Fast and Scalable Adversarial Training of Kernel SVM via Doubly Stochastic Gradients

Huimin Wu[1], **Zhengmian Hu**[2], **Bin Gu**[1,3,4*]
[1] School of Computer & Software, Nanjing University of Information Science & Technology, P.R.China
[2]Department of Electrical & Computer Engineering, University of Pittsburgh, PA, USA
[3]JD Finance America Corporation, Mountain View, CA, USA
[4]MBZUAI, United Arab Emirates
wuhuimin@nuist.edu.cn, huzhengmian@gmail.com, jsgubin@gmail.com,

Abstract

Adversarial attacks by generating examples which are almost indistinguishable from natural examples, pose a serious threat to learning models. Defending against adversarial attacks is a critical element for a reliable learning system. Support vector machine (SVM) is a classical yet still important learning algorithm even in the current deep learning era. Although a wide range of researches have been done in recent years to improve the adversarial robustness of learning models, but most of them are limited to deep neural networks (DNNs) and the work for kernel SVM is still vacant. In this paper, we aim at kernel SVM and propose adv-SVM to improve its adversarial robustness via adversarial training, which has been demonstrated to be the most promising defense techniques. To the best of our knowledge, this is the first work that devotes to the fast and scalable adversarial training of kernel SVM. Specifically, we first build connection of perturbations of samples between original and kernel spaces, and then give a reduced and equivalent formulation of adversarial training of kernel SVM based on the connection. Next, doubly stochastic gradients (DSG) based on two unbiased stochastic approximations (i.e., one is on training points and another is on random features) are applied to update the solution of our objective function. Finally, we prove that our algorithm optimized by DSG converges to the optimal solution at the rate of $O(1/t)$ under the constant and diminishing stepsizes. Comprehensive experimental results show that our adversarial training algorithm enjoys robustness against various attacks and meanwhile has the similar efficiency and scalability with classical DSG algorithm.

Introduction

Machine learning models have long been proved to be vulnerable to adversarial attacks which generate subtle perturbations to the inputs that lead to incorrect outputs. The perturbed inputs are defined as adversarial examples where the perturbations that lead to misclassification are often imperceptible. This serious threat has recently led to a large influx of contributions in adversarial attacks especially for deep neural networks (DNNs). These methods of adversarial attacks include FGSM (Goodfellow, Shlens, and Szegedy 2014), PGD (Madry et al. 2017), C&W (Carlini and Wagner 2017), ZOO (Chen et al. 2017) and so on. We give a brief review of them in the section of related work.

The topic of adversarial attacks has also attracted much attention in the field of SVM. In 2004, Dalvi et al. (2004) and later Lowd and Meek (2005a; 2005b) studied the task of spam filtering, showing that linear SVM could be easily tricked by few carefully crafted changes in the content of spam emails, without affecting their readability. Some other attacks such as label flipping attack (Biggio et al. 2011; Xiao, Xiao, and Eckert 2012), poison attack (Biggio, Nelson, and Laskov 2012; Xiao et al. 2015) and evasion attack (Biggio et al. 2013) have also proved the vulnerability of SVM to adversarial examples.

Due to the serious threat of these attacks, there is no doubt that defense techniques that can improve adversarial robustness of learning models are crucial for secure machine learning. Most defensive strategies nowadays focus on DNNs, such as defensive distillation (Papernot et al. 2016), gradient regularization (Ross and Doshivelez 2018) and adversarial training (Madry et al. 2017), among which adversarial training has been demonstrated to be the most effective (Athalye, Carlini, and Wagner 2018). This method focuses on a min-max problem, where the inner maximization is to find the most aggressive adversarial examples and the outer minimization is to find model parameters that minimize the loss on the adversarial examples. Up till now, there have been many forms of adversarial training on DNNs which further improve their robustness and training efficiency compared with standard adversarial training (Shafahi et al. 2019; Carmon et al. 2019; Miyato et al. 2019; Wang et al. 2020).

Since SVM is a classical and important learning model in machine learning, the improvement of its security and robustness is also critical. However, to the best of our knowledge, the only work of adversarial training for SVM is limited to linear SVMs. Specifically, Zhou et al. (2012) formulated a convex adversarial training formula for linear SVMs, in which the constraint is defined over the sample space based on two kinds of attack models. As we know, datasets with complex structures can be hardly classified by linear SVMs, but can be easily handled by kernel SVMs. We give a brief review of adversarial training strategies of DNNs and SVMs in Table 1. From this table, it is easy to find that how

*Corresponding Authors

	Algorithm	Reference	Applicable model	Inner maximization problem	Layers of objective function
DNN	Standard	(Madry et al. 2017)	nonlinear	K-PGD attack	2
	MART	(Wang et al. 2020)	nonlinear	K-PGD attack	2
	VAT	(Miyato et al. 2019)	nonlinear	K-PGD attack	2
	FAT	(Shafahi et al. 2019)	nonlinear	single-step FGSM attack	2
SVM	adv-linear-SVM	(Zhou et al. 2012)	linear	closed-form solution	1
	adv-SVM	Ours	nonlinear	closed-form solution	1

Table 1: Comparisons of different adversarial training algorithms on DNN and SVM.

to improve the robustness of kernel SVMs against adversarial examples is still an unstudied problem.

To fill the vacancy, in this paper, we focus on kernel SVMs and propose adv-SVM to improve their adversarial robustness via adversarial training. To the best of our knowledge, this is the first work that devotes to fast and scalable adversarial training of kernel SVMs. Specifically, we first build connections of perturbations between the original and kernel spaces, i.e., $\phi(x+\delta)$ and $\phi(x)+\delta_\phi$, where δ is the perturbation added to the normal example in the original space, δ_ϕ is the perturbation in the kernel space and $\phi(\cdot)$ is the corresponding feature mapping. Then we construct the closed-form solution of the inner maximization and transform the min-max objective function into a convex minimization problem based on the connection of the perturbations. However, directly optimizing this minimization problem is still difficult since the kernel function, which is necessary for the optimization, needs $O(n^2d)$ operations to be computed, where n is the number of training examples and d is the dimension. Huge requirement of computation complexity hinders its application on large scale datasets.

