Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Generalization Error Bounds for Optimization Algorithms via Stability

Qi Meng,[1]* Yue Wang,[2]* Wei Chen,[3] Taifeng Wang,[3] Zhi-Ming Ma,[4] Tie-Yan Liu[3]
[1] Peking University, qimeng13@pku.edu.cn; [2]Beijing Jiaotong University, 11271012@bjtu.edu.cn
[3]Microsoft Research, {wche, taifengw, tie-yan.liu}@microsoft.com
[4]Chinese Academy of Mathematics and Systems Science, mazm@amt.ac.cn

Abstract

Many machine learning tasks can be formulated as Regularized Empirical Risk Minimization (R-ERM), and solved by optimization algorithms such as gradient descent (GD), stochastic gradient descent (SGD), and stochastic variance reduction (SVRG). Conventional analysis on these optimization algorithms focuses on their convergence rates during the training process, however, people in the machine learning community may care more about the generalization performance of the learned model on unseen test data. In this paper, we investigate on this issue, by using stability as a tool. In particular, we decompose the generalization error for R-ERM, and derive its upper bound for both convex and nonconvex cases. In convex cases, we prove that the generalization error can be bounded by the convergence rate of the optimization algorithm and the stability of the R-ERM process, both in expectation (in the order of $\mathcal{O}(1/n) + \mathbb{E}\rho(T)$), where $\rho(T)$ is the convergence error and T is the number of iterations) and in high probability (in the order of $\mathcal{O}\left(\frac{\log 1/\delta}{\sqrt{n}} + \rho(T)\right)$ with probability $1-\delta$). For nonconvex cases, we can also obtain a similar expected generalization error bound. Our theorems indicate that 1) along with the training process, the generalization error will decrease for all the optimization algorithms under our investigation; 2) Comparatively speaking, SVRG has better generalization ability than GD and SGD. We have conducted experiments on both convex and nonconvex problems, and the experimental results verify our theoretical findings.

1 Introduction

Many machine learning tasks can be formulated as Regularized Empirical Risk Minimization (R-ERM). Specifically, given a training dataset, the goal of R-ERM is to learn a model from a hypothesis space by minimizing the regularized empirical risk defined as the average loss on the training data plus a regularization term.

In most cases, it is hard to achieve an exact minimization of the objective function since the problem might be too complex to have a closed-form solution. Alternatively, we seek an approximate minimization by using some optimization algorithms. Widely used optimization algorithms include the first-order methods such as gradient descent (GD), stochastic gradient descent (SGD), stochastic variance reduction (SVRG) (Johnson and Zhang 2013), and the second-order methods such as Newton's methods and quasi-Newton's methods (Nocedal and Wright 2006). In this paper, for ease of analysis and without loss of generality, we will take GD, SGD and SVRG as examples. GD calculates the gradient of the objective function at each iteration and updates the model towards the direction of negative gradient by a constant step size. It has been proved that, if the step size is not very large, GD can achieve a linear convergence rate (Nesterov 2013). SGD exploits the additive nature of the objective function in R-ERM, and randomly samples an instance at each iteration to calculate the gradient. Due to the variance introduced by stochastic sampling, SGD has to adopt a decreasing step size in order to guarantee the convergence, and the corresponding convergence rate is sublinear in expectation (Rakhlin, Shamir, and Sridharan 2011). In order to reduce the variance in SGD, SVRG divides the optimization process into multiple stages and updates the model towards a direction of the gradient at a randomly sampled instance regularized by a full gradient over all the instances. In this way, SVRG can achieve linear convergence rate in expectation with a constant step size (Johnson and Zhang 2013).

While the aforementioned convergence analysis can characterize the behaviors of the optimization algorithms in the training process, what the machine learning community cares more is the generalization performance of the learned model on unseen test data. [1] As we know, the generalization error of a machine learning algorithm can be decomposed into three parts, the approximation error, the estimation error, and the optimization error. The approximation error is caused by the limited representation power of the hypothesis space $\mathcal{F}$; the estimation error (which measures the difference between the empirical risk and the expected risk) is caused by the limited amount of training data (Vapnik and Kotz 1982; Bousquet and Elisseeff 2002); and the optimization error (which measures the difference between expected

*This work was done when the author was visiting Microsoft Research Asia.

[1] Under a related but different setting, i.e., the data instances are successively generated from the underlying distribution, people have proven regret bounds for algorithms like SGD (Kakade and Tewari 2009; Cesa-Bianchi, Conconi, and Gentile 2004) and SVRG (Frostig et al. 2015).

risks of the model obtained by the optimization algorithm after T iterations and the true optimum of the regularized empirical risk) is caused by the limited computational power. In (Bousquet and Bottou 2008), Bottou and Bousquet proved generalization error bounds for GD and SGD based on VC-dimension (Kearns and Ron 1999), which unavoidably are very loose in their nature.[2] The goal of our paper is to develop more general and tighter generalization error bounds for the widely used optimization algorithms in R-ERM.

To this end, we leverage stability (Bousquet and Elisseeff 2002) as a tool and obtain the following results:

(1) For convex objective functions, we prove that, the generalization error of an optimization algorithm can be upper bounded by a quantity related to its stability plus its convergence rate in expectation. Specifically, the generalization error bound is in the order of $\mathcal{O}(1/n + \mathbb{E}\rho(T))$, where $\rho(T)$ is the optimization convergence error and T is the number of iterations. This indicates that along with the optimization process on the training data, the generalization error will decrease, which is consistent with our intuition.

