TOWARDS UNDERSTANDING LABEL SMOOTHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Label smoothing regularization (LSR) is a prevalent component for training deep neural networks and can improve the generalization of models effectively. Although it achieves empirical success, the theoretical understanding about the power of label smoothing, especially about its influence on optimization, is still limited. In this work, we, for the first time, theoretically analyze the convergence behaviors of stochastic gradient descent with label smoothing in deep learning. Our analysis indicates that an appropriate LSR can speed up the convergence by reducing the variance in gradient, which provides a theoretical interpretation on the effectiveness of LSR. Besides, the analysis implies that LSR may slow down the convergence at the end of optimization. Therefore, a novel algorithm, namely **T**wo-**S**tage **LA**bel smoothing (TSLA), is proposed to further improve the convergence. With the extensive analysis and experiments on benchmark data sets, the effectiveness of TSLA is verified both theoretically and empirically.

1 INTRODUCTION

Due to the massive number of parameters, over-fitting becomes one of the major challenges in training effective deep neural networks (Müller et al., 2019). Various strategies such as weight decay, dropout (Hinton et al., 2012b), batch normalization (Ioffe & Szegedy, 2015), data augmentation (Simard et al., 1998), etc. have been proposed to alleviate the problem. Unlike regularization on input data or model weights, label smoothing regularization (LSR) (Szegedy et al., 2016) perturbs labels of data for better generalization. Concretely, for a K-class classification problem, the one-hot label in the prevalent cross entropy loss is replaced by a soft label as $\mathbf{y}^{\mathrm{LS}} = (1-\theta)\mathbf{y} + \theta\widehat{\mathbf{y}}$, where $\mathbf{y}$ is the one-hot label and $\widehat{\mathbf{y}} = \frac{\mathbf{1}}{K}$ is a uniform distribution for all labels. By optimizing the soft label instead, LSR helps to improve the generalization of deep neural networks in diverse applications including image classification (Zoph et al., 2018; He et al., 2019), speech recognition (Chorowski & Jaitly, 2017; Zeyer et al., 2018), and language translation (Vaswani et al., 2017; Nguyen & Salazar, 2019). Besides, it is complementary to other regularization methods and can work with others to make learned neural networks applicable (He et al., 2019).

Despite the success of LSR, efforts toward theoretical understanding are still limited. Müller et al. (2019) have empirically shown that LSR can help improve model calibration. However, they also found that LSR could impair knowledge distillation when the learned model is applied as the teacher network. Yuan et al. (2019a) have proved that LSR provides a virtual teacher model for knowledge distillation. As a widely used trick, Lukasik et al. (2020) have shown that LSR works since it can successfully mitigate label noise. Most of existing works try to study LSR as regularizing label space while little work investigates the direct impact of LSR on training deep neural networks.

In this work, we analyze the benefits of LSR for the learning process from the view of optimization. Concretely, our theory shows that an appropriate LSR can essentially reduce the variance of stochastic gradient in the assigned class labels and thus it can speed up the convergence of stochastic gradient descent, which is a standard optimizer for deep learning. Besides, the analysis implies that LSR may incur slow convergence at the end of optimization. Inspired by the result, a novel algorithm is developed to accelerate the convergence by a two-stage training paradigm. The effectiveness of the proposed method is verified both theoretically and empirically. We summarize the main contributions of this paper as follows.

- To the best of our knowledge, it is the **first work** that establishes improved iteration complexities of stochastic gradient descent (SGD) (Robbins & Monro, 1951) with LSR for

finding an ϵ-approximate stationary point (Definition 1) in solving a smooth non-convex problem in the presence of an appropriate label smoothing. The results theoretically explain why an appropriate LSR can help speed up the convergence. (Subsection 4.1)

- According to the analysis, we propose a simple yet effective method to accelerate the convergence of LSR (Subsection 4.2). It adopts a two-stage training paradigm that applies LSR in the first stage and then runs on one-hot labels in the second stage.
- The theoretical analysis demonstrates that TSLA has an **improved** iteration complexity (Subsection 4.3) and **better** generalization (Subsection 4.4) compared to the conventional LSR. Extensive experiments on benchmark data sets confirm the effectiveness of our proposed method.

