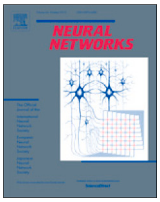
Neural Networks 131 (2020) 276–290

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet



SVM-Boosting based on Markov resampling: Theory and algorithm[☆]

Hongwei Jiang[a], Bin Zou[a,*], Chen Xu[b], Jie Xu[c,*], Yuan Yan Tang[d,1]

[a] Faculty of Mathematics and Statistics, Hubei Key Laboratory of Applied Mathematics, Hubei University, Wuhan 430062, China
[b] Department of Mathematics and Statistics, University of Ottawa, Ottawa, ON K1N 6N5, Canada
[c] Faculty of Computer Science and Information Engineering, Hubei University, Wuhan 430062, China
[d] Faculty of Science and Technology, University of Macau, China

ARTICLE INFO

Article history:
Received 12 May 2019
Received in revised form 8 June 2020
Accepted 31 July 2020
Available online 11 August 2020

Keywords:
Boosting
Consistency
Uniformly ergodic Markov chain (u.e.M.c.)
Resampling

ABSTRACT

In this article we introduce the idea of Markov resampling for Boosting methods. We first prove that Boosting algorithm with general convex loss function based on uniformly ergodic Markov chain (u.e.M.c.) examples is consistent and establish its fast convergence rate. We apply Boosting algorithm based on Markov resampling to Support Vector Machine (SVM), and introduce two new resampling-based Boosting algorithms: SVM-Boosting based on Markov resampling (SVM-BM) and improved SVM-Boosting based on Markov resampling (ISVM-BM). In contrast with SVM-BM, ISVM-BM uses the support vectors to calculate the weights of base classifiers. The numerical studies based on benchmark datasets show that the proposed two resampling-based SVM Boosting algorithms for linear base classifiers have smaller misclassification rates, less total time of sampling and training compared to three classical AdaBoost algorithms: Gentle AdaBoost, Real AdaBoost, Modest AdaBoost. In addition, we compare the proposed SVM-BM algorithm with the widely used and efficient gradient Boosting algorithm-XGBoost (eXtreme Gradient Boosting), SVM-AdaBoost and present some useful discussions on the technical parameters.

1. Introduction

Ensemble learning (Breiman, 1999; Dietterich, 2000) is a machine learning method that constructs multiple base learners and combines them with different weights to achieve higher prediction accuracy. According to the different ways of generating base learners, ensemble learning methods are mainly divided into two categories: Bagging (Breiman, 1996) and Boosting (Freund, 1995; Schapire, 1990). Bagging is an abbreviation for bootstrap aggregation (Efron & Tibshirani, 1993) and it can be dealt with using parallel method. It draws some subsets of a given training set, and the size of these subsets is same as that of the given training set. The base learners are obtained by training these subsets. While Boosting is to obtain base learners by adjusting the weights of training examples. The most famous Boosting method is AdaBoost (Adaptive Boosting), which was introduced by Freund and Schapire in Freund and Schapire (1996, 1997). Different from Bagging, the examples misclassified by the last base learner will receive more attention in the next train, and repeat the process above, until up to the given number of iterations. In Breiman (2000), Breiman proved that AdaBoost algorithm based on decision (complete) tree can convergence to the Bayes risk as the size of training examples is enough big. Jiang (2002, 2004) not only presented the examples that AdaBoost has prediction error asymptotically suboptimal as the number of iterations is enough big, but also pointed that some regularization methods may make the prediction error close to the Bayes risk when the size of training examples increases. Lugosi and Vayatis (2004) proved that the convex combination of base classifiers can close to the Bayes classifier under the condition of certain regularized assumptions. Zhang (2004) studied the consistency of Boosting by minimizing the convex risk of classification error function. Zhang and Yu (2005) showed that Boosting algorithm is consistent under the condition of early-stopping strategies. Bartlett and Traskin (2007) proved that the unmodified AdaBoost algorithm is consistent when it is stopped after $n^{1-\epsilon}$ iterations, where n is the size of training set and $\epsilon \in (0, 1)$. Chen and Guestrin (2016) proposed the Boosting algorithm based on gradient, XGBoost (eXtreme Gradient Boosting). In Chen and Guestrin (2016), they introduced a series of improvement methods such as regularization term,

[☆] This work is supported in part by NSFC project (61772011, 61977021, 61871177, 11690014), Open Project Foundation of Intelligent Information Processing Key Laboratory of Shanxi Province (No. CICIP2018002), and Natural Sciences and Engineering Research Council of Canada under Grant GPIN-2016-05024.

[*] Corresponding authors.
E-mail addresses: jhw940466281@163.com (H. Jiang), zoubin0502@gmail.com (B. Zou), cx3@uottawa.ca (C. Xu), frangipani@hubu.edu.cn (J. Xu), yytang@umac.mo (Y.Y. Tang).

[1] Fellow, IEEE.

https://doi.org/10.1016/j.neunet.2020.07.036
0893-6080/© 2020 Elsevier Ltd. All rights reserved.

second derivative of loss function, parallelism of feature granularity, and so on. Mukherjee and Schapire (2013) and Saberian and Vasconcelos (2019) studied multi-class Boosting algorithm. Lin, Lei, and Zhou (2019) considered Boosting algorithm based on kernel ridge regression and provided a new bias–variance trade-off method by adjusting the number of Boosting iterations.

With the advent of the high-tech era, the capacity of data is growing rapidly, and the value density of data is usually very low, which implies that there are many noise examples in big data. While the main idea of AdaBoost algorithm is to adjust the weights of training examples so that the examples misclassified by the last classifier will be focused in the next train. Thus AdaBoost algorithm will be very time-consuming or hard to implement as the size of data is very bigger. In addition, many experiments of machine learning indicate that the noise examples not only lead to increase the amount of storage space, but also affect the accuracy of learning. By the statistical learning theory in Vapnik (1998), we know that the most "important" examples for classification problems are the examples close to the interface of two classes data. Therefore, in this article we introduce the idea of Markov resampling for Boosting methods to sample a small amount training examples from this given data and then these examples are used to train the base classifiers. The main idea of Markov resampling proposed in this paper is to generate uniformly ergodic Markov chain (u.e.M.c.) examples for many times. In order to study systematically Boosting algorithm based on Markov resampling, we prove that Boosting algorithm with general convex loss function based on u.e.M.c. examples is consistent and establish its fast convergence rate. As an application, we also introduce a new SVM-Boosting algorithm based on Markov resampling (SVM-BM). Since the proposed SVM-BM algorithm uses all of the training examples to calculate the weights of base learners, this implies that SVM-BM algorithm will be time-consuming as the size of the given training set is bigger. To improve the proposed SVM-BM, we also introduce another new SVM-Boosting algorithm based on Markov resampling, the improved SVM-Boosting based on Markov resampling (ISVM-BM). Different from SVM-BM, the weights of base learners of ISVM-BM are calculated using the support vectors. The numerical studies based on benchmark datasets show that two SVM-Boosting algorithms based on Markov resampling proposed in this paper not only have smaller misclassification rates, but also have less sampling and training total time compared to three classical AdaBoost algorithms: Gentle AdaBoost (Friedman, Hastie, & Tibshirani, 1998), Real AdaBoost (Friedman et al., 1998) and Modest AdaBoost (Vezhnevets & Vezhnevets, 2008). Since there is only difference between SVM-BM and ISVM-BM in terms of calculating the weights of base classifiers. In other words, there is no significant difference between SVM-BM and ISVM-BM in terms of the misclassification rates, we also compare the proposed SVM-BM algorithm with the widely used and efficient gradient Boosting algorithm-XGBoost (eXtreme Gradient Boosting) algorithm (Chen & Guestrin, 2016) and SVM-AdaBoost (Schapire & Singer, 1999). In order to have a better understanding the proposed SVM-Boosting algorithms based on Markov resampling, we give some discussions on the technical parameters used in the proposed algorithms. We highlight some contributions of this paper.

- The Boosting algorithm with general convex loss function based on u.e.M.c. examples is proved to be consistent and its fast convergence rate is established.
- Two new SVM-Boosting algorithms based on Markov resampling, SVM-BM and ISVM-BM are proposed. The numerical experiments based on benchmark data show that the proposed algorithms have better classification performance compared to the classical AdaBoost, XGBoost and SVM-AdaBoost algorithms.

The rest of this paper is arranged as follows: Section 2 gives some definitions and symbols related to this article. Section 3 presents the main results on the consistency of Boosting algorithm with general convex loss function based on u.e.M.c. examples and establish its fast convergence rate. In Section 4, we apply Boosting algorithm based on Markov resampling to SVM and introduce two new SVM-Boosting algorithms based on Markov resampling. In Section 5, we present the experimental studies on the performance of the proposed two algorithms for linear kernel function and compare the proposed algorithms with the known Boosting algorithms. In Section 6, we give some discussions on the parameters involved in our algorithms and explain the performance of the proposed two algorithms. Finally, Section 7 concludes this paper.

2. Preliminaries

We give some symbols and definitions needed in this article.

2.1. Notations and definitions

Let $\mathcal{X}$ be a compact metric space and $\mathcal{Y} = \{-1, +1\}$. ρ is an unknown probability distribution on $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ and the corresponding random variable is $Z = (X, Y)$. The goal of learning is to find a classifier $\hat{f} : \mathcal{X} \to \mathcal{Y}$ based on a given training set such that if new objects are given, the classifier $\hat{f}$ will forecast them correctly. The performance of classifier $\hat{f}$ is evaluated by the misclassification rate, which is defined by the probability of the event $\{\hat{f}(X) \neq Y\}$, $L(\hat{f}) = \mathrm{P}\{\hat{f}(X) \neq Y\}$. We hope the misclassification rate $L(\hat{f})$ to be as small as possible and approach the Bayes risk $L^* = \inf_f L(f)$, where the infimum is taken over all possible functions. The corresponding Bayes classifier is defined as $f_c := \mathrm{sign}(f_\rho)$ (Devroye, Györfi, & Lugosi, 1997), which satisfies $L^* = L(f_c)$. Here f_ρ is the regression function of ρ, which is defined as $f_\rho = \int_{\mathcal{Y}} y d\rho(y|x), x \in \mathcal{X}$. The function $\mathrm{sign}(\cdot)$ is defined as $\mathrm{sign}(f) = 1$ if $f \geq 0$ and $\mathrm{sign}(f) = -1$ otherwise.

Boosting algorithm consists of two main steps: a series of base classifiers are generated by some algorithm, and then these base classifiers are combined. The base classifiers often depend on a Reproducing Kernel Hilbert Space (RKHS) associated with a Mercer kernel (Aronszajn, 1950). A Mercer kernel $K : X \times X \to \mathbb{R}$ is a continuous, symmetric and positive semidefinite function. The RKHS $\mathcal{H}_K$ associated with the kernel K completed by the set of functions $\{K_x = K(x, \cdot) : x \in \mathcal{X}\}$ with the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}_K} = \langle \cdot, \cdot \rangle_K$ satisfying $\langle K_x, K_y \rangle_K = K(x, y)$ and $\langle K_x, g \rangle_K = g(x), \forall x \in \mathcal{X}, \forall g \in \mathcal{H}_K$ (Evgeniou, Pontil, & Poggio, 2000). Let $\kappa := \sup_{x \in \mathcal{X}} \sqrt{K(x, x)}$, then the above reproducing property tells us that $\|g\|_\infty \leq \kappa \|g\|_K, \forall g \in \mathcal{H}_K$.

Let $\phi(g, z) = \phi(yg(x))$ be a strictly convex loss function, then the tth base classification function g_t of Boosting is defined by the following regularization scheme involving a training set $\mathbf{z} := \{z_i = (x_i, y_i)\}_{i=1}^{\mathrm{N}} \in \mathcal{Z}^{\mathrm{N}}$,

$$g_t = g_{t,\mathbf{z}} = \arg \min_{g \in \mathcal{H}_K} \left\{ R_{\phi,\mathrm{N}}(g) + \lambda \|g\|_K^2 \right\}, \tag{1}$$

where $R_{\phi,\mathrm{N}}(g) = \frac{1}{\mathrm{N}} \sum_{i=1}^{\mathrm{N}} \phi(g, z_i)$ is the empirical risk of g and λ is the regularization parameter which depends on $\mathrm{N} : \lambda = \lambda(\mathrm{N})$, and often satisfies $\lim_{\mathrm{N} \to \infty} \lambda(\mathrm{N}) = \infty$. The corresponding expectation risk of $\phi(g, z)$ is defined as $R_\phi(g) = \mathbb{E}[\phi(g, z)]$, where $\mathbb{E}[u]$ is the expectation of u.

Define $\mathcal{F}^T$ as the set of T-combinations of classification function $g_t (t \in \mathbb{N}^+)$ in $\mathcal{H}_K$

$$\mathcal{F}^T = \left\{ f = \sum_{t=1}^{T} \alpha'_t g_t | T \in \mathbb{N}, \sum_{i=1}^{T} \alpha'_t = 1, \alpha'_t \geq 0, \alpha'_t \in \mathbb{R} \right\}.$$

For any $f \in \mathcal{F}^T$, the l_*-norm is defined by

$$\|f\|_* = \inf\Big\{\sum |\alpha_t'|, f = \sum \alpha_i' g_i, g_i \in \mathcal{H}_K\Big\}.$$

For $\phi(g, z)$, we need the following properties (Bartlett & Traskin, 2007):
(i) Let $\phi(g, z)$ be a differentiable, strictly convex and increasing function such that

$$\phi(0) = 1, \quad \lim_{yg(x)\to\infty} \phi(yg(x)) = 0.$$

(ii) Denote the upper bound of $\phi(g, z)$ to be

$$M_{\phi,yg(x)} = \sup_{yg(x)} |\phi(yg(x))|. \tag{2}$$

(iii) The Lipschitz constant L_ϕ of $\phi(g, z)$ is defined as

$$L_\phi = \inf_{yg}\Big\{L|L > 0, |\phi(g_1, z) - \phi(g_2, z)| \le L|g_1 - g_2|\Big\}. \tag{3}$$

Different from the classical Boosting algorithm, AdaBoost (see, Bartlett & Traskin, 2007; Freund & Schapire, 1996, 1997), in this paper the base classification functions g_t ($t \in \mathbb{N}^+$) of Boosting are obtained by training u.e.M.c. examples, which are drawn from a given training set by using the method of Markov resampling.