(2) For convex objective functions, we can also obtain a high probability bound for the generalization error. In particular, the bound is in the order of $\mathcal{O}\left(\frac{\log 1/\delta}{\sqrt{n}} + \rho(T)\right)$ with probability at least $1-\delta$. That is, if an algorithm has a high-probability convergence bound, we can get a high-probability generalization error bound too, and our bound is sharper than those derived in the previous literature.

(3) Based on our theorems, we analyze the time for different optimization algorithms to achieve the same generalization error, given the same amount of training data. We find that SVRG outperforms GD and SGD in most cases, and although SGD can quickly reduce the test error at the beginning of the training process, it slows down due to the decreasing step size and can hardly obtain the same test error as GD and SVRG when n is large.

(4) Some of our theoretical results can be extended to the nonconvex objective functions, with some additional assumptions on the distance between the global minimizer and the stationary local minimizers.

We have conducted experiments on linear regression, logistic regression, and fully-connected neural networks to verify our theoretical findings. The experimental results are consistent with our theory: (1) when the training process goes on, the test error decreases; (2) in most cases, SVRG has better generalization performance than GD and SGD.

2 Preliminaries

In this section, we briefly introduce the R-ERM problem, and popular optimization algorithms to solve it.

2.1 R-ERM and its Stability

Suppose that we have a training set $S = \{z_1 = (x_1, y_1), ..., z_n = (x_n, y_n)\}$ with n instances that are i.i.d. sampled from $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ according to an unknown distribution $\mathcal{P}$. The goal is to learn a good prediction model $f \in \mathcal{F} : \mathcal{X} \to \mathcal{Y}$, whose prediction accuracy at instance (x, y) is measured by a loss function $l(y, f(x)) = l(f, z)$. Different learning tasks may use different loss functions, such as the least square loss $(f(x) - y)^2$ for regression, and the logistic loss $\log(1 + e^{-yf(x)})$ for classification. We learn the prediction model from the training set S, and will use this model to give predictions for unseen test data.

R-ERM is a very common way to achieve the above goal. Given loss function $l(f, z)$, we aim to learn a model f^* that minimizes the expected risk

$$R(f) = \mathbb{E}_{z\sim\mathcal{P}} l(f, z).$$

Because the underlying distribution $\mathcal{P}$ is unknown, in practice, we learn the prediction model by minimizing the regularized empirical risk over the training instances, which is defined as below,

$$R_S^r(f) = \frac{1}{n}\sum_{i=1}^{n} l(f, z_i) + \lambda N(f). \tag{1}$$

Here, the regularization term $\lambda N(f)$ helps to restrict the capacity of the hypothesis space $\mathcal{F}$ to avoid overfitting. In this paper, we consider $N(f)$ as a norm in a reproducing kernel Hilbert space (RKHS): $N(f) = \|f\|_k^2$ where k refers to the kernel (Wahba 2000).

As aforementioned, our goal is expected risk minimization but what we can do in practice is empirical risk minimization instead. The gap between these two goals is measured by the so-called estimation error, which is usually expressed in the following way: the expected risk is upper bounded by the empirical risk plus a quantity related to the capacity of the hypothesis space (Vapnik and Kotz 1982)(Bousquet and Bottou 2008). One can choose different ways to measure the capacity of the hypothesis space, and stability is one of them, which is proved to be able to produce tighter estimation error bound than VC dimension (Kearns and Ron 1999). There has been a venerable line of research on estimation error analysis based on stability, dated back more than thirty years ago (Bousquet and Elisseeff 2002; Devroye and Wagner 1979; Kearns and Ron 1999; Mukherjee et al. 2006; Shalev-Shwartz et al. 2010). The landmark work by Bousquet and Elisseeff (Bousquet and Elisseeff 2002) introduced the following definitions of uniform loss stability and output stability.

Definition 2.1 *(Uniform Loss Stability) An algorithm A has uniform stability β_0 with respect to loss function l if the following holds $\forall S \in \mathcal{Z}^n, \forall j \in \{1, \cdots, n\}$,*

$$|\mathbb{E}_A[l(A_S, \cdot)] - \mathbb{E}_A[l(A_{S^{\setminus j}}, \cdot)]| \le \beta_0, \tag{2}$$

where $A_S, A_{S^{\setminus j}}$ are the outputs of algorithm A based on S and $S^{\setminus j} = \{z_1, \cdots, z_{j-1}, z_{j+1}, \cdots, z_n\}$, respectively.

Definition 2.2 *(Output Stability) An algorithm has output stability β_1 if the following holds $\forall S \in \mathcal{Z}^n, \forall j \in \{1, \cdots, n\}$,*

$$\|A_S - A_{S^{\setminus j}}\|_{\mathcal{F}_c} \le \beta_1, \tag{3}$$

where $\|\cdot\|_{\mathcal{F}_c}$ denotes the norm in hypothesis space $\mathcal{F}_c$.

[2] In (Hardt, Recht, and Singer 2015), Hardt $et.al$ studied convex risk minimization via stability, but they did not consider the influence of hypothesis space and the tradeoff between approximation error and estimation error.