2 Related Work

In this section, we introduce some related work. A closely related idea to LSR is confidence penalty proposed by Pereyra et al. (2017), an output regularizer that penalizes confident output distributions by adding its negative entropy to the negative log-likelihood during the training process. The authors (Pereyra et al., 2017) presented extensive experimental results in training deep neural networks to demonstrate better generalization comparing to baselines with only focusing on the existing hyper-parameters. They have shown that LSR is equivalent to confidence penalty with a reversing direction of KL divergence between uniform distributions and the output distributions.

DisturbLabel introduced by Xie et al. (2016) imposes the regularization within the loss layer, where it randomly replaces some of the ground truth labels as incorrect values at each training iteration. Its effect is quite similar to LSR that can help to prevent the neural network training from overfitting. The authors have verified the effectiveness of DisturbLabel via several experiments on training image classification tasks.

Recently, many works (Zhang et al., 2018; Bagherinezhad et al., 2018; Goibert & Dohmatob, 2019; Shen et al., 2019; Li et al., 2020b) explored the idea of LSR technique. Ding et al. (2019) extended an adaptive label regularization method, which enables the neural network to use both correctness and incorrectness during training. Pang et al. (2018) used the reverse cross-entropy loss to smooth the classifier's gradients. Wang et al. (2020) proposed a graduated label smoothing method that uses the higher smoothing penalty for high-confidence predictions than that for low-confidence predictions. They found that the proposed method can improve both inference calibration and translation performance for neural machine translation models. By contrast, we will try to understand the power of LSR from an optimization perspective and try to study how and when to use LSR.

3 Preliminaries and Notations

We first present some notations. Let $\nabla_{\mathbf{w}} F_{\mathcal{S}}(\mathbf{w})$ denote the gradient of a function $F_{\mathcal{S}}(\mathbf{w})$. When the variable to be taken a gradient is obvious, we use $\nabla F_{\mathcal{S}}(\mathbf{w})$ for simplicity. We use $\|\cdot\|$ to denote the Euclidean norm. Let $\langle\cdot,\cdot\rangle$ be the inner product.

In classification problem, we aim to seek a classifier to map an example $\mathbf{x} \in \mathcal{X}$ onto one of K labels $\mathbf{y} \in \mathcal{Y} \subset \mathbb{R}^K$, where $\mathbf{y} = (y_1, y_2, \ldots, y_K)$ is a one-hot label, meaning that y_i is "1" for the correct class and "0" for the rest. Suppose the example-label pairs are drawn from a distribution $\mathbb{P}$, i.e., $(\mathbf{x}, \mathbf{y}) \sim \mathbb{P} = (\mathbb{P}_{\mathbf{x}}, \mathbb{P}_{\mathbf{y}})$. Let $\mathcal{S} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ denotes a set of n examples drawn from $\mathbb{P}$. We denote by $\mathrm{E}_{(\mathbf{x},\mathbf{y})}[\cdot]$ the expectation that takes over a random variable $(\mathbf{x}, \mathbf{y})$. When the randomness is obvious, we write $\mathrm{E}[\cdot]$ for simplicity. Our goal is to learn a prediction function $f(\mathbf{w}; \mathbf{x}) : \mathcal{W} \times \mathcal{X} \to \mathbb{R}^K$ that is as close as possible to $\mathbf{y}$, where $\mathbf{w} \in \mathcal{W}$ is the parameter and $\mathcal{W}$ is a closed convex set. To this end, we want to minimize the following expected loss under $\mathbb{P}$:

$$\min_{\mathbf{w}\in\mathcal{W}} F_{\mathcal{S}}(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{y}_i, f(\mathbf{w}; \mathbf{x}_i)) \tag{1}$$

where $\ell : \mathcal{Y} \times \mathbb{R}^K \to \mathbb{R}_+$ is a cross-entropy loss function given by

$$\ell(\mathbf{y}, f(\mathbf{w}; \mathbf{x})) = \sum_{i=1}^{K} -y_i \log\left(\frac{\exp(f_i(\mathbf{w}; \mathbf{x}))}{\sum_{j=1}^{K}\exp(f_j(\mathbf{w}; \mathbf{x}))}\right). \tag{2}$$

The objective function $F_{\mathcal{S}}(\mathbf{w})$ is not convex since $f(\mathbf{w};\mathbf{x})$ is non-convex in terms of $\mathbf{w}$. To solve the problem (1), one can simply use some iterative methods such as stochastic gradient descent (SGD). Specifically, at each training iteration t, SGD updates solutions iteratively by $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\nabla_{\mathbf{w}}\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t))$, where $\eta > 0$ is a learning rate.