2.2. u.e.M.c.

Suppose $(\mathcal{Z}, \mathcal{D})$ is a measurable space, a Markov chain is a sequence of random variables $\{Z_t\}_{t\ge1}$ together with a set of transition probability $\mathrm{P}^m(\mathcal{A}|z_i)$, for $\mathcal{A} \in \mathcal{D}, z_i \in \mathcal{Z}$,

$$\mathrm{P}^m(\mathcal{A}|z_i) := \mathrm{P}\{Z_{m+i} \in \mathcal{A}|Z_j, j < i, Z_i = z_i\}.$$

Thus $\mathrm{P}^m(\mathcal{A}|z_i)$ denotes the transition probability that the state z_{m+i} will belong to the set $\mathcal{A}$ after m time steps, starting from the initial state z_i at time i. The fact that the transition probability does not depend on the values of Z_j prior to time i is the Markov property, that is $\mathrm{P}^m(\mathcal{A}|z_i) = \mathrm{P}\{Z_{m+i} \in \mathcal{A}|Z_i = z_i\}$, which is expressed as "given the present state (Z_i), the future state (Z_{m+i}) is conditionally independent of the past state (Z_j)". Given two probabilities P_1 and P_2 on the measure space $(\mathcal{Z}, \mathcal{D})$, the total variation distance between the probabilities P_1 and P_2 is defined as $d_{TV}(P_1, P_2) := \sup_{\mathcal{A}\in\mathcal{D}} |P_1(\mathcal{A}) - P_2(\mathcal{A})|$. Thus we have the following definition of u.e.M.c. (Vidyasagar, 2003).

Definition 1 (*Vidyasagar, 2003*)**.** Let $\{Z_t\}_{t\ge1}$ be a Markov chain, if there exist two constants $0 < \gamma_0 < \infty$ and $0 < \varphi < 1$ such that

$$d_{TV}(\mathrm{P}^m(\cdot|z), \pi(\cdot)) \le \gamma_0\varphi^m, \forall m \ge 1, m \in \mathbb{N},$$

where $\pi(\cdot)$ is the stationary distribution of $\{Z_t\}_{t\ge1}$. We say $\{Z_t\}_{t\ge1}$ is uniformly ergodic.

Remark 1. A Markov chain $\{Z_t\}_{t\ge1}$ is u.e.M.c. (Qian & Gong, 1998), if the size of state space of Markov chain $\{Z_t\}_{t\ge1}$ is finite, and the transition probabilities of any two states are always positive.

3. Consistency and learning rate

Let D_{train} be a given training set, and T be the number of iterations. The Boosting procedure based on Markov resampling can be described as follows:

1. For a given training set D_{train}, let $f_0 = 0$.
2. Draw randomly N training examples from D_{train} and denote it D_0. Train D_0 by algorithm (1) and obtain an initial classification function g_0.
3. For $t = 1, \ldots, T$, draw N Markov chain examples from D_{train} and denote it D_t, these examples in D_t are drawn randomly from D_{train} and accepted with the corresponding probabilities $\mathrm{P}(Z_{i+1}|Z_i) = p_t^{i+1}(Z_i, Z_{i+1}, g_{t-1}, \phi)$, which are the function of Z_i, Z_{i+1}, g_{t-1} and ϕ. Train D_t by algorithm (1) and obtain the classification function g_t.
4. For $t = 1, 2, \ldots, T$, set $f_t = f_{t-1} + \alpha_t g_t$, where $\alpha_t \ge 0$ is the weight of g_t.
5. Output the final classifier $\mathrm{sign}(f_T) = \mathrm{sign}(\sum_{t=1}^T \alpha_t g_t)$.

By Remark 1, we have that if the acceptance probabilities $p_t^{i+1}(Z_i, Z_{i+1}, g_{t-1}, \phi)$ between Z_i and Z_{i+1} are always positive, then the Markov chain examples sequence in D_t is u.e.M.c. since the size of the given training set D_{train} is finite. The main idea of Markov resampling is to generate the u.e.M.c. examples sets $D_t (1 \le t \le T)$ and these samples sets $D_t (1 \le t \le T)$ are used to train the corresponding classification functions $g_t (1 \le t \le T)$.

To bound the learning performance of the classifier $\mathrm{sign}(f_T)$, we should estimate the excess misclassification rate $L(\mathrm{sign}(f_T)) - L^*$. According to Theorem 1 of Bartlett, Jordan, and McAuliffe (2006), for a nondecreasing function $\psi(\nu) : [0, 1] \to [0, \infty)$, we have

$$\psi[L(\mathrm{sign}(f_T)) - L^*] \le R_\phi(f_T) - R_\phi(f_\rho), \tag{4}$$

$$\psi(\nu) \to 0 \quad \text{implies that} \quad \nu \to 0. \tag{5}$$

In particular, if $\phi(g, z)$ is the hinge loss $\ell(g, z) := \max\{1 - yg(x), 0\}$, by Example 4 of Bartlett et al. (2006), we have $\psi(\nu) = |\nu|$, and

$$L(\mathrm{sign}(f_T)) - L^* \le R_\phi(f_T) - R_\phi(f_\rho). \tag{6}$$

Thus by inequality (4) and condition (5), we should estimate the excess ϕ-risk $R_\phi(f_T) - R_\phi(f_\rho)$ in order to estimate the excess misclassification rate $L(\mathrm{sign}(f_T)) - L^*$. Note that f_T satisfies $\mathrm{sign}(f_T) = \mathrm{sign}(\tilde{f}_T)$, where $\tilde{f}_T = \frac{1}{\|f_T\|_*} f_T \in \mathcal{F}^T$, then we replace the function f_T by $\tilde{f}_T$. In order to study the consistency of Boosting algorithm based on u.e.M.c. examples, we need to estimate $R_\phi(\tilde{f}_T) - R_\phi(f_\rho)$. Thus we firstly establish the following propositions.

Proposition 1. *Let $\mathcal{H}_K$ with measurable square integrable envelope $H(x)$ such that $\int H^2 d\rho_{\mathcal{X}} < \infty$. If the space $\mathcal{H}_K$ has a finite VC-index, $V(\mathcal{H}_K) = U$, then we have that there exist an exponent r with $r > 0$ and a constant $\tilde{C} > 0$*

$$\ln \mathcal{N}(B_1, \varepsilon) \le \tilde{C}(1/\varepsilon)^r, \ \forall\, 0 < \varepsilon < 1,$$

where $\tilde{C} = K_1 U(16e)^U(\|H\|_{\rho_{\mathcal{X}},2})^r$, $r = 2(U + 1)/U$ and K_1 is constant, which is defined in Lemma 3 of Appendix A.

Proposition 2. *Let $D_t = \{z_i\}_{i=1}^{\mathrm{N}}$ be u.e.M.c. examples and $g_\lambda =$* $\arg\min_{g\in\mathcal{H}_K} \big\{R_\phi(g) + \lambda\|g\|_K^2\big\}$*, then there exists a constant $\varsigma > 0$ such that for any $\delta \in (0, 1)$, with confidence at least $1 - \delta$,*

$$\begin{aligned} R_\phi(g_t) - R_\phi(g_\lambda) + \lambda\|g_t\|_K^2 - \lambda\|g_\lambda\|_K^2 &\le \frac{2}{\sqrt{\lambda}}\Big(\frac{112C_\varsigma(\kappa + 1)\|\varGamma_0\|^2}{\mathrm{N}}\Big)^{1/1+\varsigma} + \frac{1}{2}D(\lambda) \\ &+ \frac{56\ln(1/\delta)(\kappa\sqrt{D(\lambda)/\lambda} + B)\|\varGamma_0\|^2}{\mathrm{N}}, \end{aligned}$$

where $D(\lambda) = R_\phi(g_\lambda) - R_\phi(f_\rho) + \lambda\|g_\lambda\|_K^2$.

Proposition 1 will be proven in Appendix B. For proof of Proposition 2, see Corollary 1 of Xu, et al. (2015). Inspired by the idea from Bartlett and Traskin (2007) and Wu, Ying, and Zhou (2006), we decompose the excess ϕ-risk $R_\phi\big(\tilde{f}_T\big) - R_\phi(f_\rho)$ as follows:

Proposition 3. *For $\tilde{f}_T \in \mathcal{F}^T$, and the definition of g_λ, the excess ϕ-risk $R_\phi\big(\tilde{f}_T\big) - R_\phi(f_\rho)$ can be decomposed as*

$$R_\phi(\tilde{f}_T) - R_\phi(f_\rho) = \big\{\mathbb{S}_1 + \mathbb{S}_2 + \mathbb{S}_3\big\} + D(\lambda), \tag{7}$$

where

$$\mathbb{S}_1 = R_\phi(\tilde{f}_T) - R_{\phi,n}(\tilde{f}_T),$$
$$\mathbb{S}_2 = R_{\phi,n}(\tilde{f}_T) - R_{\phi,n}(g_\lambda) - \lambda\|g_\lambda\|_K^2,$$
$$\mathbb{S}_3 = R_{\phi,n}(g_\lambda) - R_\phi(g_\lambda).$$

In Proposition 3, the excess ϕ-risk $R_\phi\big(\tilde{f}_T\big) - R_\phi(f_\rho)$ is decomposed into two terms. The first term is called the sample error, the second term $D(\lambda)$ is called the approximation error. By Wu et al. (2006), we have that the optimal ϕ-risk $R_\phi(f_\rho)$ can be approximated by $\mathcal{H}_K$. That is, for two constants $0 < s \le 1$ and $C_s > 0$, we have $D(\lambda) \le C_s\lambda^s$, $\forall\lambda > 0$.

In this paper, we suppose that there exists a constant B such that $|y| \le B$ (see Wu et al., 2006). Then by the definition of f_ρ, we have $|f_\rho| \le B$. Thus we have Propositions 4 and 5, which will be proven in Appendix B.

Proposition 4. *Let $D = \bigcup_{t=1}^T D_t$ be u.e.M.c. examples, $D_t = \{z_i\}_{i=1}^{\mathrm{N}}$, and the size n of D satisfy $n = \mathrm{N} * T$. Then we have that for any $0 < \delta < 1$, with confidence at least $1 - \delta$,*

$$R_{\phi,n}(g_\lambda) - R_\phi(g_\lambda) \le \Big(\frac{56M_{\phi,d}^2\|\Gamma_0\|^2\ln(1/\delta)}{n}\Big)^{1/2},$$

where $M_{\phi,d} := M_{\phi,yg_\lambda(x)}$.

Proposition 5. *Under the same conditions as Proposition 4, we have that for all $n \ge n^*$ and any $0 < \delta < 1$, with confidence at least $1 - \delta$,*

$$\sup_{f\in\mathcal{F}^T}\Big|R_\phi(f) - R_{\phi,n}(f)\Big| \le \Big(\frac{112\|\Gamma_0\|^2\tilde{C}M_{\phi,1}^2L_\phi^r}{n}\Big)^{1/(2+r)},$$

where $n^* = 112M_{\phi,1}^2\|\Gamma_0\|^2(\ln(2/\delta))^{(2+r)/r}\tilde{C}^{-2/r}L_\phi^{-1}$, $\tilde{C} = K_1U(16e)^U$ $(\|H\|_{\rho_{\mathcal{X}},2})^r$ and $r = 2(U+1)/U$ are constants.

Our main results can be stated as follows:

Theorem 1. *Let $D = \bigcup_{t=1}^T D_t$ be u.e.M.c. examples, and the size n of D satisfy $n = \mathrm{N} * T$. We have that for any $0 < \delta < 1$, the inequality*

$$\mathbb{E}[R_\phi(\tilde{f}_T) - R_\phi(f_\rho)] \le \frac{2}{\sqrt{\lambda}}\Big(\frac{112C_\varsigma(\kappa+1)\|\Gamma_0\|^2T}{n}\Big)^{1/1+\varsigma}$$
$$+ \frac{(\tau-1)L_\phi GT}{n(T-1)} + \frac{56\ln(3/\delta)(\kappa\sqrt{C_s}\lambda^{(s-1)/2} + B)\|\Gamma_0\|^2T}{n}$$
$$+ \Big(\frac{56M_{\phi,d}^2\|\Gamma_0\|^2\ln(3/\delta)}{n}\Big)^{1/2} + L_\phi G\gamma_0\varphi^\tau + \frac{3}{2}C_s\lambda^s$$
$$+ \Big(\frac{112\|\Gamma_0\|^2\tilde{C}M_{\phi,1}^2L_\phi^r}{n}\Big)^{1/(2+r)}$$

is valid with confidence at least $1 - \delta$ provided that the size n of D satisfies $n \ge 112M_{\phi,1}^2\|\Gamma_0\|^2(\ln(6/\delta))^{(2+r)/r}\tilde{C}^{-2/r}L_\phi^{-1}$.

If we choose λ and τ such that

$$\lambda \to 0, \frac{\lambda^{(s-1)/2}}{n} \to 0, \frac{\lambda^{-1/2}}{n^{1/1+\varsigma}} \to 0, \frac{M_{\phi,d}}{\sqrt{n}} \to 0, \text{ as } n \to \infty,$$

and $\tau \to \infty$, $\tau/n \to 0$, as $n \to \infty$. By Theorem 1, inequality (4) and the fact that $n \to \infty$ as $T \to \infty$, we have

Theorem 2. *Under the same conditions as Theorem 1, we have that as $T \to \infty$,*

$$\psi[L(\mathrm{sign}(f_T)) - L^*] \to 0.$$

This implies that Boosting algorithm based on u.e.M.c. examples with general convex loss function is consistent.

In particular, if $\phi(g,z)$ is the hinge loss, that is, $\phi(g,z) = \ell(g,z)$, and τ, λ satisfy $\tau = \log_\varphi(n^{-\theta})(\theta > 0)$, $\lambda = n^{-\beta}(0 < \beta < 1)$, by Theorem 1, we also obtain the following convergence rate for SVM-Boosting based on u.e.M.c. examples.