To further improve its scalability, we apply the doubly stochastic gradient (DSG) algorithm (Dai et al. 2014) to solve our objective. Specifically, in each iteration, we randomly select one training example and one random feature parameter for approximating the value of a kernel function instead of computing it directly. The DSG algorithm effectively reduces the computation complexity of solving kernel methods from $O(n^2d)$ to $O(td)$, where t is the number of iterations (Gu and Huo 2018).

The main contributions of this paper are summarized as follows:

- We develop an adversarial training strategy, adv-SVM, for kernel SVM based on the relationship of perturbations between the original and kernel spaces and transform it into an equivalent convex optimization problem, then apply the DSG algorithm to solve it in an efficient way.
- We provide comprehensive theoretical analysis to prove that our proposed adv-SVM can converge to the optimal solution at the rate of $O(1/t)$ with either a constant stepsize or a diminishing stepsize even though it is based on the approximation principle.
- We investigate the performance of adv-SVM under typical white-box and black-box attacks. The empirical results suggest our proposed algorithm can complete the training process in a relatively short time and stay robust in face of various attack strategies at the same time.

Related Work

During the development of adversarial machine learning, a wide range of attacking methods have been proposed for the crafting of adversarial examples. Here we mention some frequently used ones which are useful for generating adversarial examples in our experiments.

- **Fast Gradient Sign Method (FGSM)**. FGSM, which belongs to white-box attacks, perturbs normal examples for one step by the amount ϵ along the gradient (Goodfellow, Shlens, and Szegedy 2014).
- **Projected Gradient Descent (PGD)**. PGD, which is also a white-box attack method, perturbs normal examples for a number of steps K with a smaller step size and keeps the adversarial examples in the ϵ-ball where ϵ is the maximum allowable perturbation (Madry et al. 2017).
- **C&W**. C&W is also a white-box attack method, which aims to find the minimally-distorted adversarial examples. This method is acknowledged to be one of the strongest attacks up to date (Carlini and Wagner 2017).
- **Zeroth Order Optimization (ZOO)**. ZOO is a black-box attack based on coordinate descent. It assumes that the attackers only have access to the prediction confidence from the victim classifier's outputs. This method is proved to be as effective as C&W attack (Chen et al. 2017).

It should be noted that although these methods are proposed for DNN models, they are also applicable to other learning models. We can apply them to generate adversarial examples for SVM models.

Background

In this section, we give a brief review of adversarial training on linear SVM and the random feature approximation algorithm.

Adversarial Training on Linear SVM

We assume that SVM has been trained on a 2-class dataset $\mathbb{P} = \{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}$ as a normal example in the d-dimensional input space and $y_i \in \{+1, -1\}$ as its label.

The adversarial training process aims to train a robust learning model using adversarial examples. As illustrated in (Zhou et al. 2012), it is formulated as solving a min-max problem. The inner maximization problem simulates the behavior of an attacker which constructs adversarial examples

leading to the maximum output distortion:

$$\max_{x_i'} \quad \left[1 - y_i \left(w^T x_i' + b\right)\right]_+ \qquad (1)$$
$$s.t. \quad \|x_i' - x_i\|_2^2 \leq \epsilon^2$$

where x_i' is the adversarial example of x_i, ϵ denotes the limited range of perturbations, w and b are the parameters of SVM. The loss function here is the commonly used hinge loss and we express it as $[\cdot]_+$.

The outer minimization problem targets to find parameters that minimize the loss caused by inner maximization.

$$\min_{w,b} \quad \frac{1}{2}\|w\|_2^2 + \frac{C}{n}\sum_{i=1}^{n} \max_{x'} \left[1 - y_i \left(w^T x_i' + b\right)\right]_+$$
$$s.t. \quad \|x_i' - x_i\|_2^2 \leq \epsilon^2 \qquad (2)$$

Due to the limited application of linear SVMs, we aim to extend the adversarial training strategy to kernel SVMs.

Random Feature Approximation

Random feature approximation is a powerful technique used in DSG to make kernel methods scalable. This method approximates the kernel function by mapping the decision function to a lower dimensional random feature space. Its theoretic foundation relies on the intriguing duality between kernels and random processes (Geng et al. 2019).

Specifically, the Bochner theorem (Rudin 1962) provides a relationship between the kernel function and a random process ω with measure p: for any stationary and continuous kernel $k(x, x') = k(x - x')$, there exits a probability measure p, such that $k(x, x') = \int_{\mathbb{R}^d} e^{i\omega^T(x-x')} p(\omega) d\omega$. In this way, the value of the kernel function can be approximated by explicitly computing random features $\phi_{\omega_i}(x)$, i.e.,

$$k(x, x') \approx \frac{1}{m}\sum_{i=1}^{m} \phi_{\omega_i}(x)\phi_{\omega_i}^T(x') = \phi_\omega(x)\phi_\omega^T(x') \qquad (3)$$

where m is the number of random features, $\phi_{\omega_i}(x)$ denotes $[\cos(\omega_i^T x), \sin(\omega_i^T x)]^T$ and $\phi_\omega(x)$ denotes $\frac{1}{\sqrt{m}}[\phi_{\omega_1}(x), \phi_{\omega_2}(x), \cdots, \phi_{\omega_m}(x)]$. For the detailed derivation process, please refer to (Ton et al. 2018).

It is clearly that feature mappings are $\mathbb{R}^d \to \mathbb{R}^{2m}$, where $m \ll d$. To further alleviate computational costs, $\phi_{\omega_i}(x)$ can be expressed as $\sqrt{2}\cos(\omega_i^T x + b)$, where ω_i is drawn from $p(\omega)$ and b is drawn uniformly from $[0, 2\pi]$.

It is known that most kernel methods can be expressed as convex optimization problems in reproducing kernel Hilbert space (RKHS) (Dai et al. 2014). A RKHS $\mathcal{H}$ has the reproducing property, i.e., $\forall x \in \mathcal{X}$, $k(x, \cdot) \in \mathcal{H}$ and $\forall f \in \mathcal{H}$, $\langle f(\cdot), k(x, \cdot)\rangle_{\mathcal{H}} = f(x)$. Thus we have $\nabla f(x) = k(x, \cdot)$ and $\nabla \|f\|_{\mathcal{H}}^2 = 2f$ (Dang et al. 2020).