Next, we present some notations and assumptions that will be used in the convergence analysis. Throughout this paper, we also make the following assumptions for solving the problem (1).

Assumption 1. *Assume the following conditions hold:*

(i) The stochastic gradient of $F_{\mathcal{S}}(\mathbf{w})$ is unbiased, i.e., $\mathrm{E}_{(\mathbf{x},\mathbf{y})}[\nabla\ell(\mathbf{y}, f(\mathbf{w};\mathbf{x}))] = \nabla F_{\mathcal{S}}(\mathbf{w})$, and the variance of stochastic gradient is bounded, i.e., there exists a constant $\sigma^2 > 0$, such that $\mathrm{E}_{(\mathbf{x},\mathbf{y})}\left[\|\nabla\ell(\mathbf{y}, f(\mathbf{w};\mathbf{x})) - \nabla F_{\mathcal{S}}(\mathbf{w})\|^2\right] = \sigma^2$.

(ii) $F_{\mathcal{S}}(\mathbf{w})$ is smooth with an L-Lipchitz continuous gradient, i.e., it is differentiable and there exists a constant $L > 0$ such that $\|\nabla F_{\mathcal{S}}(\mathbf{w}) - \nabla F_{\mathcal{S}}(\mathbf{u})\| \leq L\|\mathbf{w} - \mathbf{u}\|, \forall \mathbf{w}, \mathbf{u} \in \mathcal{W}$.

Remark. Assumption 1 (i) and (ii) are commonly used assumptions in the literature of non-convex optimization (Ghadimi & Lan, 2013; Yan et al., 2018; Yuan et al., 2019b; Wang et al., 2019; Li et al., 2020a). Assumption 1 (ii) says the objective function is L-smooth, and it has an equivalent expression (Nesterov, 2004) which is $F_{\mathcal{S}}(\mathbf{w}) - F_{\mathcal{S}}(\mathbf{u}) \leq \langle\nabla F_{\mathcal{S}}(\mathbf{u}), \mathbf{w}-\mathbf{u}\rangle + \frac{L}{2}\|\mathbf{w}-\mathbf{u}\|^2, \quad \forall \mathbf{w}, \mathbf{u} \in \mathcal{W}$. For a classification problem, the smoothed label $\mathbf{y}^{\mathrm{LS}}$ is given by

$$\mathbf{y}^{\mathrm{LS}} = (1-\theta)\mathbf{y} + \theta\widehat{\mathbf{y}}, \tag{3}$$

where $\theta \in (0,1)$ is the smoothing strength, $\mathbf{y}$ is the one-hot label, $\widehat{\mathbf{y}}$ is an introduced label. For example, one can simply use $\widehat{\mathbf{y}} = \frac{\mathbf{1}}{K}$ (Szegedy et al., 2016) for K-class problems. Similar to label $\mathbf{y}$, we suppose the label $\widehat{\mathbf{y}}$ is drawn from a distribution $\mathbb{P}_{\widehat{\mathbf{y}}}$. We introduce the variance of stochastic gradient using label $\widehat{\mathbf{y}}$ as follows.

$$\mathrm{E}_{(\mathbf{x},\widehat{\mathbf{y}})}\left[\|\nabla\ell(\widehat{\mathbf{y}}, f(\mathbf{w};\mathbf{x})) - \nabla F_{\mathcal{S}}(\mathbf{w})\|^2\right] = \widehat{\sigma}^2 := \delta\sigma^2. \tag{4}$$

where $\delta > 0$ is a constant and σ^2 is defined in Assumption 1 (i). We make several remarks for (4).