From the above definitions, we can see that stability measures the change of the loss function or the produced model of a given learning algorithm if one instance in the training set is changed. For example, if the loss function is convex and L-Lipschitz w.r.t. f, the corresponding R-ERM algorithm with regularization term $N(f) = \|f\|_k^2$ has stability $\beta_0 \leq \frac{L^2K^2}{2\lambda n}$ and $\beta_1 \leq \frac{LK}{2\lambda n}$, where K is the upper bound of the kernel norm (Bousquet and Elisseeff 2002).

2.2 Optimization Algorithms

Many optimization methods can be used to solve the R-ERM problem, including the first-order methods such as Gradient Descent (GD) (Nesterov 2013), Stochastic Gradient Descent (SGD) (Rakhlin, Shamir, and Sridharan 2011), and Stochastic Variance Reduction (SVRG) (Johnson and Zhang 2013), as well as the second-order methods such as Newton's methods (Nocedal and Wright 2006) and quasi-Newton's methods (Byrd et al. 2016). We will take the first-order methods as examples in this paper, although many of our analysis can be easily extended to other optimization algorithms.

Let us consider model f parameterized by w. The update rules of GD, SGD, and SVRG are summarized as follows.

Gradient Descent (GD)

$$w_{t+1} = w_t - \eta \nabla R_S^r(w_t). \tag{4}$$

Stochastic Gradient Descent (SGD)

$$w_{t+1} = w_t - \eta_t g(w_t). \tag{5}$$

Stochastic Variance Reduced Gradient (SVRG)

$$v_s^t = g(w_s^t) - \nabla R_S^r(w_s^t) + \nabla R_S^r(\tilde{w}^{t-1}) \tag{6}$$

$$w_{s+1}^t = w_s^t - \eta v_s^t. \tag{7}$$

where $g(\cdot)$ is the gradient of $\nabla R_S^r(\cdot)$ at randomly sampled training instances, w_s^t is the output parameter at the s-th iteration in the t-th stage, and $\tilde{w}^{t-1}$ is the final output in stage $t-1$.

When the loss function is strongly convex and smooth with respect to the model parameters, GD can achieve linear convergence rate; SGD can only achieve sublinear convergence rate due to the variance introduced by stochastic sampling (but in each iteration, it only needs to compute the gradient over one instance and thus can be much faster in speed); SVRG can achieve linear convergence rate by reducing the variance and in most iterations it only needs to compute the gradient over one instance. [3] When the loss functions are nonconvex w.r.t. the model parameters (e.g., neural networks), GD (Nesterov 2013), SGD (Ghadimi and Lan 2013), and SVRG (Reddi et al. 2016) still have convergence properties (although regarding a different measure of convergence). For ease of reference, we summarize the convergence rates of the aforementioned optimization algorithms in both convex and nonconvex cases in Table 1.

[3]The second-order methods can get quadratic convergence rate (Nocedal and Wright 2006). However, as compared with the first-order methods, the computation complexity of the second-order methods could be much higher due to the calculation of the second-order information.

3 Generalization Analysis

In this section, we will analyze the generalization error for optimization algorithms by using stability as a tool. Firstly, we introduce the definition of generalization error and its decomposition. Then, we prove the generalization error bounds of optimization algorithms in both convex and nonconvex cases. The proof details of all the lemmas and theorems are placed in the supplementary materials due to space limitation.

3.1 Generalization Error and its Decomposition

As we mentioned in Section 2, R-ERM minimizes the regularized empirical risk, i.e.,

$$f_{S,r}^* := argmin_{f \in \mathcal{F}} R_S^r(f) \tag{8}$$

as an approximation of the expected risk minimization:

$$f^* := argmin_f R(f). \tag{9}$$

Denote the empirical risk $R_S(f) = \frac{1}{n}\sum_{i=1}^n l(f, z_i)$. It is clear that, the minimization of $R_S^r(f)$ in $\mathcal{F}$ is equivalent to the minimization of $R_S(f)$ in $\mathcal{F}_c = \{f \in \mathcal{F}, N(f) \leq c\}$ for some constant c. That is,

$$f_{S,r}^* = f_{S,\mathcal{F}_c}^* := argmin_{f \in \mathcal{F}_c} R_S(f). \tag{10}$$

Denote the minimizer of the expected risk $R(f)$ in the hypothesis space $\mathcal{F}_c$ as $f_{\mathcal{F}_c}^*$, i.e.,

$$f_{\mathcal{F}_c}^* := argmin_{f \in \mathcal{F}_c} R(f). \tag{11}$$

In many practical cases, neither $f_{S,r}^*$ nor $f_{S,\mathcal{F}_c}^*$ has a closed form. What people do is to implement an iterative optimization algorithm A to produce the prediction model. We denote the output model of algorithm A at iteration T over n training instances as $f_T(A, n, \mathcal{F}_c)$. We use *generalization error* to denote the difference between the expected risk of this learnt model and the optimal expected risk, as follows,

$$\mathcal{E}(A, n, \mathcal{F}_c, T) = R(f_T(A, n, \mathcal{F}_c)) - R(f^*). \tag{12}$$