Adversarial Training on Kernel SVM

In this section, we extend the objective function (2) of linear SVM to kernel SVM, where the difficulty lies in the uncontrollable of the perturbations mapped in the kernel space.

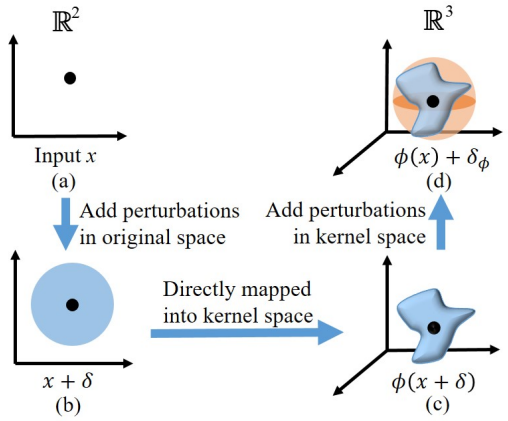


Figure 1: Conceptual illustration of perturbations in the original and kernel spaces.

Kernelization

Firstly, we discuss the kernelization of the perturbations. When constructing an adversarial example, we first add perturbations to the normal example x in the original space constrained by $\|\delta\|_2^2 \leq \epsilon^2$ as shown in Figure 1(a) and 1(b). But once the adversarial example is mapped into the kernel space, it will become unpredictable like Figure 1(c). Then the irregular perturbations greatly increase the difficulty of computation and the obtainment of the closed-form solution.

Fortunately, the following theorem provides a tight connection between perturbations in the original space and the kernel space.

Theorem 1. *(Xu, Caramanis, and Mannor 2009) Supposing that the kernel function has the form $k(x, x') = f(\|x - x'\|)$, with $f : \mathbb{R}^+ \to \mathbb{R}$, a decreasing function, which is denoted by $\mathcal{H}$ the RKHS space of $k(\cdot, \cdot)$ and $\phi(\cdot)$ the corresponding feature mapping, then we have for any $x \in \mathbb{R}^n, w \in \mathcal{H}$ and c>0,*

$$\sup_{\|\delta\| \leq c} \langle w, \phi(x + \delta)\rangle \leq \sup_{\|\delta_\phi\|_{\mathcal{H}} \leq \sqrt{2f(0) - 2f(c)}} \langle w, \phi(x) + \delta_\phi\rangle.$$

Since the perturbation range of $\phi(x) + \delta_\phi$ tightly covers that of $\phi(x+\delta)$, which is also intuitively illustrated in Figure 1(d), then we apply $\phi(x) + \delta_\phi$ to deal with the following computation, making the problem a linear problem in the kernel space. Thus, the inner maximization problem (1) in the kernel space can be expressed as

$$\max_{x'} \quad \left[1 - y_i \left(w^T \Phi(x_i') + b\right)\right]_+ \qquad (4)$$
$$s.t. \quad \|\Phi(x_i') - \phi(x_i)\|_2^2 \leq \epsilon'^2$$

where $\Phi(x_i')$ denotes $\phi(x_i) + \delta_\phi$ and ϵ' is $\sqrt{2f(0) - 2f(\epsilon)}$.

Construction of the Closed-form Solution

In this part, we aim to get the closed-form solution of Eq. (4), then the min-max optimization problem in the kernel space can be transformed into a minimization problem following the strategy of Appendix B in (Lanckriet et al. 2003).

Firstly, we construct the Lagrangian function of Eq. (4):

$$L(\Phi(x_i'), \lambda_i) = 1 - y_i(w^T\Phi(x_i') + b) + \lambda_i\left((\Phi(x_i') - \phi(x_i))^T(\Phi(x_i') - \phi(x_i)) - \epsilon'^2\right) \quad (5)$$

where $\lambda_i \geq 0$. Then let the partial differences of $L(\Phi(x_i'), \lambda_i)$ to $\Phi(x_i')$ and λ_i equal to 0, we can get

$$\Phi(x_i') = \phi(x_i) + \frac{1}{2\lambda_i} y_i w, \quad (6)$$

$$\lambda_i = \frac{\sqrt{w^T w}}{2\epsilon'}. \quad (7)$$

It should be noted that when the hinge loss of (4) equals to 0, its closed-form solution is 0 as well. Then substituting Eq. (6) and (7) into (4), we have

$$\max_{\|\Phi(x_i') - \phi(x_i)\|_2^2 \leq \epsilon'^2} \left[1 - y_i\left(w^T\Phi(x_i') + b\right)\right]_+ = \left[1 - y_i w^T\phi(x_i) - \epsilon'\|w\|_2 - y_i b\right]_+ \quad (8)$$

In this way, the original min-max optimization problem can be rewritten as the following minimization problem:

$$\min_{w\in\mathcal{H}, b} \frac{1}{2}\|w\|_2^2 + \frac{C}{n}\sum_{i=1}^{n}\left[1 - y_i w^T\phi(x_i) - \epsilon'\|w\|_2 - y_i b\right]_+ \quad (9)$$

Learning Strategy of adv-SVM

In this section, we extend the DSG algorithm to solve the objective minimization problem (9), since DSG has been proved to be a powerful technique for scalable kernel learning (Dai et al. 2014).

For easy expression, we substitute $f(\cdot)$ for $w^T\phi(\cdot)$ in Eq. (9), as $\|f\|_2^2 = w^T\phi(\cdot)\phi(\cdot)^T w = \|w\|_2^2$, which is accessible to kernels which satisfy $\phi(\cdot)\phi(\cdot)^T = 1$, such as RBF and Laplacian kernels (Hajiaboli, Ahmad, and Wang 2011), then the objective function can be expressed as follows:

$$\min_{f\in\mathcal{H}} R(f) \quad (10)$$

$$= \min_{f\in\mathcal{H}} \frac{1}{2}\|f\|_2^2 + \frac{C}{n}\sum_{i=1}^{n}\left[1 - y_i f(x_i) - \epsilon'\|f\|_2 - y_i b\right]_+$$

Doubly Stochastic Gradients

In this part, we use the DSG algorithm to update the solution of Eq. (10). For convenience, here we only discuss the case when the hinge loss is greater than 0.