Theorem 3. *Under the same conditions as Theorem 1, we have that for any $0 < \beta < \min\{1, 2/(1+\varsigma)\}$ and any $\delta \in (0,1)$, the inequality*

$$\mathbb{E}[L(\mathrm{sign}(f_T)) - L(f_c)] \le \Theta\Big(\frac{\ln T}{T}\Big)$$

is valid with confidence at least $1 - \delta$ provided $T \ge (448\|\Gamma_0\|^2(\ln(6/\delta))^{(2+r)/r}\tilde{C}^{-2/r})/N$, where Θ is a constant defined in Appendix C.

Theorems 1 and 3 will be proved in Appendix C. To have a better understanding our main results obtained in Theorems 1–3, we give the following remarks.

Remark 2. Comparing Theorems 1–3 with the corresponding results obtained in Bartlett and Traskin (2007) and Lugosi and Vayatis (2004), we can find that although our error decomposition on the excess ϕ-risk (in Proposition 3) is similar to that of Bartlett and Traskin (2007), the differences are obvious: First, Bartlett and Traskin (2007) and Lugosi and Vayatis (2004) considered the classical AdaBoost based on independent and identically distributed (i.i.d.) examples, and Lugosi and Vayatis (2004) studied the consistency of regularized Boosting method while Theorem 2 established in this article is on the consistency of Boosting algorithm based on non-i.i.d. examples, u.e.M.c. examples. Second, the proof process of Theorem 1 is different from Bartlett and Traskin (2007) since we use a new bound of covering number. In addition, In Theorems 2–3, we not only proved that the Boosting algorithm based on u.e.M.c. examples is consistent (Theorem 2) and established the fast learning rate of Boosting algorithm based on u.e.M.c. samples (Theorem 3). To my knowledge, these results are the first results on this topic.

4. Algorithms

To study the learning performance of Boosting algorithm based on Markov resampling, we apply it to SVM with linear kernel function and introduce two new Boosting algorithms: SVM-Boosting based on Markov Resampling (SVM-BM) and Improved SVM-Boosting based on Markov Resampling (ISVM-BM). Notice that SVM with linear kernel is the special case of algorithm (1) with $\phi(g,z) = \ell(g,z)$, that is

$$g_t = \arg\min_{g\in\mathcal{H}_K}\Big\{\frac{1}{\mathrm{N}}\sum_{i=1}^{\mathrm{N}}\ell(g,z_i) + \lambda\|g\|_K^2\Big\}, \quad z_i \in D_t. \tag{8}$$

4.1. SVM-BM algorithm

SVM-BM algorithm can be described as follows.

To have a better understanding Algorithm 1, we provide the following remarks.

Remark 3. (i) Since we have only the data D_{train} (the distribution of examples is unknown), to define the transition probabilities p_t^{i+1} of Markov resampling, in Algorithm 1 we first draw randomly training set D_0 from D_{train} and obtain an initial classification function g_0 by training SVM algorithm (8) with D_0. And For $1 \le t \le T$, the transition probabilities p_t^{i+1} used to generate the examples in D_t is based on the model g_{t-1}, which is different

Algorithm 1: SVM-BM

```
Input: $D_{train}$, $n_2$, $q$, N, $T$
Output: $\mathrm{sign}(f_T) = \mathrm{sign}(\sum_{t=1}^{T} \alpha_t g_t)$
Draw randomly samples $D_0 = \{z_i\}_{i=1}^{\mathrm{N}}$ from $D_{train}$, train $D_0$ by algorithm (8) and
obtain a classification function $g_0$, draw randomly a sample $z$ from $D_{train}$,
$z_1 \leftarrow z$, let $t \leftarrow 1$
while $t \leq T$ do
    $i \leftarrow 1$, $n_1 \leftarrow 0$
    while $i \leq \mathrm{N}$ do
        Draw randomly a sample $z_*$ from $D_{train}$,
        $p_t^{i+1} \leftarrow \min\{1, e^{-\ell(g_{t-1}, z_*)}/e^{-\ell(g_{t-1}, z_i)}\}$
        if $n_1 > n_2$ then
            $p_t^{i+1} \leftarrow \min\{1, qp_t^{i+1}\}$, $z_i \leftarrow z_*$, $D_t \leftarrow z_i$, $i \leftarrow i+1$, $n_1 \leftarrow 0$
        end
        if $p_t^{i+1} \equiv 1$ and $y_* y_i = 1$ then
            $p_t^{i+1} \leftarrow e^{-y_* g_{t-1}}/e^{-y_i g_{t-1}}$
        end
        if rand(1) $< p_t^{i+1}$ then
            $z_i \leftarrow z_*$, $D_t \leftarrow z_i$, $i \leftarrow i+1$, $n_1 \leftarrow 0$
        end
        if $z_*$ is not accepted then
            $n_1 \leftarrow n_1 + 1$
        end
    end
    Obtain Markov chain $D_t = \{z_i\}_{i=1}^{\mathrm{N}}$, train $D_t$ by algorithm (8) and obtain
    another classification function $g_t$.
    $e_t \leftarrow \mathrm{P}\big(Y \neq \mathrm{sign}(g_t(X)) | D_{train}\big)$,
    $\alpha_t \leftarrow (1/2) * \log\big((1 - e_t)/e_t\big)$,
    $z_1 \leftarrow z_*$,  $t \leftarrow t + 1$
    if $\alpha_t < 0$ then
        $t \leftarrow t - 1$
    end
end
```

Algorithm 2: ISVM-BM

```
Input: $D_{train}$, $n_2$, $q$, N, $T$
Output: $\mathrm{sign}(f_T) = \mathrm{sign}(\sum_{t=1}^{T} \hat{\alpha}_t g_t)$
Draw randomly samples $D_0 = \{z_i\}_{i=1}^{\mathrm{N}}$ from $D_{train}$, train $D_0$ by algorithm (8) and
obtain a classification function $g_0$, draw randomly a example $z$ from $D_{train}$ and
$z_1 \leftarrow z$, let $t \leftarrow 1$
while $t \leq T$ do
    $i \leftarrow 1$, $n_1 \leftarrow 0$
    while $i \leq \mathrm{N}$ do
        Draw randomly a sample $z_*$ from $D_{train}$,
        $p_t^{i+1} \leftarrow \min\{1, e^{-\ell(g_{t-1}, z_*)}/e^{-\ell(g_{t-1}, z_i)}\}$
        if $n_1 > n_2$ then
            $p_t^{i+1} \leftarrow \min\{1, qp_t^{i+1}\}$, $z_i \leftarrow z_*$, $D_t \leftarrow z_i$, $i \leftarrow i+1$, $n_1 \leftarrow 0$
        end
        if $p_t^{i+1} \equiv 1$ and $y_* y_i = 1$ then
            $p_t^{i+1} \leftarrow e^{-y_* g_{t-1}}/e^{-y_i g_{t-1}}$
        end
        if rand(1) $< p_t^{i+1}$ then
            $z_i \leftarrow z_*$, $D_t \leftarrow z_i$, $i \leftarrow i+1$, $n_1 \leftarrow 0$
        end
        if $z_*$ is not accepted then
            $n_1 \leftarrow n_1 + 1$
        end
    end
    Obtain Markov chain $D_t = \{z_i\}_{i=1}^{\mathrm{N}}$. Train $D_t$ by algorithm (8) and obtain
    another classification function $g_t$. Denote support vectors as $D_{SV}^{t}$.
    $e'_t \leftarrow \mathrm{P}\big(Y \neq \mathrm{sign}(g_t(X)) | \cup_{j=1}^{t} D_{SV}^{j}\big)$, $\hat{\alpha}_t \leftarrow (1/2) * \log\big((1 - e'_t)/e'_t\big)$,
    $z_1 \leftarrow z_*$,  $t \leftarrow t + 1$
    if $\hat{\alpha}_t < 0$ then
        $t \leftarrow t - 1$
    end
end
```

from MCMC method since MCMC (Geman & Geman, 1984) is a sampling method of using the information distribution of training examples. (ii) By the statistical learning theory (Vapnik, 1998), we know that the examples that are close to the interface of two classes data are the most "important" examples for classification problem, so we hope that these examples close to the interface can be drawn and accepted with high probability. However, if the loss $\ell(g_{t-1}, z_i)$ of the current examples z_i is smaller, the acceptance probability p_t^{i+1} of the candidate examples z_* will be smaller, which implies that generating the u.e.M.c examples in D_t will be time-consuming. To draw quickly these u.e.M.c examples in $D_t (t = 1, 2, \ldots, T)$, we use two technical parameters q and n_2 inspired by the idea from Xu, et al. (2015). We present some discussions on the parameters q, n_2 in the next section. (iii) In Algorithm 1, we did not require the condition that the size of $+1$ class in training examples sets $D_t (0 \leq t \leq T)$ is equal to that of -1 class. That is, Markov resampling defined in Algorithm 1 improves the Markov sampling method in Xu, et al. (2015) and extends it from the case of balanced training examples to the case of unbalanced training examples.

4.2. ISVM-BM algorithm

For SVM algorithm with linear kernel function, the optimal base classification function g_t can be expressed as

$$g_t = \sum_i \omega_i y_i x_i' x + b, \quad z_i = (x_i, y_i) \in D_t, \tag{9}$$

where x_i' is the transpose matrix of x_i. In (9), the vectors x_i that correspond to $\omega_i \neq 0$ are called to be support vector (Vapnik, 1998). Express (9) is said to be "more sparse" (Laarhouen & Aarts, 1987) if the number of support vector in express (9) is smaller. This implies that SVM has nice properties for compressing the training examples set in the form (9) of support vectors (Vapnik, 1998).

Since the support vectors are the most "important" examples for classification problem and the total support vector number of all base classifiers is usually far less than the size of D_{train}, we introduce the idea of using the support vectors to calculate the weights of base classifiers, and present another new algorithm, ISVM-BM. Let "D_{SV}^{j}" be the support vectors of the jth ($1 \leq j \leq T$) base classification g_t. The ISVM-BM algorithm can be stated as follows (see Algorithm 2).

Remark 4. (i) In Algorithm 1, we use the whole training set D_{train} to calculate the weight α_t of base classification function g_t. As the size of the given training set D_{train} is large, calculating the weights α_t of base classification functions g_t will be time-consuming. Different from Algorithm 1, we use the first tth total support vectors $\cup_{j=1}^{t} D_{SV}^{j}$ to calculate the weight $\hat{\alpha}_t$ of the tth base classification function g_t. Since the size of the first tth total support vectors $\cup_{j=1}^{t} D_{SV}^{j}$ is usually smaller than that of the given training set D_{train}. This implies that Algorithm 2 is an improved version of Algorithm 1. This improvement is just what we can expect as reflected in our experimental results presented in Tables 7–9.

(ii) Let the size of D_{train} and the maximum depth of the tree be S and h, respectively. The time complexity of AdaBoost is about $O(ThSd)$ for a normal style the weak learner such as CART (Breiman, Friedman, Olshen, & Stone, 1984), where d is the input dimension of D_{train} and T is the number of base classifiers. The time complexity of XGBoost (Chen & Guestrin, 2016) is about $O(Th\|S\|_0 + \|S\|_0 log\mathbb{B})$, where $\|S\|_0$ denotes the number of non-missing entries in the given training set D_{train} and $\mathbb{B}$ is the maximum number of rows in each block. Since the time complexity of SVM with N training examples is about $O(\mathrm{N}^{\tilde{q}})$ (Burges, 1998), where $\tilde{q}$ is a constant satisfying $2 < \tilde{q} < 3$. The time complexity of SVM-BM is approximately $O\big(T\mathrm{N}^{\tilde{q}}(1 + S)\big)$, and the time complexity of ISVM-BM is approximately $O\big(T\mathrm{N}^{\tilde{q}}(1 + |\cup_{j=1}^{T} D_{SV}^{j}|)\big)$, where $|\cup_{j=1}^{T} D_{SV}^{j}|$ is the size of $\cup_{j=1}^{T} D_{SV}^{j}$. Comparing the time complexities of AdaBoost, XGBoost, SVM-BM and ISVM-BM, we can find that for a given data D_{train} with large size S

Table 1
9 real-world datasets.

Dataset	$\sharp D_{train}$	$\sharp D_{test}$	♯ Input dimension
Cod-rnd	325 710	162 855	8
Poker	768 757	256 253	10
Seismic	73 896	24 632	50
Connect4	50 668	16 889	126
W7a	33 166	16 583	300
HAPT	7767	2589	561
Isbi	360 000	135 600	1681
TV-news	86 457	43 228	4125
Gisette	6000	6000	5000

and the same number T of base classification functions, the time complexities of SVM-BM and ISVM-BM are smaller than those of AdaBoost and XGBoost as N $\ll S$, where N is the size of training examples used to train the base classification function.

5. Experiments and comparisons

We present a comparative experiment comparing our two algorithms with four algorithms: three classical AdaBoost algorithms (Gentle AdaBoost Friedman et al., 1998, Real AdaBoost Friedman et al., 1998, Modest AdaBoost Vezhnevets & Vezhnevets, 2008), XGBoost (Chen & Guestrin, 2016) and SVM-AdaBoost (Schapire & Singer, 1999). All the experiments of comparing SVM-BM, ISVM-BM with three classical AdaBoost algorithms are implemented on an Intel(R) Xeon (R) CPU E5-2650 2.20 GHz PC, 32 GB RAM with Matlab R2018a, while all the experiments of comparing SVM-BM with XGBoost are implemented on an Intel(R) Xeon (R) CPU E5-2650 2.20 GHz PC, 32 GB RAM with Python 3.7.

5.1. Datasets and parameters choice

The numerical studies are based on the following 9 real-world datasets: Cod-rnd (https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/), Poker, Seismic, Connect4, W7a, HAPT, TV-news and Gisette are available from the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/index.php), and Isbi (https://grand-challenge.org/challenges/). For every dataset, we randomly break it down into two parts: a training set D_{train} and a test set D_{test}. Table 1 describes the properties of the selected datasets.

We simply state our experimental procedure as follows:

(i) For the given training set D_{train}, we train three classical AdaBoost algorithms, XGBoost, SVM-BM and ISVM-BM, respectively. We then test the obtained classifiers on the given test set D_{test} and calculate the corresponding misclassification rates (MR), MR is defined as

$$\mathrm{MR} = \frac{\mathrm{FP}+\mathrm{FN}}{\mathrm{FP}+\mathrm{FN}+\mathrm{TN}+\mathrm{TP}},$$

where TP is the number of true positives, TN is the number of true negatives, and FP and FN are the number of false positives and false negatives.