As known, the generalization error can be decomposed into the three components,

$$\begin{aligned} & \mathcal{E}(A, n, \mathcal{F}_c, T) && (13)\\ = \;& R(f_T) - R(f_{S,\mathcal{F}_c}^*) + R(f_{S,\mathcal{F}_c}^*) - R(f_{\mathcal{F}_c}^*) && (14)\\ & + R(f_{\mathcal{F}_c}^*) - R(f^*) \\ := \;& \mathcal{E}_{opt}(A, n, \mathcal{F}_c, T) + \mathcal{E}_{est}(n, \mathcal{F}_c) + \mathcal{E}_{app}(\mathcal{F}_c). && (15) \end{aligned}$$

The item $\mathcal{E}_{app}(\mathcal{F}_c) := R(f_{\mathcal{F}_c}^*) - R(f^*)$, is called *approximation error*, which is caused by the limited representation power of the hypothesis space $\mathcal{F}_c$. With the hypothesis space increasing, (i.e., c is increasing), the approximation error will decrease. The item $\mathcal{E}_{est}(n, \mathcal{F}_c) := R(f_{S,\mathcal{F}_c}^*) - R(f_{\mathcal{F}_c}^*)$, is called *estimation error*, which is caused by the limited amount of the training data (which leads to the gap between the empirical risk and the expected risk). It will decrease with the increasing training data size n, and the decreasing capacity of the hypothesis space $\mathcal{F}_c$. The item $\mathcal{E}_{opt}(A, n, \mathcal{F}_c, T) := R(f_T) - R(f_{S,\mathcal{F}_c}^*)$, is called *optimization error*, which measures the sub-optimality of the optimization algorithms in terms of the expected risk. It is caused by the limited computational resources. [4]

[4]For simplicity, we sometimes denote $f_T(A, n, \mathcal{F}_c)$, $\mathcal{E}(A, n, \mathcal{F}_c, T)$, $\mathcal{E}_{app}(\mathcal{F}_c)$, $\mathcal{E}_{est}(n, \mathcal{F}_c)$, $\mathcal{E}_{opt}(A, n, \mathcal{F}_c, T)$ as f_T, $\mathcal{E}$, $\mathcal{E}_{app}$, $\mathcal{E}_{est}$, $\mathcal{E}_{opt}$, respectively.

	Convex Number of iterations	Convex Number of data passes	Nonconvex Number of iterations	Nonconvex Number of data passes
GD	$\mathcal{O}(\kappa \log(1/\epsilon))$	$\mathcal{O}(n\kappa \log(1/\epsilon))$	$\mathcal{O}(1/\epsilon)$	$\mathcal{O}(n(1/\epsilon))$
SGD	$\mathcal{O}(\kappa^2/\epsilon)$	$\mathcal{O}\left(\kappa^2/\epsilon\right)$	$\mathcal{O}(1/\epsilon^2)$	$\mathcal{O}\left(1/\epsilon^2\right)$
SVRG	$\mathcal{O}(\kappa \log(1/\epsilon))$	$\mathcal{O}(n + \kappa \log(1/\epsilon))$	$\mathcal{O}(1/\epsilon)$	$\mathcal{O}\left(n + n^{2/3}(1/\epsilon)\right)$

Table 1: Convergence rate of GD, SGD, SVRG in convex and nonconvex cases, where ϵ is the targeted accuracy, κ is the condition number.

Please note that, the optimization error under our study differs from the target in the conventional convergence analysis of optimization algorithms. In the optimization community, the following two objectives

$$\rho_0(T) = R_S(f_T) - R_S(f^*_{S,\mathcal{F}_c});\ \rho_1(T) = \|f_T - f^*_{S,\mathcal{F}_c}\|^2_{\mathcal{F}_c} \quad (16)$$

are commonly used in convex cases, and

$$\rho_2(T) = \|\nabla R^r_S(f_T)\|^2 \quad (17)$$

is commonly used in nonconvex cases. To avoid confusion, we call them *convergence error* and their corresponding upper bounds *convergence error bounds*. Please note although convergence error is different from optimization error, having a convergence error bound plays an important role in guaranteeing a generalization error bound. In the following subsections, we will prove the generalization error bound for typical optimization algorithms, by using the stability techniques, based on their convergence error bounds.

3.2 Expected Generalization Bounds for Convex Case

The following theorem gives an expected generalization error bounds in the convex case.

Theorem 3.1 *Consider an R-ERM problem, if the loss function is L-Lipschitz continuous, γ-smooth, and convex with respect to the prediction output vector, we have*

$$\begin{aligned}\mathbb{E}_{S,A}\mathcal{E} \quad \leq \quad & \mathcal{E}_{app} + 2\beta_0 + \mathbb{E}_{S,A}\rho_0(T) + \frac{\gamma \mathbb{E}_{S,A}\rho_1(T)}{2} \\ & + \sqrt{\mathbb{E}_{S,A}\rho_1(T)\left(\frac{L^2}{2n} + 6L\gamma\beta_1\right)}, \end{aligned} \quad (18)$$

where β_0, β_1 are the uniform stability and output stability of the R-ERM process as defined in 2.1 and 2.2, $\rho_0(T)$ and $\rho_1(T)$ are the convergence errors defined in Eqn 16.

From Theorem 3.1, we can see that the generalization error can be upper bounded by the stability β_0 and β_1, the convergence errors of the optimization algorithms $\rho_0(T)$ and $\rho_1(T)$, and the well-studied approximation error (Vapnik and Vapnik 1998). As the training process goes on, both $\mathbb{E}\rho_0(T)$ and $\mathbb{E}\rho_1(T)$ will decrease. Therefore, the expected generalization error will decrease too. This is consistent with our intuition. Better optimizations will lead to better expected generalization performance.