Stochastic Gradient Descent. To iteratively update f in a stochastic manner, we need to sample a data point (x, y) each iteration from the data distribution. The stochastic functional gradient for $R(f)$ is

$$\nabla R(f) = f(\cdot) + C\left[-yk(x,\cdot) - \epsilon'\frac{f(\cdot)}{\|f\|_2}\right] \quad (11)$$

It is noted that $\nabla R(f)$ is the derivative wrt. f. Since it still costs too much if we compute the kernel functions directly, next, we apply the random feature approximation algorithm introduced earlier to approximate the value of the kernels.

Random Feature Approximation. According to Eq. (3), when sampling random process ω from its probability distribution $p(\omega)$, we can further approximate Eq. (11) as

$$\nabla\hat{R}(f) = f(\cdot) + C\left[-y\phi_\omega(x)\phi_\omega(\cdot) - \epsilon'\frac{f(\cdot)}{\|f\|_2}\right] \quad (12)$$

Update Rules. According to the principle of SGD method, the update rule for f in the t-th iteration is

$$f_{t+1}(\cdot) = f_t(\cdot) - \gamma_t\nabla\hat{R}(f) = \sum_{i=1}^{t} a_t^i\zeta_i(\cdot) \quad (13)$$

where γ_t is the stepsize of the t-th iteration, $\zeta_i(\cdot)$ denotes $-Cy_i\phi_{\omega_i}(x_i)\phi_{\omega_i}(\cdot)$ and the initial value $f_1(\cdot) = 0$. The value of a_t^i can be easily inferred as $-\gamma_i\prod_{j=i+1}^{t}\left(1 - \gamma_j\left(1 - \frac{\epsilon' C}{\|f_j\|_2}\right)\right)$[1].

Note that if we compute the value of kernels explicitly instead of using random features, the update rule for f is

$$h_{t+1}(\cdot) = h_t(\cdot) - \gamma_t\nabla R(f) = \sum_{i=1}^{t} a_t^i\xi_i(\cdot). \quad (14)$$

where $\xi_i(\cdot) = -Cy_i k(x_i, \cdot)$. Our algorithm apply the update rule (13) instead, which can reduce the cost of kernel computation.

Detailed Algorithm. Based on Eq. (13) above, we propose the training and prediction algorithms for the adversarial training of kernel SVM in Algorithm 1 and 2 respectively.

Algorithm 1 $\{\alpha_i\}_{i=1}^t = \textbf{Train}(\mathbb{P}(x, y))$

Input: $\mathbb{P}(x, y)$, $p(\omega)$, C.

1: **for** $i = 1, \cdots, t$ **do**
2: Sample $(x_i, y_i) \sim \mathbb{P}(x, y)$.
3: Sample $\omega_i \sim p(\omega)$ with seed i;
4: $f(x_i) = \textbf{Predict}(x_i, \{\alpha_j\}_{j=1}^{i-1})$.
5: Define $\gamma_i = \eta$ (constant) or $\gamma_i = \frac{\theta}{i}$ (diminishing).
6: $\alpha_i = \gamma_i C y_i\phi_{\omega_i}(x_i)$.
7: $\alpha_j = (1 - \gamma_i(1 - \frac{\epsilon' C}{\|f_j\|_2}))\alpha_j$ for $j = 1, \cdots, i-1$
8: **end for**

Algorithm 2 $f(x) = \textbf{Predict}(x, \{\alpha_i\}_{i=1}^t)$

Input: $p(\omega), \phi_\omega(x)$.

1: Set $f(x) = 0$.
2: **for** $i = 1, \cdots, t$ **do**
3: Sample $\omega_i \sim p(\omega)$ with seed i;
4: $f(x) = f(x) + \alpha_i\phi_{\omega_i}(x)$.
5: **end for**

A crucial step of DSG in Algorithm 1 and 2 is sampling ω_i with seed i. As the seeds are aligned for the training and

[1]The value of a_t^i is gotten by expanding the middle term of Eq. (13) iteratively with the definition of $\nabla\hat{R}(f)$.

prediction processes in the same iteration (Shi et al. 2019), we only need to save the seeds instead of the whole random features, which is memory friendly.

Different to the diminishing stepsize used in the original version of DSG (Dai et al. 2014), our algorithm here supports both diminishing and constant stepsize strategies (line 5 of Algorithm 1). The process of gradient descent is composed of a transient phase followed by a stationary phase. In the diminishing stepsize case, the transient phase is relatively long and can be impractical if the stepsize is misspecified (Toulis, Airoldi et al. 2017), but once entering the stationary phase, it will converge to the optimal solution f_* gently. While in the constant stepsize case, the transient phase is much shorter and less sensitive to the stepsize (Chee and Toulis 2018), but it may oscillate in the region of f_* during the stationary phase.

Convergence Analysis

In this section, we aim to prove that adv-SVM can converge to the optimal solution at the rate of $O(1/t)$ based on the framework of (Dai et al. 2014), where t is the number of iterations. We first provide some assumptions.

