, New York, USA.

Crammer K, Singer Y (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2: 265–292.

Cristianini N, Shawe-Taylor J (2000). *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*. Cambridge University Press, Cambridge, UK.

Ding S, Zhao X, Zhang J, Zhang X, Xue Y (2019). A review on multi-class TWSVM. *Artificial Intelligence Review*, 52(2): 775–801.

Dua D, Graff C (2019). UCI machine learning repository. http://archive.ics.uci.edu/ml.

Dudoit S, Fridlyand J, Speed TP (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, 97(457): 77–87.

Guo C, Pleiss G, Sun Y, Weinberger KQ (2017). On calibration of modern neural networks. In: *Proceedings of the 34th International Conference on Machine Learning* (D Precup, YW Teh, eds.), volume 70, 1321–1330. Sydney, Australia.

Hastie T, Tibshirani R, Friedman J (2009). *The Elements of Statistical Learning: Data mining, Inference and Prediction*. Springer, New York, New York, USA. 2 edition.

Herbei R, Wegkamp MH (2006). Classification with reject option. *Canadian Journal of Statistics*, 34(4): 709–721.

Ho TK (1995). Random decision forests. In: *Proceedings of the Third International Conference on Document Analysis and Recognition* (CY Suen, ed.), volume 1, 278–282. Montreal, Canada.

Horton P, Nakai K (1996). A probabilistic classification system for predicting the cellular localization sites of proteins. In: *Proceeding of the Fourth International Conference on Intelligent Systems for Molecular Biology* (DJ States, P Agarwal, T Gaasterland, L Hunter, RF Smith, eds.), 109–115. St. Louis, Missouri, USA.

Huang H, Liu Y, Du Y, Perou CM, Hayes DN, Todd MJ, et al. (2013). Multiclass distance-weighted discrimination. *Journal of Computational and Graphical Statistics*, 22(4): 953–969.

Islam R, Khan SA, Jm K (2016). Discriminant feature distribution analysis-based hybrid feature selection for online bearing fault diagnosis in induction motors. *Journal of Sensors*, 2016: 1–16.

Kallas M, Francis C, Kanaan L, Merheb D, Honeine P, Amoud H (2012). Multi-class SVM classification combined with kernel PCA feature extraction of ECG signals. In: *Proceeding of the 19th International Conference on Telecommunications* (H Abumarshoud, A Shojaeifard, H Aghvami, F Marvasti, eds.), 1–5. Jounieh, Lebanon.

Kimeldorf G, Wahba G (1971). Some results on Tchebycheffian spline functions. *Journal of*

Mathematical Analysis and Applications, 33: 82–95.

Krawczyk B, Woźniak M, Cyganek B (2014). Clustering-based ensembles for one-class classification. *Information Sciences*, 264: 182–195.

Lee Y, Lin Y, Wahba G (2004). Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99: 67–81.

Lei Y, Dogan U, Binder A, Kloft M (2015). Multi-class SVMs: from tighter data-dependent generalization bounds to novel algorithms. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems* (C Cortes, DD Lee, M Sugiyama, R Garnett, eds.), volume 2, 2035–2043. Montreal, Canada.

Lin Y (2002). Support vector machines and the bayes rule in classification. *Data Mining and Knowledge Discovery*, 6: 259–275.

Liu Y (2007). Fisher consistency of multicategory support vector machines. In: *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics* (M Meila, X Shen, eds.), 291–298. San Juan, Puerto Rico.

Liu Y, Yuan M (2011). Reinforced multicategory support vector machine. *Journal of Computational and Graphical Statistics*, 20: 901–919.

McCullagh P, Nelder J (1989). *Generalized Linear Models*. Chapman and Hall, London, UK.

Mezzoudj F, Benyettou A (2012). On the optimization of multiclass support vector machines dedicated to speech recognition. In: *Proceedings of the 19th International Conference on Neural Information Processing* (T Huang, Z Zeng, C Li, CS Leung, eds.), volume 2, 1–8. Berlin, Germany.

Minderer M, Djolonga J, Romijnders R, Hubis F, Zhai X, Houlsby N, et al. (2021). Revisiting the calibration of modern neural networks. In: *Proceedings of the 35th Advances in Neural Information Processing Systems* (M Ranzato, A Beygelzimer, Y Dauphin, PS Liang, JW Vaughan, eds.), volume 34, 15682–15694.

Rifkin R, Klautau A (2004). In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5: 101–141.

Saigal P, Khanna V (2020). Multi-category news classification using support vector machine based classifiers. *SN Applied Sciences*, 2(3): 458.

Tomar D, Agarwal S (2015). A comparison on multi-class classification methods based on least squares twin support vector machine. *Knowledge-Based Systems*, 81: 131–147.

Vapnik V (1998). *Statistical Learning Theory*. Wiley, New York, New York, USA.

Wahba G (1990). *Spline Models for Observational Data CBMS-NSF Regional Conference Series in Applied Mathematics*. SIAM, Philadelphia, Pennsylvania, USA.

Wang J, Shen X (2006). Estimation of generalization error: random and fixed inputs. *Statistica Sinica*, 16(2): 569–588.

Wang J, Shen X, Liu Y (2008). Probability estimation for large margin classifiers. *Biometrika*, 95: 149–167.

Wang L, Shen X (2007). On L_1-norm multiclass support vector machines. *Journal of the American Statistical Association*, 102: 583–594.

Wang X, Zhang HH, Wu Y (2019). Multiclass probability estimation with support vector machines. *Journal of Computational and Graphical Statistics*, 28(3): 586–595.

Weston J, Watkins C (1999). Support vector machines for multi-class pattern recognition. In: *Proceedings of the Seventh European Symposium on Artificial Neural Networks* (M Gori, ed.), 21–23. Bruges, Belgium.

Wu Y, Zhang HH, Liu Y (2010). Robust model-free multiclass probability estimation. *Journal of the American Statistical Association*, 105: 424–436.

Ye Y, Tse E (1989). An extension of Karmarkar's projective algorithm for convex quadratic programming. *Mathematical Programming*, 44(1–3): 157–179.

Yeoh EJ, Ross ME, Shurtleff SA, Williams WK, Patel D, Mahfouz R, et al. (2002). Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell*, 1(2): 133–143.

Zhang C, Liu Y (2013). Multicategory large-margin unified machines. *Journal of Machine Learning Research*, 14: 1349–1386.

Zhu J, Hastie T (2005). Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14: 185–205.

Zhu J, Rosset S, Hastie T, Tibshirani R (2003). 1-norm support vector machines. In: *Proceedings of the 16th International Conference on Neural Information Processing Systems* (S Thrun, LK Saul, B Schölkopf, eds.), 49–56. Whistler, Canada.