In order to prove Theorem 3.1, we need the following two lemmas, whose proofs are placed in the supplementary materials due to space restrictions.

Lemma 3.2 *For R-ERM problems, we have $\forall j \in \{1, \cdots, n\}$:*

$$\begin{aligned} & \mathbb{E}_S\left[R(f^*_{S,\mathcal{F}_c}) - R_S(f^*_{S,\mathcal{F}_c})\right] \\ & = \mathbb{E}_S\left[l(f^*_{S,\mathcal{F}_c}, z'_j) - l(f^*_{S^j,\mathcal{F}_c}, z'_j)\right] \quad (19) \\ \textit{and} \quad & \mathbb{E}_S[\nabla R(f^*_{S,\mathcal{F}_c}) - \nabla R_S(f^*_{S,\mathcal{F}_c})] \\ & = \mathbb{E}_S[\nabla_f l(f^*_{S,\mathcal{F}_c}, z'_j) - \nabla_f l(f^*_{S^j,\mathcal{F}_c}, z'_j)], \quad (20) \end{aligned}$$

where $S^j = \{z_1, \cdots, z_{j-1}, z'_j, z_{j+1}, \cdots, z_n\}$, and $f^*_{S^j,\mathcal{F}_c}$ is the minimizer of $R_{S^j}(f)$ in $\mathcal{F}_c$.

Lemma 3.3 *Assume that the loss function is L-Lipschitz and γ-smooth w.r.t. the prediction output vector, we have*

$$\mathbb{E}_S[\nabla R(f^*_{S,\mathcal{F}_c}) - \nabla R_S(f^*_{S,\mathcal{F}_c})]^2 \leq \frac{L^2}{2n} + 6L\gamma\beta_1. \quad (21)$$

Proof Sketch of Theorem 3.1:

Step 1: Since the loss function is convex and γ-smooth w.r.t. f, we can get that $R(f)$ is γ-smooth and $R_S(f)$ is convex w.r.t f. We decompose $\mathcal{E}_{opt}$ as below:

$$\begin{aligned}\mathcal{E}_{opt} \quad \leq \quad & \left(\nabla R(f^*_{S,\mathcal{F}_c}) - \nabla R_S(f^*_{S,\mathcal{F}_c})\right)^T (f_T - f^*_{S,\mathcal{F}_c}) \\ & + R_S(f_T) - R_S(f^*_{S,\mathcal{F}_c}) + \frac{\gamma}{2}\|f_T - f^*_{S,\mathcal{F}_c}\|^2_{\mathcal{F}_c}, \end{aligned}$$

We can use $\rho_0(T)$, $\rho_1(T)$ and Lemma 3.3, to get an upper bound of $\mathbb{E}_{S,A}\mathcal{E}_{opt}$.

Step 2: Since $R_S(f^*_{S,\mathcal{F}_c}) \leq R_S(f^*_{\mathcal{F}_c})$, we have

$$\mathcal{E}_{est} \leq \left[R(f^*_{S,\mathcal{F}_c}) - R_S(f^*_{S,\mathcal{F}_c})\right] + \left[R_S(f^*_{\mathcal{F}_c}) - R(f^*_{\mathcal{F}_c})\right].$$

We have $\mathbb{E}_S\left[R_S(f^*_{\mathcal{F}_c}) - R(f^*_{\mathcal{F}_c})\right] = 0$. By using Lemma 3.2, we can bound $\mathbb{E}_S\mathcal{E}_{est}$. By combining the upper bounds of $\mathbb{E}_{S,A}\mathcal{E}_{opt}$ and $\mathbb{E}_S\mathcal{E}_{opt}$, we can get the results.

After proving the general theorem, we consider a special case - an R-ERM problem with kernel regularization term $\lambda\|f\|^2_k$. In this case, we can derive the concrete expressions of the stability and convergence error. In particular, $\beta_0 = \mathcal{O}(1/\lambda n)$, $\beta_1 = \mathcal{O}(1/\lambda n)$ and $\rho_1(T)$ is equivalent to $\|w_T - w^*_{S,r}\|^2$. If the loss function is convex and smooth w.r.t. parameter w, $R^r_S(w)$ with $N(f) = \|f\|^2_k$ is strongly convex and smooth w.r.t w. In this case, $\rho_0(T)$ dominates $\rho_1(T)$, i.e., $\rho_0(T)$ is larger than $\rho_1(T)$ w.r.t the order of T. Therefore, we can obtain the following corollary.

Corollary 3.4 *For an R-ERM problem with a regularization term $\lambda\|f\|^2_k$, under the same assumptions in Theorem 3.1, and further assuming that the loss function is convex and smooth w.r.t parameter w, we have*

$$\mathbb{E}_{S,A}\mathcal{E} \leq \mathcal{E}_{app} + \mathcal{O}\left(\frac{1}{\lambda n} + \mathbb{E}_{S,A}\rho_0(T)\right). \quad (22)$$

3.3 High-Probability Generalization Bounds for Convex Case

The following theorem gives a high-probability bound of $\mathcal{E}$ in the convex case.