(ii) Combine the training set D_{train} and the test set D_{test}, and break it randomly down into two parts: a new training set D'_{train} and a new test set D'_{test}. The sizes of D'_{train} and D'_{test} are same as that of D_{train} and D_{test}, respectively.

(iii) We repeat procedures (i)–(ii) for $k-1$-times, where k is the number of (repeat) experiments. And then we compute the standard deviations, means of misclassification rates and the total time (sampling and training) of k-times for the above algorithms.

For simplicity, we use "G-AB", "R-AB", "M-AB", "X-GB", "SVM-BM" and "ISVM-BM" to denote the corresponding experimental results of Gentle AdaBoost, Real AdaBoost, Modest AdaBoost, XGBoost, SVM-Boosting based on Markov resampling and improved SVM-Boosting based on Markov resampling, respectively. All these experimental results are based on $k = 50$ except for Isbi (since the size of Isbi is very big, the experimental results of Isbi are based on $k = 10$).

The experimental results presented in this paper are based on linear kernel function, and the regularization parameters of SVM in SVM-BM and ISVM-BM are selected from (0.001, 0.01, 0.1, 1, 100, 1000) by the method of 5-fold cross-validation. The parameter n_2 is chosen from (5, 10, 20, 30), and the parameter q is chosen from (1.1, 1.3, 1.5, 1.7). We set the parameter booster of XGBoost to be the gbtree model, and the depth of estimator is 3 and define the number T of trees as 'estimator'. The other parameters are chosen by the method of 5-fold cross-validation, that is, 'eta' is chosen from $(0.01, 0.1, 0.3)$, 'alpha' is chosen from $(0.001, 0.01, 0.1, \ldots, 100)$, 'gamma' is chosen from $(0.1, 0.2, \ldots, 0.5)$, 'min-child-weight' is chosen from $(4, 5, 6)$, 'colsample-bytree' and 'subsample' are chosen from $(0.5, 0.6, \ldots, 1)$, respectively.

5.2. Comparisons with the classical AdaBoost

In this section, we compare SVM-BM and ISVM-BM with the three classical AdaBoost algorithms: Gentle AdaBoost (G-AB), Real AdaBoost (R-AB), Modest AdaBoost (M-AB) for the case of $T = 10, 20, 30$, where T is the number of iterations or the number of base classification functions.

5.2.1. Comparison of misclassification rates

In Tables 2–4, we present the mean, the standard deviations of misclassification rates for k-times experiments.

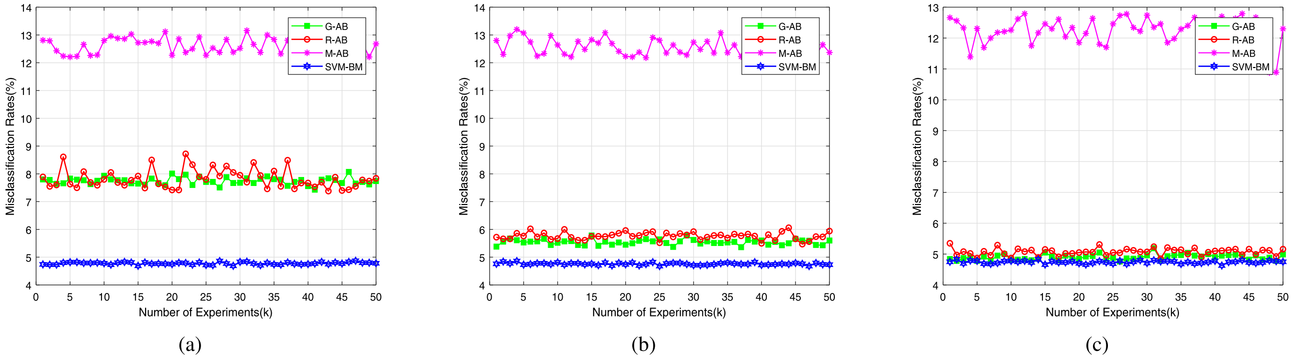
From Tables 2–4, we can find that the means of misclassification rates of SVM-BM (or ISVM-BM) are obviously smaller than those of three classical AdaBoost algorithms for $T = 10, 20$ and 30. And the standard deviations of misclassification rates of SVM-BM (or ISVM-BM) are smaller than those of three classical AdaBoost algorithms except for Seismic with $T = 10, 20, 30$, W7a with $T = 30$ and Isbi with $T = 30$.

In Tables 5 and 6, we use the Wilcoxon signed-rank test (Wilcoxon, 1945) to find out whether there exist significant differences between three classical AdaBoost algorithms, SVM-BM and ISVM-BM based on the means of misclassification rates presented in Tables 2–4.

By Tables 5 and 6, we can find that for $T = 10, 20, 30$, there are significant differences between our two algorithms and three classical AdaBoost algorithms, which implies that SVM-BM and ISVM-BM have better performance than three classical AdaBoost algorithms. But there is no significant difference between SVM-BM and ISVM-BM in terms of the misclassification rates.

To display more intuitively the learning performance of SVM-BM, we also present the (repeat) k-times misclassification rates in Figs. 1–9 (because there is no significant difference between SVM-BM and ISVM-BM in terms of the misclassification rates, we only show the experimental results of SVM-BM in Figs. 1–9). Here "blue hexagram", "red circle", "magenta-star" and "green-square" denote the experimental results of "M-AB", "R-AB", "G-AB" and "SVM-BM" respectively. The numbers on the horizontal axis, the vertical axis are the number k of repeat experiments and the misclassification rates, respectively.

By Figs. 1–9, we can clearly find that almost all the k-times misclassification rates of SVM-BM are smaller than those of three classical AdaBoost algorithms for $T = 10, 20$. And almost all the k-times misclassification rates of SVM-BM are smaller than those of Modest AdaBoost (M-AB) for $T = 30$ except for Poker dataset.



(a)

(b)

(c)

Fig. 1. *k*-times ($k = 50$) misclassification rates for Cod-rnd dataset. (a) N = 1000, $T = 10$; (b) N = 1000, $T = 20$; (c) N = 1000, $T = 30$.

Table 2
Average misclassification rates (%) for $T = 10$.

Dataset	G-AB	R-AB	M-AB	SVM-BM	ISVM-BM
Cod-rnd	7.74 ± 0.13	7.82 ± 0.34	12.62 ± 0.29	**4.78** ± **0.04** (N = 1000)	**4.78** ± 0.05 (N = 1000)
Poker	**0.37** ± **0.01**	0.86 ± 3.52	0.47 ± 0.16	**0.37** ± **0.01** (N = 1000)	**0.37** ± **0.01** (N = 1000)
Seismic	16.06 ± 0.23	16.04 ± **0.22**	17.18 ± 0.29	15.41 ± 0.33 (N = 1000)	**15.38** ± 0.30 (N = 1000)
Connect4	25.18 ± 0.51	25.06 ± 0.47	31.39 ± 0.34	**21.16** ± **0.25** (N = 1500)	**21.16** ± 0.29 (N = 1500)
W7a	2.50 ± 0.14	2.39 ± 0.14	2.58 ± 0.12	**1.74** ± **0.10** (N = 2500)	**1.74** ± 0.11 (N = 2500)
HAPT	0.83 ± 0.16	0.85 ± 0.22	6.29 ± 0.71	0.22 ± **0.09** (N = 1500)	**0.21** ± 0.10 (N = 1500)
Isbi	14.67 ± 0.09	14.81 ± 0.22	15.07 ± **0.08**	**13.33** ± 0.10 (N = 15 000)	13.34 ± 0.11 (N = 15 000)
TV-news	13.34 ± 0.23	13.46 ± 0.29	14.80 ± 0.25	**11.29** ± **0.13** (N = 6000)	**11.29** ± 0.17 (N = 6000)
Gisette	5.02 ± 0.42	5.11 ± 0.41	8.84 ± 0.38	**2.25** ± 0.19 (N = 2500)	**2.25** ± **0.17** (N = 2500)

Table 3
Average misclassification rates (%) for $T = 20$.

Dataset	G-AB	R-AB	M-AB	SVM-BM	ISVM-BM
Cod-rnd	5.54 ± 0.09	5.76 ± 0.13	12.61 ± 0.29	4.76 ± **0.04** (N = 1000)	**4.75** ± 0.05 (N = 1000)
Poker	**0.37** ± **0.01**	**0.37** ± **0.01**	**0.37** ± 0.03	**0.37** ± **0.01** (N = 1000)	**0.37** ± **0.01** (N = 1000)
Seismic	15.56 ± 0.25	15.51 ± **0.22**	16.17 ± 0.25	15.01 ± 0.28 (N = 1000)	**15.00** ± 0.25 (N = 1000)
Connect4	22.59 ± 0.41	22.54 ± 0.43	31.50 ± 0.39	**20.87** ± **0.29** (N = 1500)	20.95 ± 0.39 (N = 1500)
W7a	2.23 ± 0.14	2.00 ± 0.13	2.59 ± 0.14	1.72 ± 0.12 (N = 2500)	**1.68** ± **0.10** (N = 2500)
HAPT	0.34 ± 0.12	0.39 ± 0.11	3.35 ± 0.54	0.19 ± **0.08** (N = 1500)	**0.18** ± **0.08** (N = 1500)
Isbi	13.69 ± 0.14	13.66 ± 0.12	14.94 ± 0.08	**13.10** ± 0.08 (N = 15 000)	13.17 ± **0.06** (N = 15 000)
TV-news	11.85 ± 0.19	11.86 ± 0.17	13.83 ± 0.20	11.17 ± **0.14** (N = 6000)	**11.16** ± 0.16 (N = 6000)
Gisette	3.24 ± 0.24	3.34 ± 0.30	6.49 ± 0.34	**2.22** ± **0.18** (N = 2500)	**2.22** ± **0.18** (N = 2500)

Table 4
Average misclassification rates (%) for $T = 30$.

Dataset	G-AB	R-AB	M-AB	SVM-BM	ISVM-BM
Cod-rnd	4.91 ± 0.08	5.08 ± 0.10	12.13 ± 0.44	**4.74** ± **0.04** (N = 1000)	**4.74** ± 0.05 (N = 1000)
Poker	**0.37** ± **0.01**	**0.37** ± **0.01**	**0.37** ± 0.02	**0.37** ± **0.01** (N = 1000)	**0.37** ± **0.01** (N = 1000)
Seismic	15.32 ± 0.22	15.21 ± 0.23	15.85 ± **0.19**	14.83 ± 0.27 (N = 1000)	**14.81** ± 0.25 (N = 1000)
Connect4	21.16 ± 0.33	21.08 ± 0.42	31.21 ± 0.39	**20.85** ± **0.28** (N = 1500)	20.88 ± 0.32 (N = 1500)
W7a	2.02 ± 0.13	1.80 ± 0.11	2.58 ± **0.09**	1.69 ± 0.11 (N = 2500)	**1.65** ± 0.10 (N = 2500)
HAPT	0.25 ± 0.09	0.26 ± 0.09	2.92 ± 0.60	**0.18** ± **0.07** (N = 1500)	**0.18** ± 0.10 (N = 1500)
Isbi	13.18 ± 0.21	13.18 ± 0.09	14.92 ± **0.07**	**13.05** ± 0.09 (N = 15 000)	13.06 ± **0.07** (N = 15 000)
TV-news	11.22 ± 0.15	11.26 ± 0.16	13.35 ± 0.20	**11.11** ± 0.13 (N = 6000)	**11.11** ± **0.12** (N = 6000)
Gisette	2.40 ± 0.28	2.41 ± 0.27	5.29 ± 0.30	2.21 ± **0.20** (N = 2500)	**2.20** ± 0.23 (N = 2500)

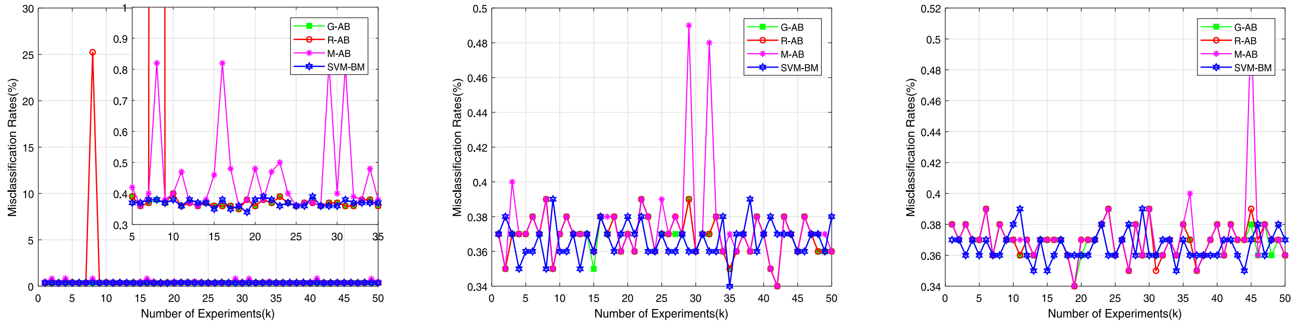
5.2.2. Comparison of sampling and training total time

In Tables 7–9, we compare the total time of sampling and training of three classical AdaBoost algorithms, SVM-BM and ISVM-BM for *k*-times repeat experiments.

By Tables 7–9, we can find that the sampling and training total time of *k*-times experiments of SVM-BM (or ISVM-BM) is less compared to three classical AdaBoost algorithms for $T = 10, 20, 30$. And the sampling and training total time of ISVM-BM are less than those of SVM-BM, the reason of which has been presented in Remark 4.