Remark. (a) We do not require that the stochastic gradient $\nabla\ell(\widehat{\mathbf{y}}, f(\mathbf{w};\mathbf{x}))$ is unbiased, i.e., it could be $\mathrm{E}[\nabla\ell(\widehat{\mathbf{y}}, f(\mathbf{w};\mathbf{x}))] \neq \nabla F_{\mathcal{S}}(\mathbf{w})$. (b) The variance $\widehat{\sigma}^2$ is defined based on the label $\widehat{\mathbf{y}}$ rather than the smoothed label $\mathbf{y}^{\mathrm{LS}}$. (c) We do not assume the variance $\widehat{\sigma}^2$ is bounded since δ could be an arbitrary value, however, we will discuss the different cases of δ in our analysis. If $\delta \geq 1$, then $\widehat{\sigma}^2 \geq \sigma^2$; while if $0 < \delta < 1$, then $\widehat{\sigma}^2 < \sigma^2$. It is worth mentioning that δ could be small when an appropriate label is used in the label smoothing. For example, one can smooth labels by using a teacher model (Hinton et al., 2014) or the model's own distribution (Reed et al., 2014). In the first paper of label smoothing (Szegedy et al., 2016) and the following related studies (Müller et al., 2019; Yuan et al., 2019a), researchers consider a uniform distribution over all K classes of labels as the label $\widehat{\mathbf{y}}$, i.e., set $\widehat{\mathbf{y}} = \frac{\mathbf{1}}{K}$. Due to the space limitation, we include more discussions and the evaluations of δ in real-world applications in Appendix A. For example, we show that $\delta < 1$ for CIFAR-100 in practice.

We now introduce an important property regarding $F_{\mathcal{S}}(\mathbf{w})$, i.e. Polyak-Łojasiewicz (PL) condition (Polyak, 1963). More specifically, the following assumption holds.

Assumption 2. *There exists a constant $\mu > 0$ such that $2\mu(F_{\mathcal{S}}(\mathbf{w}) - F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})) \leq \|\nabla F_{\mathcal{S}}(\mathbf{w})\|^2, \forall \mathbf{w} \in \mathcal{W}$, where $\mathbf{w}^*_{\mathcal{S}} \in \min_{\mathbf{w}\in\mathcal{W}} F_{\mathcal{S}}(\mathbf{w})$ is a optimal solution.*

Remark. This property has been theoretically and empirically observed in training deep neural networks (Allen-Zhu et al., 2019; Yuan et al., 2019b). This condition is widely used to establish convergence in the literature of non-convex optimization, please see (Yuan et al., 2019b; Wang et al., 2019; Karimi et al., 2016; Li & Li, 2018; Charles & Papailiopoulos, 2018; Li et al., 2020a) and references therein.

To measure the convergence of non-convex and smooth optimization problems as in (Nesterov, 1998; Ghadimi & Lan, 2013; Yan et al., 2018), we need the following definition of the first-order stationary point.

Definition 1 (First-order stationary point)**.** *For the problem of $\min_{\mathbf{w}\in\mathcal{W}} F_{\mathcal{S}}(\mathbf{w})$, a point $\mathbf{w} \in \mathcal{W}$ is called a first-order stationary point if $\|\nabla F_{\mathcal{S}}(\mathbf{w})\| = 0$. Moreover, if $\|\nabla F_{\mathcal{S}}(\mathbf{w})\| \leq \epsilon$, then the point $\mathbf{w}$ is said to be an ϵ-stationary point, where $\epsilon \in (0,1)$ is a small positive value.*

Algorithm 1 SGD with Label Smoothing Regularization

1: **Initialize**: $\mathbf{w}_0 \in \mathcal{W}$, $\theta \in (0,1)$, set η as the value in Theorem 3.
2: **for** $t = 0, 1, \ldots, T-1$ **do**
3: sample $(\mathbf{x}_t, \mathbf{y}_t)$, set $\mathbf{y}_t^{\mathrm{LS}} = (1-\theta)\mathbf{y}_t + \theta\widehat{\mathbf{y}}_t$
4: update $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\nabla_{\mathbf{w}}\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t; \mathbf{x}_t))$
5: **end for**
6: **Output:** w_R, where R is uniformly sampled from $\{0, 1, \ldots, T-1\}$.

4 TSLA: A Two-Stage Label Smoothing Algorithm

In this section, we present our main method with its convergence analysis. As a warm-up, we first show the convergence analysis of SGD with LSR to understand LS from theoretical side. Then we will introduce the proposed TSLA and study its convergence results both in training error (optimization) and testing error (generalization).

4.1 Convergence Analysis of SGD with LSR

To understand LSR from the optimization perspective, we consider SGD with LSR in Algorithm 1 for the sake of simplicity. The only difference between Algorithm 1 and standard SGD is the use of the output label for constructing a stochastic gradient. The following theorem shows that Algorithm 1 converges to an approximate stationary point in expectation under some conditions. We include its proof in Appendix C.