Theorem 3.5 *For an R-ERM problem, if the loss function is L-Lipschitz continuous, γ-smooth and convex with respect to the prediction output vector, and $0 \leq l(f^*_{S,\mathcal{F}_c}, z) \leq M$ for arbitrary $z \in \mathcal{Z}$ and $S \in \mathcal{Z}^n$, then with probability at least $1-\delta$, we have*

$$\begin{aligned}\mathcal{E} \quad &\leq \quad \mathcal{E}_{app} + 2\beta_0 + \rho_0(T) + \frac{\gamma}{2}\rho_1(T) + 2\gamma\beta_1\sqrt{\rho_1(T)} \\ &\quad + \left(4n\beta_0 + 2M + (4n\gamma\beta_1 + L)\sqrt{\rho_1(T)}\right)\sqrt{\frac{\log 4/\delta}{2n}}.\end{aligned}$$

The high-probability bound is consistent with the expected bound given in the previous subsection. That is, the high-probability generalization bound will decrease along with the training process. In addition, we can also get a corollary for the special case of R-ERM with kernel regularization.

Corollary 3.6 *For an R-ERM problem with kernel regularization term $\lambda\|f\|_k^2$, under the same assumptions in Theorem 3.5, and further assuming that the loss function is convex and smooth w.r.t parameter w, we have, with probability at least $1-\delta$,*

$$\mathcal{E} \quad \leq \quad \mathcal{E}_{app} + \mathcal{O}\left(\sqrt{\frac{\log 1/\delta}{n}} + \rho(T)\right).$$

(Rakhlin, Shamir, and Sridharan 2011) proved a high-probability convergence rate for SGD. For GD, the training process is deterministic. By plugging the order of β_0 and β_1 in SGD and GD, we have the following corollary.

Corollary 3.7 *For an R-ERM problem with kernel regularization, under the assumptions in Corollary 3.6, with probability at least $1-\delta$, the generalization error of SGD and GD can be upper bounded as follows,*

$$\mathcal{E}_{SGD} \leq \mathcal{E}_{app} + \mathcal{O}\left(\sqrt{\frac{\log 1/\delta}{n}}\right) + \mathcal{O}\left(\frac{\kappa^2 \log(\frac{\log(T)}{\delta})}{T}\right)$$

$$\mathcal{E}_{GD} \leq \mathcal{E}_{app} + \mathcal{O}\left(\sqrt{\frac{\log 1/\delta}{n}}\right) + \mathcal{O}\left(e^{-\kappa T}\right),$$

where κ is the condition number.

3.4 Expected Generalization Bounds for Nonconvex Case

In this subsection, we consider the case in which the loss function is convex w.r.t. the prediction output vector, but nonconvex w.r.t. the model parameter. This case can cover deep neural networks, which are state-of-the-art AI techniques nowadays. For the nonconvex case, the definition of convergence error is a little different, as shown by Eq. (17). It measures whether the solution is close to a critical point, which is defined and further categorized as follows.

Definition 3.8 *Consider the objective R^r_S and parameter w. If $\nabla R^r_S(w) = 0$, we say w is a critical point of R^r_S; if $\nabla R^r_S(w)$ has at least one strictly negative eigenvalue, we say w is a strict saddle point. If each critical point w is either a local minimum or a strict saddle point, we say that R^r_S satisfies the strict saddle property.*

The following theorem gives the expected generalization error bound for nonconvex cases under the widely used assumptions (Lian et al. 2015; Reddi et al. 2016; Lee et al. 2016; Panageas and Piliouras 2016).

Theorem 3.9 *If R^r_S is μ-strongly convex in the ϵ_0- neighborhood of arbitrary local minimum w_{loc}, satisfies strict saddle point property, L- Lipschitz continuous, γ-smooth and continuously twice differential w.r.t the model parameter w, and the loss function is convex w.r.t f, then we have*

$$\begin{aligned}\mathbb{E}_{S,A}\mathcal{E} \leq \quad &\mathcal{E}_{app} + 2\beta_0 + R(w_{loc}) - R(w^*_{S,\mathcal{F}_c}) \\ &+ \tfrac{L}{\mu}\sqrt{\min_{t=1,\cdots,T}\mathbb{E}_{S,A}\rho_2(t)},\end{aligned}$$

where $T \geq T_1$ and T_1 is the number of iterations to achieve $\min_{t=1,\cdots,T_1}\mathbb{E}_{S,A}\left[\rho_2(t)\right] \leq \gamma^2\epsilon_0^2$.

Similarly to the convex case, Theorem 3.9 shows that with the training process going on, the generalization error in the nonconvex case will also decrease.

4 Sufficient Training and Optimal Generalization Error

In this section, we make further discussions on the generalization bound. In particular, we will explore the sufficient training iterations, and the optimal generalization error given the training data size.

As shown in Section 3, the generalization error bounds consist of an estimation error related to the training data size n and an optimization error related to the training iteration T. Given a machine learning task with fixed training size n, at the early stage of the training process (i.e., T is relatively small), the optimization error will dominate the generalization error; when T becomes larger than a threshold, the optimization error will decrease to be smaller than the estimation error, and then the estimation error will dominate the generalization error. We call this threshold *sufficient training iteration* and the corresponding training time *sufficient training time*. The generalization error with the optimization algorithm sufficiently trained is called *optimal generalization error*. Given the generalization error bound, we can derive the sufficient training iteration/time. For ease of analysis, we list the sufficient training iteration/time of GD, SGD, and SVRG for both convex and nonconvex cases in Table 2.