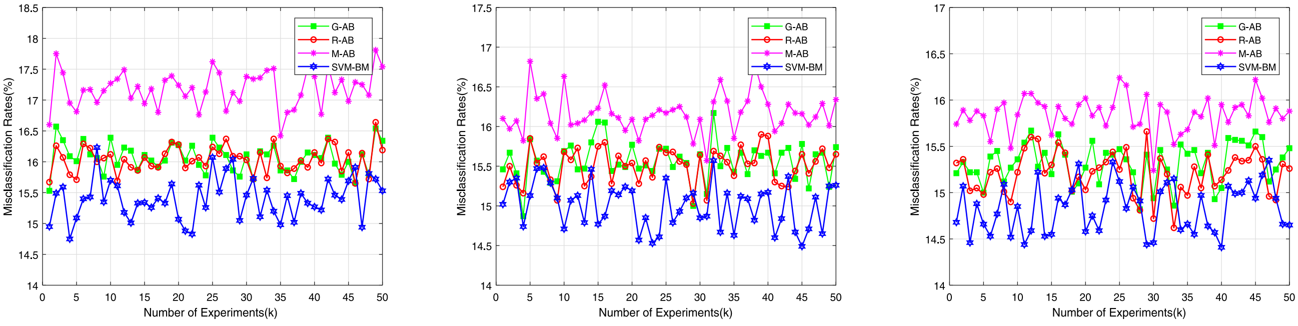
5.3. Comparison with the XGBoost introduced in Chen and Guestrin (2016)

In this section, we compare the proposed algorithms with XGBoost. Since there is no significant difference between SVM-BM and ISVM-BM, we only compare the experimental results of SVM-BM with that of XGBoost. Different from these experimental results presented in the last subsection (see Tables 2–4), we adjust the value of N of SVM-BM.



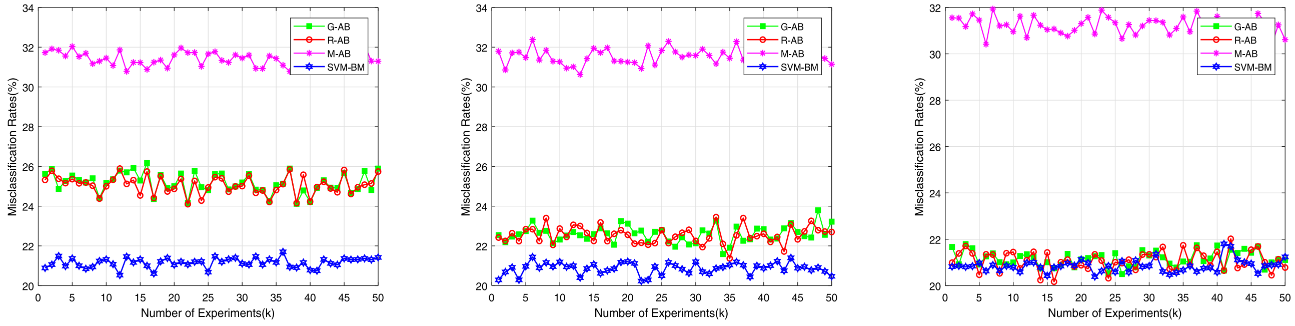
(a) (b) (c)

Fig. 2. *k*-times ($k = 50$) misclassification rates for Poker dataset. (a) N = 1000, $T = 10$; (b) N = 1000, $T = 20$; (c) N = 1000, $T = 30$.



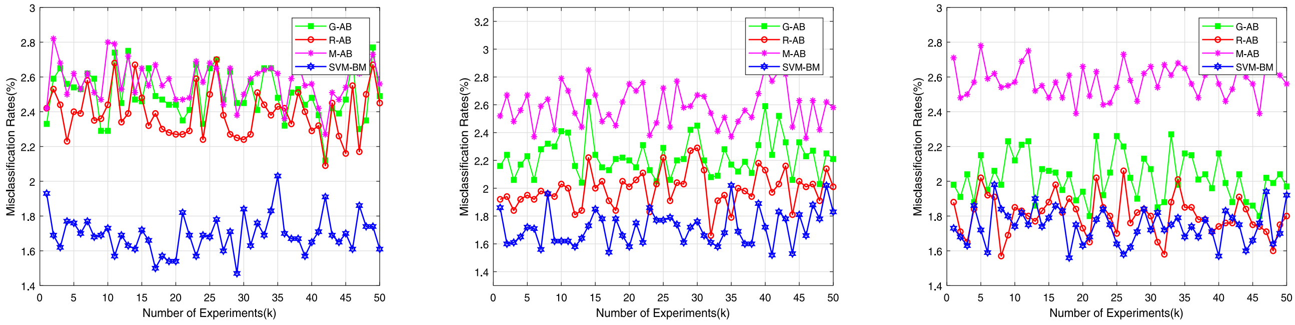
(a) (b) (c)

Fig. 3. *k*-times ($k = 50$) misclassification rates for Seismic dataset. (a) N = 1000, $T = 10$; (b) N = 1000, $T = 20$; (c) N = 1000, $T = 30$.



(a) (b) (c)

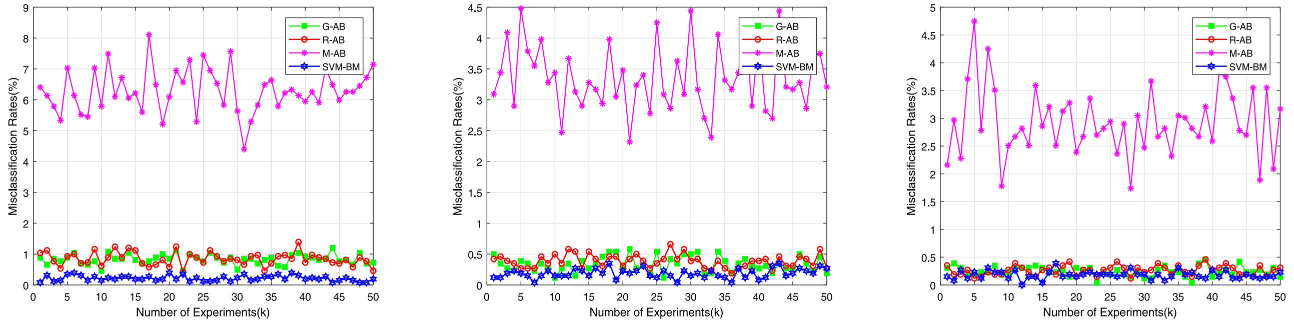
Fig. 4. *k*-times ($k = 50$) misclassification rates for Connect4 dataset. (a) N = 1500, $T = 10$; (b) N = 1500, $T = 20$; (c) N = 1500, $T = 30$.



(a) (b) (c)

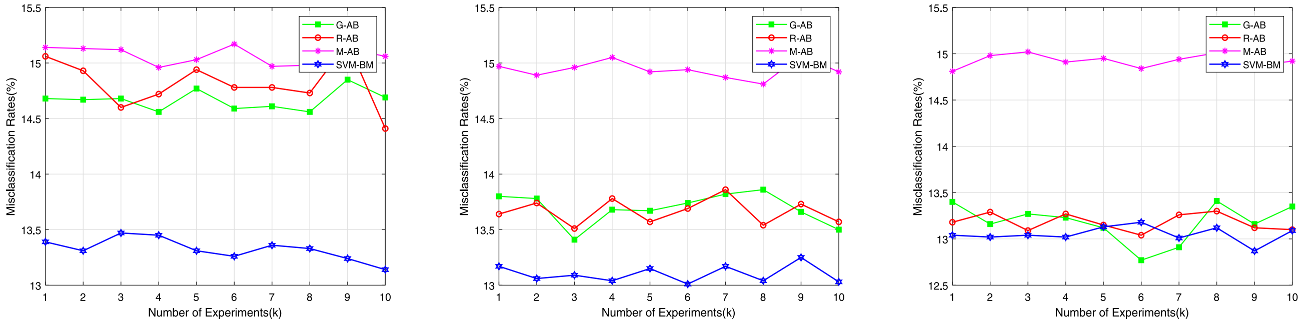
Fig. 5. *k*-times ($k = 50$) misclassification rates for W7a dataset. (a) N = 2500, $T = 10$; (b) N = 2500, $T = 20$; (c) N = 2500, $T = 30$.

By Table 10, we can find that for $T = 10$, 20 and 30, almost all the means of misclassification rates of SVM-BM are smaller than those of XGBoost, except for Isbi with $T = 20$, 30, TV-news with $T = 20$, 30, and Connect4 with $T = 30$.



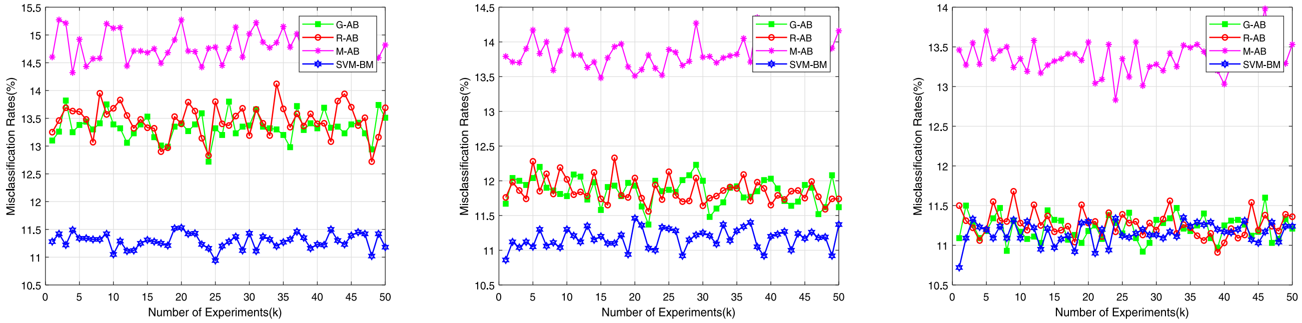
(a) (b) (c)

Fig. 6. k-times ($k = 50$) misclassification rates for HAPT dataset. (a) N = 1500, $T = 10$; (b) N = 1500, $T = 20$; (c) N = 1500, $T = 30$.



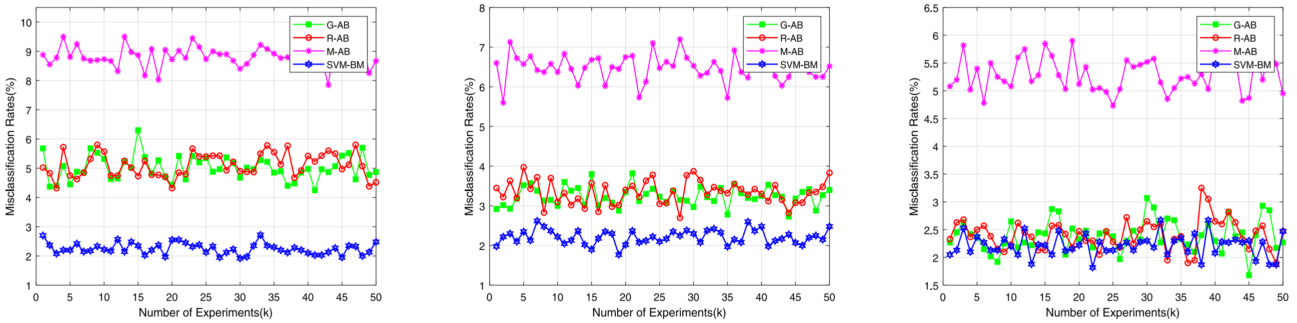
(a) (b) (c)

Fig. 7. k-times ($k = 10$) misclassification rates for Isbi dataset. (a) N = 15000, $T = 10$; (b) N = 15000, $T = 20$; (c) N = 15000, $T = 30$.



(a) (b) (c)

Fig. 8. k-times ($k = 50$) misclassification rates for TV-news dataset. (a) N = 6000, $T = 10$; (b) N = 6000, $T = 20$; (c) N = 6000, $T = 30$.



(a) (b) (c)

Fig. 9. k-times ($k = 50$) misclassification rates for Gisette dataset. (a) N = 2500, $T = 10$; (b) N = 2500, $T = 20$; (c) N = 2500, $T = 30$.

Table 5
Wilcoxon tests of G-AB, R-AB, M-AB and SVM-BM.

T	Comparison	R_+	R_-	Hypothesis ($\alpha = 0.05$)	Selected
10	SVM-BM vs G-AB	0.5	44.5	Rejected	SVM-BM
	SVM-BM vs R-AB	0	45	Rejected	SVM-BM
	SVM-BM vs M-AB	0	45	Rejected	SVM-BM
20	SVM-BM vs G-AB	0.5	44.5	Rejected	SVM-BM
	SVM-BM vs R-AB	0.5	44.5	Rejected	SVM-BM
	SVM-BM vs M-AB	0.5	44.5	Rejected	SVM-BM
30	SVM-BM vs G-AB	0.5	44.5	Rejected	SVM-BM
	SVM-BM vs R-AB	0.5	44.5	Rejected	SVM-BM
	SVM-BM vs M-AB	0.5	44.5	Rejected	SVM-BM

Table 6
Wilcoxon tests of G-AB, R-AB, M-AB, SVM-BM and ISVM-BM.

T	Comparison	R_+	R_-	Hypothesis ($\alpha = 0.05$)	Selected
10	ISVM-BM vs G-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs R-AB	0	45	Rejected	ISVM-BM
	ISVM-BM vs M-AB	0	45	Rejected	ISVM-BM
	ISVM-BM vs SVM-BM	17.5	27.5	Not Rejected	ISVM-BM
20	ISVM-BM vs G-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs R-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs M-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs SVM-BM	18.5	26.5	Not Rejected	ISVM-BM
30	ISVM-BM vs G-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs R-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs M-AB	0.5	44.5	Rejected	ISVM-BM
	ISVM-BM vs SVM-BM	18.5	26.5	Not Rejected	ISVM-BM

Table 7
Sampling and training total time (s) for $T = 10$.

Dataset	G-AB	R-AB	M-AB	SVM-BM	ISVM-BM
Cod-rnd	695.63	698.86	726.54	61.75	**60.18**
Poker	1432.1	1794.2	707.58	28.55	**25.15**
Seismic	879.89	878.24	899.20	472.86	**463.93**
Connect4	887.48	888.17	851.62	363.87	**356.68**
W7a	1508.69	1493.21	1524.22	838.47	**765.06**
HAPT	1056.8	1078.1	930.27	141.64	**135.9**
Isbi	25 387	25 546	26 830	25 254	**19 301**
TV-news	44 392	44 039	42 969	21 122.99	**14 570**
Gisette	5469.79	5295.47	5039.22	4039.40	**3953**

Table 8
Sampling and training total time (s) for $T = 20$.