Theorem 3. *Under Assumption 1, run Algorithm 1 with* $\eta = \frac{1}{L}$ *and* $\theta = \frac{1}{1+\delta}$*, then* $\mathrm{E}_R[\|\nabla F_{\mathcal{S}}(\mathbf{w}_R)\|^2] \leq \frac{2F_{\mathcal{S}}(\mathbf{w}_0)}{\eta T} + 2\delta\sigma^2$*, where* R *is uniformly sampled from* $\{0, 1, \ldots, T-1\}$*. Furthermore, given a target accuracy level* ϵ*, we have the following two results.*
(1) when $\delta \leq \frac{\epsilon^2}{4\sigma^2}$*, if we set* $T = \frac{4F_{\mathcal{S}}(\mathbf{w}_0)}{\eta\epsilon^2}$*, then Algorithm 1 converges to an* ϵ*-stationary point in expectation, i.e.,* $\mathrm{E}_R[\|\nabla F_{\mathcal{S}}(\mathbf{w}_R)\|^2] \leq \epsilon^2$*. The total sample complexity is* $T = O\left(\frac{1}{\epsilon^2}\right)$*.*
(2) when $\delta > \frac{\epsilon^2}{4\sigma^2}$*, if we set* $T = \frac{F_{\mathcal{S}}(\mathbf{w}_0)}{\eta\delta\sigma^2}$*, then Algorithm 1 does not converge to an* ϵ*-stationary point, but we have* $\mathrm{E}_R[\|\nabla F_{\mathcal{S}}(\mathbf{w}_R)\|^2] \leq 4\delta\sigma^2 \leq O(\delta)$*.*

Remark. We observe that the variance term is $2\delta\sigma^2$, instead of $\eta L\sigma^2$ for standard analysis of SGD without LSR (i.e., $\theta = 0$, please see the detailed analysis of Theorem 6 in Appendix D). For the convergence analysis, the difference between SGD with LSR and SGD without LSR is that $\nabla\ell(\widehat{\mathbf{y}}, f(\mathbf{w}; \mathbf{x}))$ is not an unbiased estimator of $\nabla F_{\mathcal{S}}(\mathbf{w})$ when using LSR. The convergence behavior of Algorithm 1 heavily depends on the parameter δ. When δ is small enough, say $\delta \leq O(\epsilon^2)$ with a small positive value $\epsilon \in (0,1)$, then Algorithm 1 converges to an ϵ-stationary point with the total sample complexity of $O\left(\frac{1}{\epsilon^2}\right)$. Recall that the total sample complexity of standard SGD without LSR for finding an ϵ-stationary point is $O\left(\frac{1}{\epsilon^4}\right)$ ((Ghadimi & Lan, 2016; Ghadimi et al., 2016), please also see the detailed analysis of Theorem 6 in Appendix D). The convergence result shows that if we could find a label $\widehat{\mathbf{y}}$ that has a reasonably small amount of δ, we will be able to reduce sample complexity for training a machine learning model from $O\left(\frac{1}{\epsilon^4}\right)$ to $O\left(\frac{1}{\epsilon^2}\right)$. Thus, the reduction in variance will happen when an appropriate label smoothing with $\delta \in (0,1)$ is introduced. We will find in the empirical evaluations that different label $\widehat{\mathbf{y}}$ lead to different performances and an appropriate selection of label $\widehat{\mathbf{y}}$ has a better performance (see the performances of LSR and LSR-pre in Table 2). On the other hand, when the parameter δ is large such that $\delta > \Omega(\epsilon^2)$, that is to say, if an inappropriate label smoothing is used, then Algorithm 1 does not converge to an ϵ-stationary point, but it converges to a worse level of $O(\delta)$, indicating that it may slow down the convergence at the end of optimization.