From Table 2, we have the following observations. For the convex case, when the condition number κ is much smaller than n, GD, SGD and SVRG have no big differences from each other in their sufficient training iterations; when κ is comparable with n, e.g., $\kappa = \mathcal{O}(\sqrt{n})$, [5] the sufficient training time for GD, SGD and SVRG is $\mathcal{O}(n\sqrt{n}d\log n)$, $\mathcal{O}(n^2 d)$, $\mathcal{O}(nd\log n)$, respectively. That is, SVRG corresponds to a shorter sufficient training time than GD and SVRG. For the nonconvex case, if $\epsilon_0 \leq \mathcal{O}(1/n)$, which is more likely to happen for small data size n, the first term in the sufficient training time dominates, and it is fine to terminate the training process at $T = T_1$. SVRG requires shorter training time than

[5] In some cases, κ is related to the regularization coefficient λ and λ is determined by the data size n (Vapnik and Vapnik 1998)(Shamir, Srebro, and Zhang 2014).

	Cost per Iteration	Convex Iterations	Convex Time	Nonconvex Iterations	Nonconvex Time
GD	$\mathcal{O}(nd)$	$\mathcal{O}(\kappa \log n)$	$\mathcal{O}(nd\kappa \log n)$	$\mathcal{O}\left(1/\epsilon_0^2 + n^2\right)$	$\mathcal{O}\left(n/\epsilon_0^2 + n^3\right)$
SGD	$\mathcal{O}(d)$	$\mathcal{O}\left(\kappa^2 n\right)$	$\mathcal{O}\left(nd\kappa^2\right)$	$\mathcal{O}\left(1/\epsilon_0^4 + n^4\right)$	$\mathcal{O}\left(1/\epsilon_0^4 + n^4\right)$
SVRG	$\mathcal{O}(d)$	$\mathcal{O}(\kappa \log n\kappa)$	$\mathcal{O}((nd + d\kappa) \log n\kappa)$	$\mathcal{O}\left(1/\epsilon_0^2 + n^2\right)$	$\mathcal{O}(n^{2/3}/\epsilon_0^2 + n^{8/3})$

Table 2: Sufficient training iteration/time for convex and nonconvex case

GD and SGD by at least an order of $\mathcal{O}(n^{1/3})$ and $\mathcal{O}(n^{4/3})$, respectively. If ϵ_0 is larger than $\mathcal{O}(1/n)$, which is more likely to happen for large data size n, the sufficient training time for GD, SGD, and SVRG is $\mathcal{O}(n^3)$, $\mathcal{O}(n^4)$, and $\mathcal{O}(n^{8/3})$, respectively. In this case, SVRG requires shorter training time than GD and SGD by an order of $\mathcal{O}(n^{1/3})$ and $\mathcal{O}(n^{4/3})$.

5 Experiments

In this section, we report experimental results to validate our theoretical findings. We conducted experiments on three tasks: linear regression, logistic regression, and fully connected neural networks, whose objective functions are least square loss, logistic loss, and cross-entropy loss respectively, plus an L_2 regularization term with $\lambda = 1/\sqrt{n}$. The three tasks are used to verify our results for convex problems, and nonconvex problems. For each task, we report three figures. The horizontal axis of each figure corresponds to the number of data passes and the vertical axis corresponds to the training loss, test loss, and log-scaled test loss, respectively. For linear regression, we independently sample data instances from a Gaussian distribution. We set the step size for GD, SGD, SVRG as 0.032, $0.01/t$ and 0.005, respectively, according to the condition number κ. For our simulated data, the condition number $\kappa \approx 116$. For logistic regression, we conduct binary classification on benchmark dataset *rcv1*. We set the step sizes for GD, SGD, SVRG as 400, $200/t$ and 1, respectively. For neural networks, we work on a model with one fully connected hidden layer of 100 nodes, ten softmax output nodes, and sigmoid activation. We tune the step size for GD, SGD, SVRG and eventually choose 0.03, $0.25/\sqrt{t}$ and 0.001, respectively, which correspond to the best performances in our experiments. The inner loop size for SVRG for convex problems is set as $2n$ and that for nonconvex problem is set as $5n$. The results are shown in Fig.1.

From the results for all the three tasks, we have the following observations. (1) As training error decreases, the test error also decreases. (2) According to Fig.1(c), SVRG is faster than GD by a factor of $\mathcal{O}(\kappa)$ and faster than SGD by a factor of more than $\mathcal{O}(\kappa)$. (3) According to Fig. 1(c)1(f)1(i), SGD is the slowest although it is fast in the beginning, which is consistent with our discussions in Section 4.

By comparing the results of logistic regression and linear regression, we have the following observations. (1) The test error for logistic regression converges after fewer rounds of data passes than linear regression. This is because the condition number κ for logistic regression is smaller than linear regression. (2) SVRG is faster than GD and SGD but the differences between them are less significant for logistic regression, due to a smaller κ. As compared to the results for logistic regression and linear regression, we have the following observations on the results of neural networks. (1) The convergence rate is slower and the accuracy is lower. This is because of the nonconvexity and the gap between global optimum and local optimum. (2) SVRG is faster than GD and SGD but the differences between them are not as significant as in the convex cases, which is consistent with our discussions in Section 4 by considering the data size of CIFAR 10.