Dataset	G-AB	R-AB	M-AB	SVM-BM	ISVM-BM
Cod-rnd	1411.2	1404.6	1500.7	121.37	**118.31**
Poker	3460.4	3846.6	1433.5	55.31	**42.63**
Seismic	1818.3	1814.8	1864.1	**624.70**	632.65
Connect4	1801.6	1801.4	1539.6	667.25	**664.40**
W7a	2902.63	2898.75	2971.48	1495.60	**1462.90**
HAPT	2167.1	2188.6	1896.6	281.48	**278.56**
Isbi	53 880	53 298	56 953	33 951	**27 742**
TV-news	93 040	92 135	92 645	28 611.20	**23 750**
Gisette	10 991	10 834	10 643	8230.60	**7932.20**

Table 9
Sampling and training total time (s) for $T = 30$.

Dataset	G-AB	R-AB	M-AB	SVM-BM	ISVM-BM
Cod-rnd	2084.5	2082	2243	178.7	**173.29**
Poker	5227	5577.2	2032.9	82.2	**66.05**
Seismic	2611.2	2608.4	2648	793.22	**782.18**
Connect4	2666.5	2660.7	2190.2	983.66	**967.53**
W7a	4377.2	4382.4	4506.88	2168.3	**2165.4**
HAPT	3173.5	3201.7	2796.7	423.36	**415.68**
Isbi	77 379	78 000	83 004	38 952	**37 090**
TV-news	131 925	131 330	134 250	35 807.26	**31 071**
Gisette	15 900.8	15 933.67	15 906.46	12 353	**11 783**

By Table 11, we can find that the sampling and training total time of SVM-BM are less or close to that of XGBoost for Cod-rnd, Poker, Seismic, Connect4 and HAPT datasets.

5.4. Comparison with the SVM-AdaBoost introduced in Schapire and Singer (1999)

Since there is no significant difference between SVM-BM and ISVM-BM, we also compare the proposed SVM-BM with the AdaBoost introduced in Schapire and Singer (1999), which uses SVM as a base learner (SVM-AB). Different from these experimental results presented in the last sections, we adjust the value of N of the SVM-BM. Since the SVM-AB (Schapire & Singer, 1999) uses all the examples in the given training set to train the corresponding classifier, the SVM-AB algorithm exceeds the memory of our current operating environment or runs out of memory for Cod-rnd, Poker, Seismic, Isbi and TV-news datasets. We then present the experimental results of Connect4, W7a, HAPT, Gisette in Table 12. Since the SVM-AB is very time-consuming for Connect4, W7a datasets, we only present the experimental results of HAPT, Gisette in Table 13. These experimental results presented in Tables 12 and 13 are based on 5-times repetitive experiments, where "MR", "Time" denotes the average misclassification rates, the sampling and training total time of 5-times repetitive experiments, respectively.

By Tables 12 and 13, we can find that for $T = 10$ or 20, all the means of average misclassification rates of the SVM-BM are smaller than those of the SVM-AB, and the sampling and training total time of the SVM-BM are less than those of the SVM-AB. These experiments imply that SVM classifiers obtained by training all the examples in the given training set are not suitable to be base learners of AdaBoost algorithm when the size of the given training set is larger.

6. Discussions and explanations

In this section, we first give some discussions on the choice of parameters n_2, q, N (for simplicity, we only give the discussions on the choice of parameters n_2, q, N for the proposed SVM-BM algorithm with part datasets since SVM-BM is similar to ISVM-BM except for the method of calculating the weights of base classifiers). We then give some explanations on the learning performance of the proposed algorithms.

6.1. Choices of n_2 and q

In Table 14, we present the experimental results of SVM-BM algorithm based on different n_2 with $T = 10$, $q = 1.3$ for W7a, TV-news datasets, where n_2 is chosen from $(5, 10, 20, 30)$. In Table 15, we present the experimental results of SVM-BM based on different q with $T = 10$, $n_2 = 5$ for Seismic, Gisette dataset, where q is chosen from $(1.1, 1.3, 1.5, 1.7)$. The experimental results are based on 20-times repeat experiments.

Table 14 shows that the misclassification rates of SVM-BM have a tendency of decrease, while the sampling and training total time of SVM-BM has a tendency of increase as n_2 increases. Table 15 shows that the misclassification rates of SVM-BM have a tendency of increase, while the sampling and training total time of SVM-BM has a tendency to decrease as q increases. To have a tread-off between the misclassification rate and the sampling and training total time, we choose $n_2 = 5$ and $q = 1.3$ for the experimental results presented in the last section.

Table 10
Average misclassification rates (%) of X-GB and SVM-BM.

Dataset	$T = 10$		$T = 20$		$T = 30$	
	X-GB	SVM-BM	X-GB	SVM-BM	X-GB	SVM-BM
Cod-rnd	7.78 ± 0.17	**4.80** ± **0.03** (N = 600)	5.68 ± 0.09	**4.77** ± **0.06** (N = 600)	4.76 ± 0.07	**4.76** ± **0.04** (N = 600)
Poker	**0.37** ± **0.01**	**0.37** ± **0.01** (N = 600)	**0.37** ± **0.01**	**0.37** ± 0.02 (N = 600)	**0.37** ± **0.01**	**0.37** ± **0.01** (N = 600)
Seismic	16.92 ± **0.23**	**15.52** ± 0.41 (N = 600)	15.79 ± **0.20**	**14.91** ± 0.23 (N = 600)	15.29 ± 0.19	**14.85** ± **0.10** (N = 600)
Connect4	25.23 ± 0.43	**21.80** ± **0.32** (N = 600)	22.14 ± 0.37	**21.43** ± **0.35** (N = 600)	**20.56** ± 0.34	21.11 ± **0.22** (N = 600)
W7a	2.71 ± **0.14**	**2.25** ± 0.16 (N = 1500)	2.28 ± **0.11**	**2.24** ± 0.23 (N = 1500)	2.23 ± **0.11**	**2.20** ± 0.17 (N = 1500)
HAPT	0.96 ± 0.19	**0.36** ± **0.07** (N = 700)	0.52 ± 0.16	**0.33** ± **0.07** (N = 700)	0.29 ± 0.12	**0.22** ± **0.04** (N = 700)
Isbi	14.21 ± 0.12	**13.22** ± **0.05** (N = 10 000)	**12.69** ± **0.09**	13.11 ± 0.14 (N = 10 000)	**12.05** ± 0.09	13.08 ± **0.06** (N = 10 000)
TV-news	12.15 ± 0.14	**11.66** ± **0.12** (N = 3000)	**10.81** ± 0.17	11.39 ± **0.13** (N = 3000)	**10.11** ± **0.13**	11.35 ± **0.13** (N = 3000)
Gisette	4.53 ± 0.35	**2.42** ± **0.22** (N = 1500)	3.00 ± 0.25	**2.34** ± **0.09** (N = 1500)	2.22 ± 0.24	**2.21** ± **0.13** (N = 1500)

Table 11
Sampling and training total time (s) of X-GB and SVM-BM.

Dataset	$T = 10$		$T = 20$		$T = 30$	
	X-GB	SVM-BM	X-GB	SVM-BM	X-GB	SVM-BM
Cod-rnd	143	**113**	258	**197**	393	**290**
Poker	**159**	239	**362**	513	**481**	786
Seismic	159	**143**	362	**241**	481	**369**
Connect4	161	**94**	256	**194**	381	**270**
W7a	**202**	2037	**422**	3827	**505**	5682
HAPT	178	**172**	**299**	351	**341**	538
Isbi	**3349**	13 691	**5446**	26 063	**10 330**	38 476
TV-news	**8674**	10 148	**14 882**	19 231	**17 793**	28 346
Gisette	**914**	1509	**1399**	2859	**1840**	4234

Table 12
Experimental results of SVM-BM and SVM-AB for $T = 10$.

Dataset	MR(%)		Time (s)	
	SVM-AB	SVM-BM	SVM-AB	SVM-BM
Connect4	20.76 ± 0.37	**20.55** ± **0.10** (N = 8000)	1 065 855	**2532**
W7a	1.31 ± 0.04	**1.31** ± **0.02** (N = 18 000)	228 447	**71 001**
HAPT	0.08 ± 0.04	**0.06** ± **0.04** (N = 4000)	214	**101**
Gisette	1.37 ± 0.24	**1.35** ± **0.18** (N = 4500)	1366	**1140**

Table 13
Experimental results of SVM-BM and SVM-AB for $T = 20$.

Dataset	MR (%)		Time (s)	
	SVM-AB	SVM-BM	SVM-AB	SVM-BM
HAPT	0.05 ± 0.05	**0.02** ± **0.03** (N = 4000)	420	**201**
Gisette	1.33 ± 0.30	**1.27** ± **0.08** (N = 4500)	2783	**2143**

Table 14
Average misclassification rates (MR) (%), sampling and training total time (Time) (s).

n_2	W7a (N = 2500)		TV-news (N = 3000)	
	MR	Time	MR	Time
5	1.74 ± 0.07	275.9	11.67 ± 0.17	1337.2
10	1.74 ± 0.10	281.7	11.63 ± 0.16	1348.0
20	1.72 ± 0.12	291.0	11.61 ± 0.19	1379.8
30	1.71 ± 0.13	304.8	11.60 ± 0.19	1399.3

Table 15
Average misclassification rates (MR) (%), sampling and training total time (Time) (s).

q	Seismic (N = 1000)		Gisette (N = 2500)	
	MR	Time	MR	Time
1.1	15.34 ± 0.31	177.0	2.23 ± 0.16	1753.7
1.3	15.35 ± 0.26	172.6	2.25 ± 0.19	1737.7
1.5	15.37 ± 0.29	168.8	2.25 ± 0.23	1700.9
1.7	15.40 ± 0.27	165.2	2.32 ± 0.22	1687.0

Table 16
Experimental results of Cod-rnd for different N.

N	400	600	1000	2000	3000	4000	5000
MR$_0$(%)	4.60	4.59	4.64	4.57	4.69	4.69	4.72
MR$_1$(%)	5.26	5.05	4.92	4.83	4.82	4.80	4.78
d_{MR}(%)	0.66	0.45	0.29	0.27	0.13	0.11	0.06
Time	1.11	1.5	2.85	8.03	26.35	66.13	117.96

Table 17
Experimental results of Connect4 for different N.

N	400	600	1000	1500	2000	3000	4000	5000	10 000
MR$_0$(%)	15.48	16.13	18.15	18.81	19.34	19.77	19.95	20.05	20.15
MR$_1$(%)	26.15	24.42	23.07	22.36	21.84	21.43	21.23	21.08	20.75
d_{MR}(%)	10.68	8.29	4.92	3.55	2.49	1.66	1.28	1.03	0.60
Time	3.2	6.5	16.3	33.4	54.2	165.4	336.9	594.6	4056

Table 18
Experimental results of Isbi for different N.

N	3000	5000	8000	10 000	15 000	20 000	40 000
MR$_0$(%)	4.67	7.27	8.95	9.24	10.69	11.00	11.78
MR$_1$(%)	17.06	16.22	15.33	14.94	14.50	14.21	13.61
d_{MR}(%)	12.39	8.95	6.38	5.70	3.81	3.21	1.86
Time	241.5	1049.6	3737.6	6723.8	20 446.3	43 087.9	238 504.6

Table 19
The choice range of N for different Datasets.

Dataset	N	Dataset	N	Dataset	N
Cod-rnd	[600, 4000]	Connect4	[600, 3000]	Isbi	[5000, 20 000]
Poker	[600, 5000]	W7a	[1000, 5000]	TV-news	[2000, 10 000]
Seismic	[600, 4000]	HAPT	[600, 3000]	Gisette	[1000, 3000]

6.2. *Choice of* N

In this subsection, we give some discussions on the choice of N. In Tables 16–18, we give the experimental results of SVM-BM for different N with $n_2 = 5$ and $q = 1.3$. Here we use “MR$_0$”, “MR$_1$” to denote the misclassification rates of the classifier sign(g_0) for the training set D_0, D_{train}, respectively, and use “Time”, “d_{MR}” to denote the sampling and training total time, the difference between “MR$_0$” and “MR$_1$”, respectively. $D_0 = \{z_i\}_{i=1}^{\mathrm{N}}$ is drawn randomly from the given training set D_{train}, and g_0 is obtained by algorithm (8) with D_0. The experimental results are based on 20-times repetitive experiments.

By Tables 16–18, we can find that as N increases, MR$_1$ has a tendency to decrease, but the sampling and training total time increases obviously. In addition, as N increases, the difference d_{MR} between “MR$_0$” and “MR$_1$” decreases, which implies that the difference of the base classifiers decreases. This is inconsistent with the idea of ensemble learning. That is, as the differences between different base classifiers are obvious, the final ensemble classifier will have good learning performance. Thus we present the choice range of N for 9 datasets in Table 19 according to the different values d_{MR} of different datasets.

We suggest the choice of N as follows: if sampling and training total time is a major concern, we should set N to be a small value. If the misclassification rate is a major concern, we should set N to be a big value. If we want to have a trade-off between the misclassification rate and the training time, we should set N to be an intermediate value.

6.3. Explanations of learning performance

In this section, we give some explanations on the learning performance of the proposed two algorithms.

Since the examples that are close to the interface of two classes data are the most "important" examples for classification problems and the size of these examples is smaller compared to the size of total training set, in Algorithm 1 we define the accept probability $p_t^{i+1}(t = 1, 2, \ldots, T)$ by the last classification function g_{t-1}, and then we use them to accept the training examples of Markov resampling such that the "important" examples can be accepted with high probabilities. In other words, in Algorithm 1, these "important" examples for classification problems are drawn, which is the reason that the misclassification rates of the proposed algorithms in this paper are smaller than those of three classical AdaBoost algorithms, XGBoost and SVM-AdaBoost algorithms. It is also the reason that we introduce the idea of resampling for Boosting algorithm and propose the SVM-BM algorithm.

In addition, since the size of examples used to obtain the base classification functions is obviously smaller compared to the size of the given training set, the sampling and training total time of the proposed algorithms are better than those of three classical AdaBoost algorithms and SVM-AdaBoost algorithm (Schapire & Singer, 1999). The sampling and training total time of the proposed algorithms are better than those of XGBoost only for part datsets, the reason is that XGBoost is a Boosting algorithm based on gradient, and a series of improvement methods such as parallelism of feature granularity are used in XGBoost such that it has fast convergence rate. How to improve the proposed algorithms in this paper such that it has less sampling and training total time compared to XGBoost algorithm, at the same time keeping its smaller classification rates is our future investigation.