Assumption 1. *(Bound of kernel function) There exists $\kappa>0$, such that $k(x,x')\leq\kappa$.*

Assumption 2. *(Bound of random feature norm) There exits $\phi>0$, such that $|\phi_\omega(x)\phi_\omega(x')|\leq\phi$.*

Assumption 3. *The spectral radius $\rho(f)$ of a function $f(\cdot)$ has a lower bound that $\rho(f)\geq\epsilon' C\geq 0$, where a spectral radius is the maximum modulus of eigenvalues(Gurvits, Shorten, and Mason 2007), i.e., $\rho(f)=\max_{1\leq i\leq\infty}\left\{\left|\sqrt{\lambda_i}\right|\right\}$.*

For Assumption 3, it is known that we can find n eigenvalues $\{\lambda_i\}_{i=1}^n$ for a matrix A in $\mathbb{R}^n$ space and the spectral radius $\rho(A)$ of matrix A is defined as the maximum modulus of the eigenvalues of A (Gurvits, Shorten, and Mason 2007), i.e., $\rho(A)=\max_{1\leq i\leq n}\{|\lambda_i|\}$. Similar to matrix case, in RKHS space, a function $f(\cdot)$ can be viewed as an infinite matrix, then infinite eigenvalues $\left\{\sqrt{\lambda_i}\right\}_{i=1}^\infty$ and infinite eigenfunctions $\{\psi_i\}_{i=1}^\infty$ can be found (Iii 2004). Treat $\left\{\sqrt{\lambda_i}\psi_i\right\}_{i=1}^\infty$ as a set of orthogonal basis, then $f(\cdot)$ can be represented as the linear combination of the basis, i.e., $f=\sum_{i=1}^\infty f_i\sqrt{\lambda_i}\psi_i$. Similar to the definition of spectral radius in matrix, for function $f(\cdot)$, $\rho(f)=\max_{1\leq i\leq\infty}\left\{\left|\sqrt{\lambda_i}\right|\right\}$.

We update the solution f through random features and random data points according to (13). As a result, f_{t+1} may be outside of RKHS $\mathcal{H}$, making it hard to directly evaluate the error between $f_{t+1}(\cdot)$ and the optimal solution f_*. In this case, we utilize $h_{t+1}(\cdot)$ as an intermediate value to decompose the difference between f_{t+1} and f_* (Shi et al. 2020):

$$\begin{aligned}&|f_{t+1}(x)-f_*|^2 \\ \leq 2&\underbrace{|f_{t+1}(x)-h_{t+1}(x)|^2}_{\text{error due to random features}}+2\kappa\underbrace{\|h_{t+1}-f_*\|_2^2}_{\text{error due to random data}}.\end{aligned}\tag{15}$$

We introduce our main lemmas and theorems as below. All the detailed proofs are provided in our appendix.

Convergence Analysis on Diminishing Stepsize

We first prove that the convergence rate of our algorithm with diminishing stepsize is $O(1/t)$.

Lemma 1. ***(Error due to random features)*** *For any $x\in\mathcal{X}$,*

$$\mathbb{E}_{\mathcal{D}^t,\,\omega^t}\left[|f_{t+1}(x)-h_{t+1}(x)|^2\right]\leq\frac{1}{t^2}C^2\theta^2(k+\phi)^2$$

Lemma 2. ***(Error due to random data)*** *Let f_* be the optimal solution to our target problem, we set $\gamma_t=\frac{\theta}{t}$ with θ such that $\theta>\frac{1}{2}$, then we have*

$$\mathbb{E}_{\mathcal{D}_t,\boldsymbol{\omega}_t}[\|h_{t+1}-f_*\|_2^2]\leq\frac{Q_1^2}{t}\tag{16}$$

where $Q_1=\max\left\{\|f_*\|_2,\frac{\beta_0+\sqrt{\beta_0^2+4(2\theta-1)\beta}}{2(2\theta-1)}\right\}$, $\beta=C^2\theta^2\left[(\kappa+\epsilon')+\kappa^{1/2}\theta\right]^2$, β_0 is a constant value and $\beta_0>0$.

Theorem 2. ***(Convergence in expectation)*** *When $\gamma_t=\frac{\theta}{t}$ with $\theta>\frac{1}{2}$, $\forall x\in\mathcal{X}$,*

$$\mathbb{E}_{\mathcal{D}^t,\omega^t}\left[|f_{t+1}(x)-f_*|^2\right]\leq\frac{2Q_0^2}{t}+\frac{2\kappa Q_1^2}{t}\tag{17}$$

where $Q_0=C\theta(\kappa+\phi)$.

Remark 1. *According to Eq. (15), the error caused by doubly stochastic approximation can be computed via the combination of Lemma 1 and 2 and we prove in Theorem 2 that it converges at the rate of $O(1/t)$.*

Convergence Analysis on Constant Stepsize

In this part, we provide a novel theoretical analysis to prove that adv-SVM with constant stepsize converges to the optimal solution at a rate near $O(1/t)$.

Notice that the diminishing stepsize θ/t provides $1/t$ to the convergence rate, while in the case of constant stepsize, the stepsize η makes the analysis more challenging.

Lemma 3. ***(Error due to random features)*** *For any $x\in\mathcal{X}$,*

$$\mathbb{E}_{\mathcal{D}^t,\omega^t}\left[|f_{t+1}(x)-h_{t+1}(x)|^2\right]\leq C^2\frac{\eta}{c}(\kappa+\phi)^2$$

Lemma 4. ***(Error due to random data)*** *Let f_* be the optimal solution to our target problem, set $t\in[T]$ and $\eta\in(0,1)$, with $\eta=\frac{\epsilon\vartheta}{2B}$ for $\vartheta\in(0,1]$, we will reach $\mathbb{E}_{\mathcal{D}^t,\omega^t,\omega'^t}[\|h_{t+1}-f_*\|_2^2]\leq\epsilon$ after*

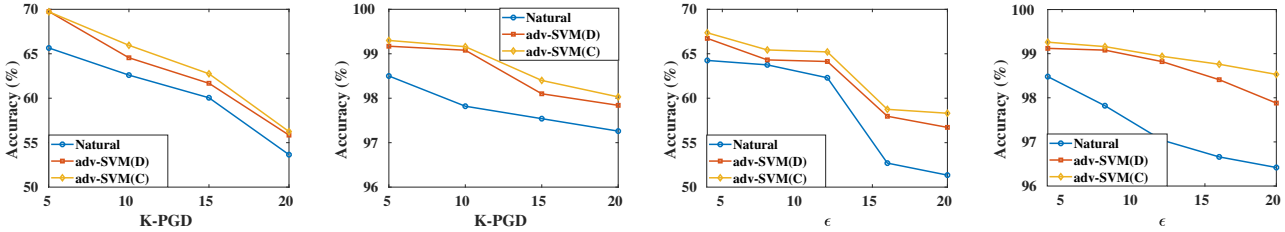
$$T\geq\frac{B\log(2e_1/\epsilon)}{\vartheta\epsilon}\tag{18}$$

iterations, where $B=\frac{1}{2}C\left[(\kappa+\epsilon')+\kappa^{1/2}\frac{1}{c}\right]^2$ and $e_1=\mathbb{E}_{\mathcal{D}_1,\omega_1}[\|h_1-f_*\|_2^2]$.