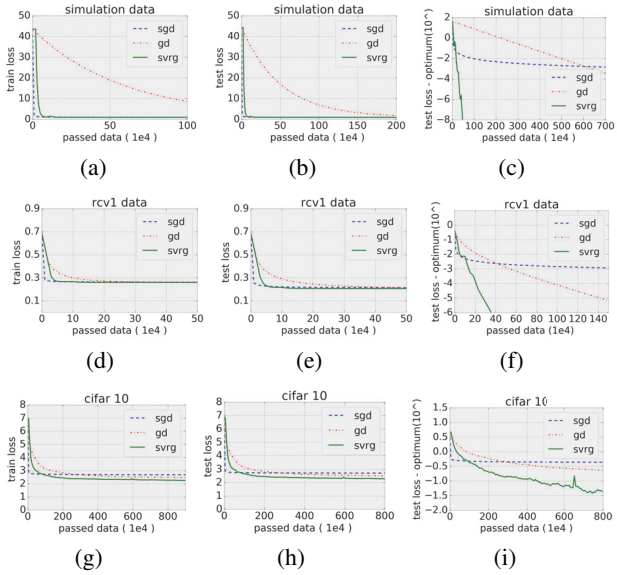

Figure 1: Experimental Results

6 Conclusion

In this paper, we have studied the generalization error bounds for optimization algorithms to solve R-ERM problems, by using stability as a tool. For convex problems, we have obtained both expected bounds and high-probability bounds. Some of our results can be extended to the nonconvex case. Roughly speaking, our theoretical analysis has shown: (1) Along with the training process, the generalization error will decrease; (2) SVRG outperforms GD and SGD in most cases. We have verified the theoretical findings by using experiments on linear regression, logistic regression and fully connected neural networks. In the future, we plan to study the stability of R-ERM with other regularization terms, e.g., the L_1 regularizer.

7 Acknowledgments

Zhi-Ming Ma was partially supported by National Center for Mathematics and Interdisciplinary Sciences (NCMIS) of

China and NSF of China (11526214).

References

Bousquet, O., and Bottou, L. 2008. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, 161–168.

Bousquet, O., and Elisseeff, A. 2002. Stability and generalization. *Journal of Machine Learning Research* 2(Mar):499–526.

Byrd, R. H.; Hansen, S.; Nocedal, J.; and Singer, Y. 2016. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization* 26(2):1008–1031.

Cesa-Bianchi, N.; Conconi, A.; and Gentile, C. 2004. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory* 50(9):2050–2057.

Devroye, L., and Wagner, T. 1979. Distribution-free performance bounds for potential function rules. *IEEE Transactions on Information Theory* 25(5):601–604.

Frostig, R.; Ge, R.; Kakade, S. M.; and Sidford, A. 2015. Competing with the empirical risk minimizer in a single pass. In *Conference on Learning Theory*.

Ghadimi, S., and Lan, G. 2013. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization* 23(4):2341–2368.

Hardt, M.; Recht, B.; and Singer, Y. 2015. Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240*.

Johnson, R., and Zhang, T. 2013. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, 315–323.

Kakade, S. M., and Tewari, A. 2009. On the generalization ability of online strongly convex programming algorithms. In *Advances in Neural Information Processing Systems*, 801–808.

Kearns, M., and Ron, D. 1999. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural Computation* 11(6):1427–1453.

Lee, J. D.; Simchowitz, M.; Jordan, M. I.; and Recht, B. 2016. Gradient descent converges to minimizers. *University of California, Berkeley* 1050:16.

Lian, X.; Huang, Y.; Li, Y.; and Liu, J. 2015. Asynchronous parallel stochastic gradient for nonconvex optimization. In *Advances in Neural Information Processing Systems*, 2737–2745.

Mukherjee, S.; Niyogi, P.; Poggio, T.; and Rifkin, R. 2006. Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics* 25(1-3):161–193.

Nesterov, Y. 2013. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Nocedal, J., and Wright, S. 2006. *Numerical optimization*. Springer Science & Business Media.

Panageas, I., and Piliouras, G. 2016. Gradient descent only converges to minimizers: Non-isolated critical points and invariant regions. *arXiv preprint arXiv:1605.00405*.

Rakhlin, A.; Shamir, O.; and Sridharan, K. 2011. Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647*.

Reddi, S. J.; Hefny, A.; Sra, S.; Póczós, B.; and Smola, A. 2016. Stochastic variance reduction for nonconvex optimization. *arXiv preprint arXiv:1603.06160*.

Shalev-Shwartz, S.; Shamir, O.; Srebro, N.; and Sridharan, K. 2010. Learnability, stability and uniform convergence. *Journal of Machine Learning Research* 11(Oct):2635–2670.

Shamir, O.; Srebro, N.; and Zhang, T. 2014. Communication-efficient distributed optimization using an approximate newton-type method. In *ICML*, volume 32, 1000–1008.

Vapnik, V. N., and Kotz, S. 1982. *Estimation of dependences based on empirical data*, volume 40. Springer-Verlag New York.

Vapnik, V. N., and Vapnik, V. 1998. *Statistical learning theory*, volume 1. Wiley New York.

Wahba, G. 2000. An introduction to model building with reproducing kernel hilbert spaces. *Statistics Department TR* 1020.