4.2 The TSLA Algorithm

Previous subsection have shown that LSR could not find an $\epsilon-$stationary point under some situations. These motivate us to investigate a strategy that combines the algorithm with and without LSR during the training progress. Let think in this way, one possible scenario is that training one-hot

Algorithm 2 The TSLA algorithm

1: **Initialize**: $\mathbf{w}_0 \in \mathcal{W}$, T_1, $\theta \in (0,1)$, $\eta_1, \eta_2 > 0$
// First stage: SGD with LSR
2: **for** $t = 0, 1, \ldots, T_1 - 1$ **do**
3: set $\mathbf{y}_t^{\text{LS}} = (1-\theta)\mathbf{y}_t + \theta\widehat{\mathbf{y}}_t$
4: update $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_1 \nabla \ell(\mathbf{y}_t^{\text{LS}}, f(\mathbf{w}_t; \mathbf{x}_t))$
5: **end for**
// Second stage: SGD without LSR
6: **for** $t = T_1, 1, \ldots, T_1 + T_2 - 1$ **do**
7: update $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_2 \nabla \ell(\mathbf{y}_t, f(\mathbf{w}_t; \mathbf{x}_t))$.
8: **end for**
9: **Output:** $\mathbf{w}_R$, where R is uniformly sampled from $\{T_1, \ldots, T_1 + T_2 - 1\}$.

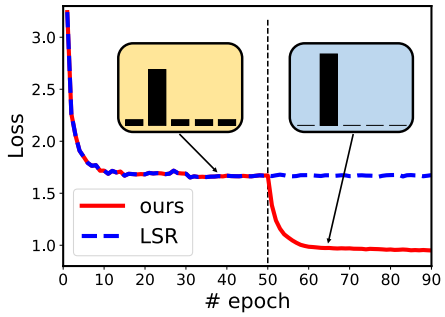

Figure 1: Loss on ResNet-18 over CUB-2011.

label could be "easier" than training smoothed label [1]. Taking the cross entropy loss in (2) for an example, one need to optimize a single loss function $-\log\left(\exp(f_k(\mathbf{w};\mathbf{x}))/\sum_{j=1}^K \exp(f_j(\mathbf{w};\mathbf{x}))\right)$ when one-hot label (e.g, $y_k = 1$ and $y_i = 0$ for all $i \neq k$) is used, but need to optimize all K loss functions $-\sum_{i=1}^K \mathbf{y}_i^{\text{LS}} \log\left(\exp(f_i(\mathbf{w};\mathbf{x}))/\sum_{j=1}^K \exp(f_j(\mathbf{w};\mathbf{x}))\right)$ when smoothed label (e.g., $\mathbf{y}^{\text{LS}} = (1-\theta)\mathbf{y} + \theta\frac{\mathbf{1}}{K}$ so that $y_k^{\text{LS}} = 1 - (K-1)\theta/K$ and $y_i^{\text{LS}} = \theta/K$ for all $i \neq k$) is used. Nevertheless, training deep neural networks is gradually focusing on hard examples with the increase of training epochs. It seems that training with smoothed label in the late epochs makes the learning progress more difficult. In addition, after LSR, we focus on optimizing the overall distribution that contains the minor classes, which are probably not important at the end of training progress. Please see the blue dashed line (trained by SGD with LSR) in Figure 1[2], and it shows that there is almost no change in loss after 30 epochs. One question is whether LSR helps at the early training epochs but it has less (even negative) effect during the later training epochs? This question encourages us to propose and analyze a simple strategy with LSR dropping that switches a stochastic algorithm with LSR to the algorithm without LSR.

A natural choice could be just dropping off LSR after certain epochs. For example, one can run SGD with LSR in the first 50 epochs and then run SGD without LSR after that (see red solid line in Figure 1). From Figure 1, we can see that the drop of LSR after 50 epochs can further decrease the loss function.

With this in mind in this subsection, we propose a generic framework that consists of two stages, wherein the first stage it runs SGD with LSR for T_1 iterations and the second stage it runs SGD without LSR up to T_2 iterations. This framework is referred to as **T**wo-**S**tage **LA**bel smoothing (TSLA) algorithm, whose updating details are presented in Algorithm 2. Although SGD is considered as the pipeline (Line 4 and Line 8 in Algorithm 2) in the convergence analysis, in practice, SGD can be replaced by any stochastic algorithms such as momentum SGD (Polyak, 1964), Stochastic Nesterov's Accelerated Gradient (Nesterov, 1983), and adaptive algorithms including ADAGRAD (Duchi et al., 2011), RMSProp (Hinton et al., 2012a), AdaDelta (Zeiler, 2012), Adam (Kingma & Ba, 2015), Nadam (Dozat, 2016) and AMSGRAD (Reddi et al., 2018). In this paper, we will not study the theoretical guarantees and empirical evaluations of other optimizers, which can be considered as future work. Please note that the algorithm can use different learning rates η_1 and η_2 during the two stages. The last solution of the first stage will be used as the initial solution of the second stage. If $T_1 = 0$, then TSLA reduces to the baseline, i.e., SGD without LSR; while if $T_2 = 0$, TSLA becomes to LSR method, i.e., SGD with LSR.