Theorem 3. ***(Convergence in expectation)*** *Set $t\in[T]$, $T>0$ and $\epsilon>0$, $0<\eta<1$, with $\eta=\frac{\epsilon\vartheta}{8\kappa B}$ where $\vartheta\in(0,1]$, we will reach $\mathbb{E}_{\mathcal{D}_t,\omega_t}\left[|f_{t+1}(x)-f_*|^2\right]\leq\epsilon$ after*

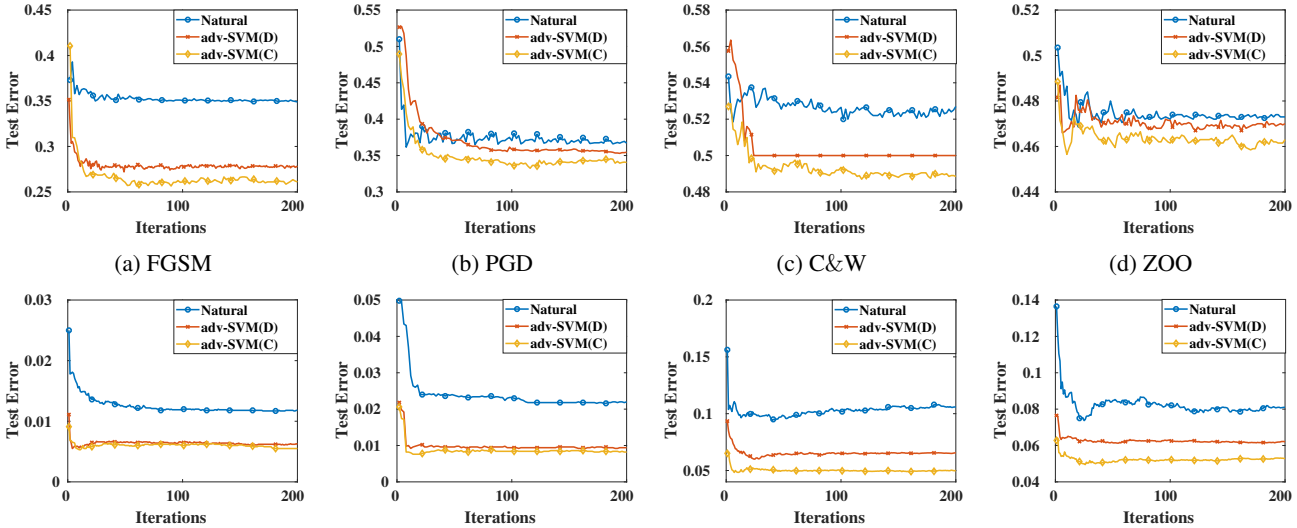
$$T\geq\frac{4\kappa B\log(8\kappa e_1/\epsilon)}{\vartheta\epsilon}\tag{19}$$

iterations, where B and e_1 are defined in Lemma 4.



(a) CIFAR automobile vs. truck (b) MNIST8m 0 vs. 4 (c) CIFAR automobile vs. truck (d) MNIST8m 0 vs. 4

Figure 2: Accuracy of different models when applying different steps PGD attack (Fig. 2a, 2b) and different max perturbation ϵ (Fig. 2c, 2d) to generate adversarial examples.



(a) FGSM (b) PGD (c) C&W (d) ZOO

(e) FGSM (f) PGD (g) C&W (h) ZOO

Figure 3: Test error vs. iterations of different models on four attack methods on CIFAR10 automobile vs. truck (Fig. 3a- 3d) and MNIST8m 0v4 (Fig. 3e-3h). (Since adv-linear-SVM cannot run iteratively like adv-SVM, we do not include it here.)

Remark 2. *Based on Theorem 3, $f_{t+1}(x)$ will converge to the optimal solution f_* at a rate near $O(1/t)$ if eliminating the $\log(1/\epsilon)$ factor. This rate is nearly the same as the one of diminishing stepsize, even though the stepsize of our algorithm keeps constant.*

Experiments

In this section, we will accomplish comprehensive experiments to show the effectiveness and efficiency of adv-SVM.

Experimental Setup

Models compared in experiments include **Natural:** normal DSG algorithm (Dai et al. 2014); **adv-linear-SVM**: adversarial training of linear SVM proposed by Zhou et al. (2012); **adv-SVM(C)**: our proposed adversarial training algorithm with constant stepsize; **adv-SVM(D)**: our proposed adversarial training algorithm with diminishing stepsize.

The four attack methods of constructing adversarial samples we applied cover both white-box and black-box attacks and are already introduced in the section of related work. For FGSM and PGD, the maximum perturbation ϵ is set as $8/255$ and the step size for PGD is $\epsilon/4$. We use the L_2 version of C&W to generate adversarial examples. For ZOO, we use the ZOO-ADAM algorithm and set the step size $\eta = 0.01$, ADAM parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$.

Implementation. We perform experiments on Intel Xeon E5-2696 machine with 48GB RAM. It has been mentioned that our model is implemented[2] based on the DSG framework (Dai et al. 2014). For the sake of efficiency, in the experiment, we use a mini-batch setting. The random features used in DSG are sampled according to pseudo-random number generators. RBF kernel is used for natural DSG and adv-SVM algorithms, the number of random features is set as 2^{10} and the batch size is 500. 5-fold cross validation is used to choose the optimal hyper-parameters (the regularization parameter C and the step size γ). The parameters C and γ are searched in the region $\{(C,\gamma)| -3 \leq \log_2 C \leq 3\ , -3 \leq \log_2 \gamma \leq 3\}$. For algorithm adv-linear-SVM, we

[2]The DSG code is available at https://github.com/zixu1986/Doubly_Stochastic_Gradients.