7. Conclusions

In this paper, we introduced the idea of resampling for Boosting algorithm. We firstly proved that the resampling-based Boosting algorithm with general convex loss function is consistent and established the fast learning rate for resampling-based Boosting algorithm. To our knowledge, these results are the first results on this topic. We also applied the Boosting algorithm based on resampling to the classical classification algorithm, SVM, and proposed the SVM Boosting based on Markov resampling algorithm (SVM-BM). Since the SVM-BM algorithm uses all the examples in the given training set to calculate the weights of these base classifiers, this implies that SVM-BM will be time-consuming as the size of the given training set is larger. Thus we also improved the SVM-BM algorithm and introduced the improved SVM Boosting based on Markov resampling (ISVM-BM). Different from SVM-BM, ISVM-BM uses the support vectors to calculate the weights of these base classifiers. The differences between SVM-BM and ISVM-BM imply that SVM-BM is not only suitable for SVM, but also suitable for other regularization algorithms, such as SVMR (SVM for regression), least squares regularized regression algorithm. However, ISVM-BM is not suitable for the classification algorithms without "the support vector" such as least squares SVM, and the regression algorithms, such as SVMR, least squares regularized regression algorithm.

The experimental studies based on the linear classification function have shown that the proposed two algorithms not only have the smaller misclassification rates, but also have the less sampling and training total time compared to three classical AdaBoost algorithms and SVM-AdaBoost algorithm (Schapire & Singer, 1999). The sampling and training total time of ISVM-BM are less than those of SVM-BM. We also compared the proposed SVM-BM with the widely used and efficient gradient Boosting algorithm, XGBoost (since there is no significant difference between SVM-BM and ISVM-BM in terms of the misclassification rates, we only compared SVM-BM with XGBoost). The experimental results have shown that the misclassification rate of the SVM-BM is smaller compared to XGBoost and the sampling and training total time of SVM-BM are also less or close to that of XGBoost for part datasets. The experiments of the SVM-AdaBoost introduced in Schapire and Singer (1999) imply that SVM classifiers obtained by training all the examples in the given training set are not suitable to be the base learners of AdaBoost algorithm when the size of the given training set is larger since obtaining these the base learners or SVM classifiers by training all the examples in the given training set is very time-consuming or runs out of memory.

Since the algorithmic complexity of classical SVM is high (the algorithmic complexity of SVM with n training examples is about $O(\mathrm{n}^{\tilde{q}})$ (Burges, 1998), where $\tilde{q}$ is a constant satisfying $2 < \tilde{q} < 3$), which implies that the proposed algorithms also have the high algorithmic complexity as the size N of training examples used to train the base classifier is bigger. This is also the limitation of the proposed algorithms. How to deal with this limitation is under our present investigation. In addition, we given some useful discussions on the parameters used in our algorithms. For the parameters n_2, q and N, The discussions above suggest that if sampling and training total time are major concerns, we should set small values for N and n_2, a big value for q. If the misclassification rate is a major concern, we should set big values for N and n_2, a small avlue for q. If we want to have a tread-off between the misclassification rate and the training time, we should set an intermediate value for N, n_2 and q. All the experimental results are based on $n_2 = 5$ and $q = 1.3$.

Along the line of the present work, there are several open problems worth further study. For example, improving the proposed algorithms such that it has less sampling and training total time compared to XGBoost, at the same time keeping its smaller classification rates. Applying the resampling-based Boosting algorithm to other problems such as regression estimation and multi-class. All these problems are under our present investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A

Lemma 1 (*Agarwal & Duchi, 2013*)**.** *Let $g \in \mathcal{H}_K$ be measurable with respect to the σ-field $\mathcal{F}_\mathrm{N} = \sigma(z_1, \ldots, z_\mathrm{N})$. We have that for any fixed $g^* \in \mathcal{H}_K$, and any $\tau \in \mathbb{N}$,*

$$\mathbb{E}\Big[\frac{1}{m}\sum_{t=\mathrm{N}+1}^{\mathrm{N}+m}\mathbb{E}[\phi(g, z_t) - \phi(g^*, z_t)|\mathcal{F}_\mathrm{N}]\Big] \le R_\phi(g) - R_\phi(g^*) + L_\phi G\mathbb{E}[d_{TV}(\mathrm{P}^\tau(\cdot|\mathcal{F}_\mathrm{N}), \pi(\cdot))] + \frac{(\tau - 1)L_\phi G}{m},$$

where G is a positive constant.

Lemma 2 (*Xu, et al., 2015*)**.** *Let $\mathcal{G}$ be a countable class of bounded measurable functions, and $z_1, \ldots, z_n$ be u.e.M.c examples. Assume* *that $0 \le g(z) \le b < +\infty$, for any $g \in \mathcal{G}$ and any $z \in \mathcal{Z}$. Then we have that for any $\varepsilon > 0$,*

$$\mathrm{P}\left\{\left|\frac{1}{n}\sum_{i=1}^{n} g(z_i) - \mathbb{E}(g)\right| \ge \varepsilon\right\} \le 2\exp\left\{\frac{-n\varepsilon^2}{56b\|\Gamma_0\|^2\mathbb{E}(g)}\right\},$$

where $\|\Gamma_0\|$ is a constant, which is defined as that in Lemma 3 of Xu, et al. (2015).

Since $\tilde{f}_T$ involves the capacity of the space $\mathcal{F}^T$, we introduce the concept of covering number.

Definition 2 (Wu et al., 2006)**.** For a subset $\mathcal{G}$ of a metric space and $\varepsilon > 0$, the covering number $\mathcal{N}(\mathcal{G}, \varepsilon)$ is defined to be the minimal integer $q \in \mathbb{N}$ such that there exist q disks with radius ε covering $\mathcal{G}$.

For any $R > 0$, let $B_R = \left\{f \in \mathcal{F}^T : \|f\|_* \le R\right\}$. Then the covering number of B_1 with the metric $\|\cdot\|_{\rho_{\mathcal{X}},2}$ is defined as $\mathcal{N}(B_1, \varepsilon)$ for any $\varepsilon > 0$, where $\rho_{\mathcal{X}}$ is the marginal distribution on $\mathcal{X}$.

Define the convex hull of $\mathcal{F}$ as

$$\mathrm{conv}\mathcal{F} = \left\{\sum \alpha_j f_j \middle| \alpha_t \in \mathbb{R}, \alpha_j \ge 0, \sum \alpha_j = 1, f_j \in \mathcal{F}\right\},$$

and $\overline{\mathrm{conv}}\mathcal{F}$ its closure with respect to $L_2(Q)$-norm, where Q is a probability measure on $\mathcal{X}$, the $L_2(Q)$-norm is defined as

$$\|f\|_{Q,2} = \left(\int_{\mathcal{X}} |f(x)|^2 \mathrm{d}x\right)^{1/2}.$$

An envelope function of a class $\mathcal{F}$ of functions on $\mathcal{X}$ is a function F on $\mathcal{X}$ satisfies $|f(x)| \le F(x)$ for every $x \in \mathcal{X}$ and $f \in \mathcal{F}$. Then we have

Lemma 3 (van der Vaart & Wellner, 1996)**.** *For a VC-class of function $\mathcal{F}$ with measurable envelope function F, we have that for any probability measure Q with $\|F\|_{Q,2} > 0$, inequality*

$$\mathcal{N}(\mathcal{F}, \varepsilon\|F\|_{Q,2}, L_2(Q)) \le K_1 V(\mathcal{F})(16e)^{V(\mathcal{F})}\big(1/\varepsilon\big)^{2(V(\mathcal{F})-1)}$$

holds true for a universal constant K_1 and $0 < \varepsilon < 1$, where $V(\mathcal{F})$ is the VC-index of $\mathcal{F}$.

Lemma 4 (van der Vaart & Wellner, 1996). *Let Q be a probability measure on $\mathcal{X}$ and $\mathcal{F}$ be a class of measurable functions with a measurable square integrable envelope F such that $\int F^2 dQ < \infty$ and*

$$\mathcal{N}\big(\mathcal{F}, \varepsilon\|F\|_{Q,2}, L_2(Q)\big) \le W\big(1/\varepsilon\big)^V, 0 < \varepsilon < 1.$$

Then there exists a constant A which is dependent on W and V only such that

$$\ln\mathcal{N}\big(\overline{\mathrm{conv}}\mathcal{F}, \varepsilon\|F\|_{Q,2}, L_2(Q)\big) \le A\big(1/\varepsilon\big)^{2V/(V+2)}.$$

Appendix B

In this section, we give the proof of Propositions 1, 4–5.

Proof of Proposition 1. Let $\mathcal{F} = \mathcal{H}_K$. According to the assumption of $V(\mathcal{H}_K) = U$ and the condition of $\int H^2 d\rho_{\mathcal{X}} < \infty$, we have $\mathcal{H}_K$ satisfies Lemma 3,

$$\mathcal{N}(\mathcal{H}_K, \varepsilon\|H\|_{\rho_{\mathcal{X}},2}, L_2(\rho_{\mathcal{X}})) \le K_1 U(16e)^U \left(\frac{1}{\varepsilon}\right)^{2(U-1)}.$$

By Lemma 4, we know that for any base classification function $g_t (1 \le t \le T)$, then we have

$$\ln\mathcal{N}(\overline{\mathrm{conv}}\mathcal{H}_K, \varepsilon\|H\|_{\rho_{\mathcal{X}},2}, L_2(\rho_{\mathcal{X}})) \le K_1 U(16e)^U \left(\frac{1}{\varepsilon}\right)^{\frac{2(U+1)}{U}}.$$

By the definition of B_1, we have $B_1 \subset \overline{\mathrm{conv}}\mathcal{H}_K$. It follows that $\ln\mathcal{N}(B_1, \varepsilon) \le \tilde{C}\big(1/\varepsilon\big)^r$, $\forall\ 0 < \varepsilon < 1$, where $\tilde{C} = K_1 U(16e)^U(\|H\|_{\rho_{\mathcal{X}},2})^r$ and $r = 2(U+1)/U$. ■

Proof of Proposition 4. For $g_\lambda \in \mathcal{H}_K$, by the definition of $D(\lambda)$, we have

$$\lambda\|g_\lambda\|_K^2 \le R_\phi(g_\lambda) - R_\phi(f_\rho) + \lambda\|g_\lambda\|_K^2 = D(\lambda).$$

It follows that $\|g_\lambda\|_\infty \le \kappa\|g_\lambda\|_K \le \kappa\sqrt{\mathcal{D}(\lambda)/\lambda}$. By Eq. (2), and notice that $yg_\lambda(x) \le d := \kappa\sqrt{\mathcal{D}(\lambda)/\lambda}$, we have that $0 < g(z) = \phi(yg_\lambda) \le M_{\phi,d}$ and $R_\phi(g_\lambda) \le M_{\phi,d}$. By Lemma 2, we deduce that for any $\varepsilon > 0$,

$$\mathrm{P}\Big\{R_{\phi,n}(g_\lambda) - R_\phi(g_\lambda) \ge \varepsilon\Big\} \le \exp\left\{\frac{-n\varepsilon^2}{56\|\Gamma_0\|^2 M_{\phi,d}^2}\right\}. \tag{10}$$

Take the right-hand side of (10) to be $\delta \in (0, 1)$, we have that with confidence at least $1-\delta$,

$$R_{\phi,n}(g_\lambda) - R_\phi(g_\lambda) \le \left(\frac{56M_{\phi,d}^2\|\Gamma_0\|^2\ln(1/\delta)}{n}\right)^{1/2}. \quad \square$$

Proof of Proposition 5. Let $\mathcal{G} = \big\{\phi(yf) : f \in B_1\big\}$. By Eq. (2), we have $yf(x) \le \|f\|_* = 1$, and $0 < g(z) = \phi(yf) \le M_{\phi,1}$. It follows that $R_\phi(f) \le M_{\phi,1}$. By Lemma 2, we deduce that for any $\varepsilon > 0$,

$$\mathrm{P}\Big\{\sup_{f\in\mathcal{F}^T}\big|R_\phi(f) - R_{\phi,n}(f)\big| \ge \varepsilon\Big\}$$
$$\le 2\mathcal{N}(\mathcal{G}, \varepsilon)\exp\left\{\frac{-n\varepsilon^2}{56M_{\phi,1}^2\|\Gamma_0\|^2}\right\}.$$

By Eq. (3), and the fact that $|\phi(yf_i) - \phi(yf_j)| \le L_\phi\|f_i - f_j\|_{\rho_{\mathcal{X}},2}$, for any $f_i, f_j \in B_1$, we have that

$$\mathrm{P}\Big\{\sup_{f\in\mathcal{F}^T}\big|R_\phi(f) - R_{\phi,n}(f)\big| \ge \varepsilon\Big\}$$
$$\le 2\mathcal{N}\big(B_1, \varepsilon/L_\phi\big)\exp\left\{\frac{-n\varepsilon^2}{56M_{\phi,1}^2\|\Gamma_0\|^2}\right\}. \tag{11}$$

Let the right-hand side of (11) be the same as δ above, we have

$$\varepsilon^{2+r} - a_1\varepsilon^r - a_2 = 0, \tag{12}$$

where,

$$a_1 = \frac{56\ln(2/\delta)M_{\phi,1}^2\|\Gamma_0\|^2}{n},$$
$$a_2 = \frac{56M_{\phi,1}^2\tilde{C}L_\phi^r\|\Gamma_0\|^2}{n}.$$

By Cucker and Smale (2002), we can get the solution of Eq. (12) is $\varepsilon^* := \varepsilon'$ and $\varepsilon' \le \max\big\{\hat{\varepsilon}, \tilde{\varepsilon}\big\}$, where

$$\hat{\varepsilon} = \left(\frac{112M_{\phi,1}^2\|\Gamma_0\|^2\ln(2/\delta)}{n}\right)^{1/2},$$
$$\tilde{\varepsilon} = \left(\frac{112M_{\phi,1}^2\|\Gamma_0\|^2\tilde{C}L_\phi^r}{n}\right)^{1/(2+r)}.$$