model	Normal		FGSM		PGD		C&W		ZOO	
	acc	time	acc	time	acc	time	acc	time	acc	time
Natural	**75.65±0.36**	18.03	65.10±0.51	22.50	63.32±0.47	19.63	47.30±0.76	23.52	52.76±0.89	21.23
adv-linear-SVM	73.65±0.58	671.23	70.00±0.52	592.93	60.39±0.75	665.73	48.82±0.93	594.90	50.00±0.24	606.98
adv-SVM(D)	75.00±0.43	19.35	72.70±0.57	22.89	64.55±0.49	20.31	50.00±0.95	17.65	53.03±0.69	19.74
adv-SVM(C)	75.25±0.32	18.25	**73.95±0.53**	18.46	**65.95±0.67**	21.94	**50.55±0.79**	19.73	**53.70±0.81**	18.65

Table 2: Accuracy (%) and running time (min) on CIFAR10 automobile vs. truck against different attacks.

model	Normal		FGSM		PGD		C&W		ZOO	
	acc	time	acc	time	acc	time	acc	time	acc	time
Natural	**99.48±0.08**	50.11	98.82±0.24	47.72	97.82±0.31	58.14	89.42±0.37	49.18	91.90±0.48	39.50
adv-linear-SVM	98.39±0.37	2472.93	97.61±0.17	2624.47	98.81±0.34	2795.34	87.26±0.56	2485.43	88.39±0.75	2602.25
adv-SVM(D)	99.45±0.09	49.98	99.38± 0.15	53.94	99.02±0.47	56.47	93.46±0.36	49.02	93.81±0.64	54.65
adv-SVM(C)	99.46±0.13	57.15	**99.45±0.17**	48.16	**99.16±0.35**	62.52	**95.00±0.57**	47.89	**94.72±0.76**	50.14

Table 3: Accuracy (%) and running time (min) on MNIST8m 0 vs. 4 against different attacks.

use its free-range training model and set the hyper-parameter C_f as 0.1 according to their analysis. This algorithm is implemented in CVX−a package for specifying and solving convex programs (Grant and Boyd 2014). The stop criterion for all experiments is one pass over each entire dataset. All results are the average of 10 trials.

Datasets. We evaluate the robustness of adv-SVM on two well-known datasets, MNIST (Lecun and Bottou 1998) and CIFAR10 (Krizhevsky and Hinton 2009). Since we focus on binary classification of kernel SVM, here we select two similar classes from the datasets respectively. Each pixel value of the data is normalized into $[0, 1]^d$ via dividing its value by 255. Table 4 summarizes the 6 datasets used in our experiments. Due to the page limit, we only show the results of CIFAR10 automobile vs. truck and MNIST8m 0 vs. 4 here. The results of other datasets are provided in the appendix .

Dataset	Features	Sample size
MNIST 1 vs. 7	784	15,170
MNIST 8 vs. 9	784	13,783
CIFAR10 automobile vs. truck	3,072	12,000
CIFAR10 dog vs. horse	3,072	12,000
MNIST 8M 0 vs. 4	784	200,000
MNIST 8M 6 vs. 8	784	200,000

Table 4: Datasets used in the experiments.

Experimental Results

We explore the defensive capability of our model against PGD attack in terms of the attack steps K (Fig. 2a, 2b) and the maximum allowable perturbation ϵ (Fig. 2c, 2d). For Fig. 2a and 2b, the maximum allowable perturbation ϵ is fixed as $8/255$, for Fig. 2c and 2d, the attack step K is fixed as 10.

It can be seen clearly that PGD attack strengthens with the increase of either K or ϵ. Meanwhile, increasing ϵ has greater impact on test accuracy than increasing K. However, due to the large allowable disturbance range, it increases the risks of the detection of adversarial examples at the same time since these perturbed examples are not so much similar as original examples, which explains the reason why our algorithm has a better defensive capability for large ϵ.

We evaluate robustness of the 4 competing methods against 4 types of attacks introduced earlier plus the clean datasets (Normal). Here the attack strategy for PGD is 10 steps with max perturbation $\epsilon = 8/255$. From Table 2 and 3, we can see that on both datasets, the natural model achieves the best accuracy on normal test images, but it's not robust to adversarial examples. Among four attacks, C&W and ZOO have the strongest ability to trick models. Although PGD and FGSM belong to the same type attack method, PGD has stronger attack ability and is more difficult to defend since it's a multi-step iterative attack method rather than a single-step one. According to the results of adv-linear-SVM, we can see that this algorithm is not only time-consuming in training examples, but also vulnerable to strong attacks like C&W and ZOO, which even gets higher test error than unsecured algorithm (natural DSG). In comparison, our proposed adv-SVM can finish tasks in just a few minutes and can defend both white-box and black-box attacks.

Fig. 3 shows test error vs. iterations on three models against four attacks. The results indicate that adv-SVM can converge in a fast speed. Moreover, compared with adv-SVM(D), adv-SVM(C) enjoys a faster convergence rate and lower test error although it may oscillate slightly in the stationary phase, which is consistent with our analysis.

Conclusion

To alleviate SVMs' fragility to adversarial examples, we propose an adversarial training strategy named as adv-SVM which is applicable to kernel SVM. DSG algorithm is also applied to improve its scalability. Although we use the principle of approximation, the theoretical analysis shows that our algorithm can converge to the optimal solution. Moreover, comprehensive experimental results also reveal its efficiency in adversarial training models and robustness against various attacks.

Acknowledgments

B. Gu was partially supported by National Natural Science Foundation of China (No: 62076138), the Qing Lan Project (No.R2020Q04), the National Natural Science Foundation of China (No.62076138), the Six talent peaks project (No.XYDXX-042) and the 333 Project (No. BRA2017455) in Jiangsu Province.

References

Athalye, A.; Carlini, N.; and Wagner, D. 2018. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples. *international conference on machine learning* 274–283.

Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Srndic, N.; Laskov, P.; Giacinto, G.; and Roli, F. 2013. Evasion attacks against machine learning at test time. *European conference on machine learning* 8190: 387–402.

Biggio, B.; Nelson, B.; and Laskov, P. 2012. Poisoning Attacks against Support Vector Machines. *International conference on machine learning* 1467–1474.