Set $n^* = 112M_{\phi,1}^2\|\Gamma_0\|^2(\ln(2/\delta))^{(2+r)/r}\tilde{C}^{-2/r}L_\phi^{-1}$, by Proposition 1, we have that for any $\delta \in (0, 1)$, as $n \ge n^*$, with confidence at least $1-\delta$,

$$\sup_{f\in\mathcal{F}^T}\big|R_\phi(f) - R_{\phi,n}(f)\big| \le \left(\frac{112\|\Gamma_0\|^2\tilde{C}M_{\phi,1}^2 L_\phi^r}{n}\right)^{1/(2+r)}. \quad ■$$

Appendix C

Proof of Theorem 1. By Proposition 3, the excess ϕ-risk $R_\phi(\tilde{f}_T) - R_\phi(f_\rho)$ can be decomposed as

$$R_\phi(\tilde{f}_T) - R_\phi(f_\rho) = \{\mathbb{S}_1 + \mathbb{S}_2 + \mathbb{S}_3\} + D(\lambda),$$

By Propositions 4 and 5, we have

$$\mathbb{S}_1 \leq \Big(\frac{112\|\mathit{\Gamma}_0\|^2\tilde{C}M_{\phi,1}^2L_\phi^r}{n}\Big)^{1/(2+r)},$$

$$\mathbb{S}_3 \leq \Big(\frac{56M_{\phi,d}^2\|\mathit{\Gamma}_0\|^2\ln(1/\delta)}{n}\Big)^{1/2}.$$

We rewrite $\mathbb{S}_2 = R_{\phi,n}(\tilde{f}_T) - R_{\phi,n}(g_\lambda) - \lambda\|g_\lambda\|_K^2$ as

$$\mathbb{S}_2 = \frac{1}{n}\sum_{i=1}^{n}\big[\phi(\sum_{t=1}^{T}\alpha_t' g_t, z_i) - \phi(g_\lambda, z_i)\big] - \lambda\|g_\lambda\|_K^2.$$

By Jensen's inequality, we have

$$\mathbb{S}_2 \leq \sum_{t=1}^{T}\alpha_t'\Big\{\frac{1}{n}\sum_{i=1}^{n}\big[\phi(g_t, z_i) - \phi(g_\lambda, z_i)\big]\Big\} - \lambda\|g_\lambda\|_K^2.$$

We assume that g_w is the function maximize $\frac{1}{n}\sum_{i=1}^{n}\big[\phi(g_t, z_i) - \phi(g_\lambda, z_i)\big]$, $w \in (1, 2, \dots, T)$, then we have

$$\mathbb{S}_2 \leq \frac{1}{n}\sum_{i=1}^{n}\big[\phi(g_w, z_i) - \phi(g_\lambda, z_i)\big] + \lambda\|g_w\|_K^2 - \lambda\|g_\lambda\|_K^2.$$

Let $D = \bigcup_{1\leq t\leq T}D_t$, by the definition of g_w and notice that g_w satisfies $R_{\phi,\mathrm{N}}(g_w) - R_{\phi,\mathrm{N}}(g_\lambda) + \lambda\|g_w\|_K^2 - \lambda\|g_\lambda\|_K^2 \leq 0$ on D_w, thus we have

$$\begin{aligned}\mathbb{S}_2 \leq& \frac{1}{n-\mathrm{N}}\sum_{t\neq w,t=1}^{T}\sum_{z\in D_t}\big[\phi(yg_w(x)) - \phi(yg_\lambda(x))\big]\\ &+ \lambda\|g_w\|_K^2 - \lambda\|g_\lambda\|_K^2.\end{aligned}$$

Take the expectation of the examples on $D_t(t \neq w)$, and reorder the set D with D_w as the starting point, we have

$$\begin{aligned}&\mathbb{E}(\mathbb{S}_2) \leq \mathbb{E}\Big[\frac{1}{n-\mathrm{N}}\sum_{t\neq w,t=1}^{T}\sum_{z\in D_t}\big[\phi(yg_w(x)) - \phi(yg_\lambda(x))\big]\Big]\\ &\ + \lambda\|g_w\|_K^2 - \lambda\|g_\lambda\|_K^2\\ =&\mathbb{E}\Big[\frac{1}{n-\mathrm{N}}\sum_{i=\mathrm{N}+1}^{n}\mathbb{E}\big[\phi(y_ig_w(x_i)) - \phi(y_ig_\lambda(x_i))|D_w\big]\Big]\\ &\ + \lambda\|g_w\|_K^2 - \lambda\|g_\lambda\|_K^2.\end{aligned}$$

By Lemma 1 and Definition 1, we have

$$\begin{aligned}\mathbb{E}(\mathbb{S}_2) \leq& R_\phi(g_w) - R_\phi(g_\lambda) + \lambda\|g_w\|_K^2 - \lambda\|g_\lambda\|_K^2\\ &+ \frac{(\tau-1)L_\phi G}{n-\mathrm{N}} + L_\phi G\gamma_0\varphi^\tau.\end{aligned}$$

By Proposition 2, then we have

$$\begin{aligned}\mathbb{E}(\mathbb{S}_2) \leq& \frac{56\ln(1/\delta)(\kappa\sqrt{D(\lambda)/\lambda} + B)\|\mathit{\Gamma}_0\|^2}{\mathrm{N}} + \frac{(\tau-1)L_\phi G}{n-\mathrm{N}}\\ +&\frac{1}{2}D(\lambda) + L_\phi G\gamma_0\varphi^\tau + \frac{2}{\sqrt{\lambda}}\Big(\frac{112C_\varsigma(\kappa+1)\|\mathit{\Gamma}_0\|^2}{\mathrm{N}}\Big)^{1/1+\varsigma}.\end{aligned}$$

Since $\mathcal{D}(\lambda) \leq C_s\lambda^s$, then we have that for the same δ above and $n \geq 112M_{\phi,1}^2\|\mathit{\Gamma}_0\|^2(\ln(6/\delta))^{(2+r)/r}\tilde{C}^{-2/r}L_\phi^{-1}$, with confidence at least $1-\delta$,

$$\begin{aligned}&\mathbb{E}[R_\phi(\tilde{f}_T) - R_\phi(f_\rho)] \leq \frac{2}{\sqrt{\lambda}}\Big(\frac{112C_\varsigma(\kappa+1)\|\mathit{\Gamma}_0\|^2T}{n}\Big)^{1/1+\varsigma}\\ &+ \frac{(\tau-1)L_\phi GT}{n(T-1)} + \frac{56\ln(3/\delta)(\kappa\sqrt{C_s}\lambda^{(s-1)/2} + B)\|\mathit{\Gamma}_0\|^2T}{n}\\ &+ \Big(\frac{56M_{\phi,d}^2\|\mathit{\Gamma}_0\|^2\ln(3/\delta)}{n}\Big)^{1/2} + \frac{3}{2}C_s\lambda^s + L_\phi G\gamma_0\varphi^\tau\\ &+ \Big(\frac{112\|\mathit{\Gamma}_0\|^2\tilde{C}M_{\phi,1}^2L_\phi^r}{n}\Big)^{1/(2+r)}. \quad \blacksquare\end{aligned}$$

Proof of Theorem 3. If $\phi(g, z)$ is the hinge loss $\ell(g, z)$, then we have $L_\ell = 1$, $M_{\ell,d} \leq 1 + d$ and $M_{\ell,1} \leq 2$. By Theorem 1, we have that for any $0 < \delta < 1$, with confidence at least $1-\delta$,

$$\begin{aligned}&\mathbb{E}[R_\ell(\tilde{f}_T) - R_\ell(f_\rho)] \leq \frac{2}{\sqrt{\lambda}}\Big(\frac{112C_\varsigma(\kappa+1)\|\mathit{\Gamma}_0\|^2T}{n}\Big)^{1/1+\varsigma}\\ &+ \Big(\frac{56(1+\kappa\sqrt{C_s}\lambda^{(s-1)/2})^2\|\mathit{\Gamma}_0\|^2\ln(3/\delta)}{n}\Big)^{1/2} + \frac{3}{2}C_s\lambda^s\\ &+ \frac{56\ln(3/\delta)(\kappa\sqrt{C_s}\lambda^{(s-1)/2} + B)\|\mathit{\Gamma}_0\|^2T}{n} + \frac{(\tau-1)GT}{n(T-1)}\\ &+ \Big(\frac{448\|\mathit{\Gamma}_0\|^2\tilde{C}}{n}\Big)^{1/(2+r)} + G\gamma_0\varphi^\tau.\end{aligned}$$

Set $\lambda = n^{-\beta}$, $\beta \in (0, 1)$ and $\tau = \log_\varphi(n^{-\theta})(\theta > 0)$, then there exist some constants $c_j(j = 1, 2, \dots, 6)$ such that

$$\begin{aligned}\mathbb{E}[R_\ell(\tilde{f}_T) - R_\ell(f_\rho)] \leq& c_1\Big(\frac{1}{n}\Big)^{\frac{1}{1+\varsigma}-\frac{\beta}{2}} + c_2\Big(\frac{1}{n}\Big)^{\frac{1+\beta(s-1)}{2}}\\ &+ c_3\Big(\frac{1}{n}\Big)^{s\beta} + c_4\Big(\frac{\ln n}{n}\Big) + c_5\Big(\frac{1}{n}\Big)^{\theta} + c_6\Big(\frac{1}{n}\Big)^{1/(2+r)}.\end{aligned}$$

For $0 < \beta < \min\{1, 2/(1+\varsigma)\}$, it follows that

$$\mathbb{E}[L(\mathrm{sign}(\tilde{f}_T)) - L(f_c)] \leq \mathit{\Theta}\Big(\frac{\ln n}{n}\Big),$$

where $\mathit{\Theta}$ is a constant with respect to $c_j(j = 1, 2, \dots, 6)$. Because $\mathrm{sign}(f_T) = \mathrm{sign}(\tilde{f}_T)$, by inequality (6) and $\psi(\nu) = |\nu|$, we complete the proof of Theorem 3. ■

References

Agarwal, A., & Duchi, J. C. (2013). The generalzation ability of online algorithms for dependent data. *IEEE Transactions on Information Theory*, *59*(1), 573–587.

Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, *68*(3), 337–404.

Bartlett, P. L., Jordan, M. I., & McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, *101*(473), 138–156.

Bartlett, P. L., & Traskin, M. (2007). AdaBoost is consistent. *Journal of Machine Learning Research (JMLR)*, *8*, 2347–2368.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123–140.

Breiman, L. (1999). Arcing classifiers. *The Annals of Statistics*, *26*(3), 801–824.

Breiman, L. (2000). *Some infinite theory for predictor ensembles*: *Tech. rep. 577*, Berkeley, California: Statistics Department, University of California.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. CRC Press.

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

Chen, T. Q., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 785–794).

Cucker, F., & Smale, S. (2002). Best choices for regularization parameters in learning theory: On the bias–variance problem. *Foundations of Computational Mathematics*, *2*(4), 413–428.

Devroye, L., Györfi, L., & Lugosi, G. (1997). *A probabilistic theory of pattern recognition*. New York: Springer.

Dietterich, T. G. (2000). Ensemble methods in machine learing. In *Proc. international workshop on multiple classifer systems*.

Efron, B., & Tibshirani, R. J. (1993). *An introduction to the bootstrap*. New York, NY: Chapman & Hall.

Evgeniou, T., Pontil, M., & Poggio, T. (2000). Regularization networks and support vector machines. *Advances in Computational Mathematics*, *13*(1), 1–50.

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, *121*, 256–285.

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Machine learning: Proc. 13th international conference* (pp. 148–156). San Francisco: Morgan Kaufman.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of online learning and an application to Boosting. *Journal of Computer and System Sciences*, *55*, 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (1998). Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, *28*(2), 337–407.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *6*(6), 721–741.

Jiang, W. X. (2002). On weak base hypotheses and their implications for Boosting regression and classification. *The Annals of Statistics*, *30*(1), 51–73.

Jiang, W. X. (2004). Process consistency for AdaBoost. *The Annals of Statistics*, *32*(1), 13–29.

Laarhouen, P. M., & Aarts, E. L. (1987). *Simulated annealing: theory and application*. Norwell, MA, USA: Kluwer Acdemic.

Lin, S. B., Lei, Y. W., & Zhou, D. X. (2019). Boosted kernel ridge regression: Optimal learning rates and early stopping. *Journal of Machine Learning Research (JMLR)*, *20*, 1–36.

Lugosi, G., & Vayatis, N. (2004). On the Bayes-risk consistency of regularized Boosting methods. *The Annals of Statistics*, *32*(1), 30–55.

Mukherjee, I., & Schapire, R. E. (2013). A theory of multiclass boosting. *Journal of Machine Learning Research (JMLR)*, *14*, 437–497.

Qian, M. P., & Gong, G. L. (1998). *Applied random processes*. Beijing, China: Peking Univ. Press.

Saberian, M., & Vasconcelos, N. (2019). Multiclass boosting: Margins, codewords, losses, and algorithms. *Journal of Machine Learning Research (JMLR)*, *20*, 1–68.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*, 197–227.

Schapire, R. E., & Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, *37*(3), 297–336.

van der Vaart, A. W., & Wellner, J. A. (1996). *Weak convergence and empirical processes*. New York: Springer-Verlag.

Vapnik, V. (1998). *Statistical learning theory*. New York, NY, USA: Wiley.

Vezhnevets, A., & Vezhnevets, Vladimir (2008). *'Modest AdaBoost'-Teaching AdaBoost to generalize better*. Moscow State University.

Vidyasagar, M. (2003). *Learning and generalization with applications to neural networks* (2nd ed.). London, U.K.: Springer.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*(6), 80–83.

Wu, Q., Ying, Y. M., & Zhou, D. X. (2006). Learning rates of least-square regularized regression. *Foundations of Computational Mathematics*, *6*(2), 171–192.

Xu, J., Tang, Y. Y., Zou, B., Xu, Z. B., Li, L. Q., Lu, Y., & Zhang, B. C. (2015). The generalization ability of SVM classification based on Markov sampling. *IEEE Transactions on Cybernetics*, *45*(6), 1169–1179.

Zhang, T. (2004). Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, *32*(1), 56–134.

Zhang, T., & Yu, B. (2005). Boosting with early stopping: convergence and consistency. *The Annals of Statistics*, *33*(4), 1538–1579.