Biggio, B.; Nelson, B.; Laskov, P.; nan Hsu, C.; and Lee, W. S. 2011. Support vector machines under adversarial label noise. In *Proc. 3rd Asian Conf. Machine Learning*, 97–112.

Carlini, N.; and Wagner, D. 2017. Towards Evaluating the Robustness of Neural Networks. *ieee symposium on security and privacy* 39–57.

Carmon, Y.; Raghunathan, A.; Schmidt, L.; Duchi, J. C.; and Liang, P. 2019. Unlabeled Data Improves Adversarial Robustness. *neural information processing systems* 11192–11203.

Chee, J.; and Toulis, P. 2018. Convergence diagnostics for stochastic gradient descent with constant learning rate. In *International Conference on Artificial Intelligence and Statistics*, 1476–1485.

Chen, P.; Zhang, H.; Sharma, Y.; Yi, J.; and Hsieh, C. 2017. ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models. *arXiv: Machine Learning* 15–26.

Dai, B.; Xie, B.; He, N.; Liang, Y.; Raj, A.; Balcan, M.-F. F.; and Song, L. 2014. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, 3041–3049.

Dalvi, N.; Domingos, P.; Sumit, M.; and Verma, S. D. 2004. Adversarial classification. *ACM SIGKDD Conference on Knowledge Discovery and Data Mining* 99–108.

Dang, Z.; Li, X.; Gu, B.; Deng, C.; and Huang, H. 2020. Large-Scale Nonlinear AUC Maximization via Triply Stochastic Gradients. *IEEE Transactions on Software Engineering* PP.

Geng, X.; Gu, B.; Li, X.; Shi, W.; Zheng, G.; and Huang, H. 2019. Scalable semi-supervised SVM via triply stochastic gradients. *arXiv preprint arXiv:1907.11584* .

Goodfellow, I.; Shlens, J.; and Szegedy, C. 2014. Explaining and Harnessing Adversarial Examples. *International conference on learning representations* .

Grant, M.; and Boyd, S. 2014. CVX: Matlab Software for Disciplined Convex Programming, version 2.1. http://cvxr.com/cvx.

Gu, B.; and Huo, Z. 2018. Asynchronous doubly stochastic group regularized learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS 2018)*.

Gurvits, L.; Shorten, R.; and Mason, O. 2007. On the Stability of Switched Positive Linear Systems. *IEEE Transactions on Automatic Control* 52(6): 1099–1103.

Hajiaboli, M. R.; Ahmad, M. O.; and Wang, C. 2011. An edge-adapting Laplacian kernel for nonlinear diffusion filters. *IEEE Transactions on Image Processing* 21(4): 1561–1572.

Iii, H. D. 2004. From Zero to Reproducing Kernel Hilbert Spaces in Twelve Pages or Less. http://legacydirs.umiacs.umd.edu/ hal/docs/daume04rkhs.pdf.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto* .

Lanckriet, G. R. G.; Ghaoui, L. E.; Bhattacharyya, C.; and Jordan, M. I. 2003. A Robust Minimax Approach to Classification. *Journal of Machine Learning Research* 3(3): 555–582.

Lecun, Y.; and Bottou, L. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Lowd, D.; and Meek, C. 2005a. Adversarial learning. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, Illinois, USA, August 21-24, 2005*.

Lowd, D.; and Meek, C. 2005b. Good Word Attacks on Statistical Spam Filters. In *CEAS*.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards Deep Learning Models Resistant to Adversarial Attacks. *International conference on learning representations* .

Miyato, T.; Maeda, S.; Koyama, M.; and Ishii, S. 2019. Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(8): 1979–1993.

Papernot, N.; Mcdaniel, P.; Wu, X.; Jha, S.; and Swami, A. 2016. Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks. *ieee symposium on security and privacy* 582–597.

Ross, A. S.; and Doshivelez, F. 2018. Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing their Input Gradients. *national conference on artificial intelligence* 1660–1669.

Rudin, W. 1962. *Fourier analysis on groups*, volume 121967. Wiley Online Library.

Shafahi, A.; Najibi, M.; Ghiasi, M. A.; Xu, Z.; Dickerson, J. P.; Studer, C.; Davis, L. S.; Taylor, G.; and Goldstein, T. 2019. Adversarial training for free 3358–3369.

Shi, W.; Gu, B.; Li, X.; Geng, X.; and Huang, H. 2019. Quadruply stochastic gradients for large scale nonlinear semi-supervised AUC optimization. *arXiv preprint arXiv:1907.12416* .

Shi, W.; Gu, B.; Li, X.; and Huang, H. 2020. Quadruply Stochastic Gradient Method for Large Scale Nonlinear Semi-Supervised Ordinal Regression AUC Optimization. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(4): 5734–5741.

Ton, J.-F.; Flaxman, S.; Sejdinovic, D.; and Bhatt, S. 2018. Spatial mapping with Gaussian processes and nonstationary Fourier features. *Spatial statistics* 28: 59–78.

Toulis, P.; Airoldi, E. M.; et al. 2017. Asymptotic and finite-sample properties of estimators based on stochastic gradients. *The Annals of Statistics* 45(4): 1694–1727.

Wang, Y.; Zou, D.; Yi, J.; Bailey, J.; Ma, X.; and Gu, Q. 2020. Improving Adversarial Robustness Requires Revisiting Misclassified Examples. *international conference on learning representations* .

Xiao, H.; Biggio, B.; Brown, G.; Fumera, G.; Eckert, C.; and Roli, F. 2015. Is Feature Selection Secure against Training Data Poisoning. *international conference on machine learning* 2: 1689–1698.

Xiao, H.; Xiao, H.; and Eckert, C. 2012. Adversarial label flips attack on support vector machines. *european conference on artificial intelligence* 870–875.

Xu, H.; Caramanis, C.; and Mannor, S. 2009. Robustness and Regularization of Support Vector Machines. *Journal of Machine Learning Research* 10: 1485–1510.

Zhou, Y.; Kantarcioglu, M.; Thuraisingham, B.; and Xi, B. 2012. Adversarial Support Vector Machine Learning. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, 1059–1067.