[1] While this is not a rigorous mathematical conclusion, we want to roughly show that optimizing loss of soft label has to consider minimizing the combination of K functions while one-hot loss only needs to focus on one. For soft label, each f_j appears both in numerator (need to maximize) and denominator (need to minimize) of the loss function, while for one-hot label, $f_j, j \neq k$ only appears in denominator (need to minimize) of the loss function, indicating that one-hot label has less "constraints" than soft label.

[2] The blue dashed line is trained by SGD with LSR. The red solid line is trained by SGD with LSR for the first 50 epochs and trained by SGD without LSR (i.e., using one-hot label) for the last 40 epochs.

4.3 Optimization Result of TSLA

In this subsection, we will give the convergence result of the proposed TSLA algorithm. For simplicity, we use SGD as the subroutine algorithm $\mathcal{A}$ in the analysis. The convergence result in the following theorem shows the power of LSR from the optimization perspective. Its proof is presented in Appendix E. It is easy to see from the proof that by using the last output of the first stage as the initial point of the second stage, TSLA can enjoy the advantage of LSR in the second stage with an improved convergence.

Theorem 4. *Under Assumptions 1, 2, suppose $\sigma^2\delta/\mu \leq F(\mathbf{w}_0)$, run Algorithm 2, $\theta = \frac{1}{1+\delta}$, $\eta_1 = \frac{1}{L}$, $T_1 = \log\left(\frac{2\mu F(\mathbf{w}_0)(1+\delta)}{2\delta\sigma^2}\right)/(\eta_1\mu)$, $\eta_2 = \frac{\epsilon^2}{2L\sigma^2}$ and $T_2 = \frac{8\delta\sigma^2}{\mu\eta_2\epsilon^2}$, then $\mathrm{E}_R[\|\nabla F_{\mathcal{S}}(\mathbf{w}_R)\|^2] \leq \epsilon^2$, where R is uniformly sampled from $\{T_1, \ldots, T_1 + T_2 - 1\}$.*

Remark. It is obvious that the learning rate η_2 in the second stage is roughly smaller than the learning rate η_1 in the first stage, which matches the widely used stage-wise learning rate decay scheme in training neural networks. To explore the total sample complexity of TSLA, we consider different conditions on δ. For a fixed the target convergence level $\epsilon \in (0, 1)$, let us discuss the total sample complexities of finding ϵ-stationary points for SGD with TSLA (TSLA), SGD with LSR (LSR), and SGD without LSR (baseline), where we only consider the orders of the complexities but ignore all constants. When $\Omega(\epsilon^2) < \delta < 1$, LSR does not converge to an ϵ-stationary point, while TSLA reduces sample complexity from $O\left(\frac{1}{\epsilon^4}\right)$ to $O\left(\frac{\delta}{\epsilon^4}\right)$, compared to the baseline. When $\delta < O(\epsilon^2)$, the total complexity of TSLA is between $\log(1/\epsilon)$ and $1/\epsilon^2$, which is always better than LSR and the baseline. In summary, TSLA achieves the best total sample complexity by enjoying the good property of an appropriate label smoothing (i.e., when $0 < \delta < 1$). However, when $\delta \geq 1$, baseline has better convergence than TSLA, meaning that the selection of label $\widehat{\mathbf{y}}$ is not appropriate. Since T_1 contains unknown parameters, it is difficulty to know its ground-truth value. However, we can tune different values in practice.

4.4 Generalization Result of TSLA

In this subsection, we study the generalization result of TSLA. First, we give some notations, where most of them are followed by (Hardt et al., 2015; Yuan et al., 2019b). Let $\mathbf{w}_{\mathcal{S}} = \mathcal{A}(\mathcal{S})$ be a solution that generated by a random algorithm $\mathcal{A}$ based on dataset $\mathcal{S}$. Recall that problem (1) is called empirical risk minimization in the literature, and the true risk minimization is given by

$$\min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}) := \mathrm{E}_{(\mathbf{x},\mathbf{y})}\left[\ell(\mathbf{y}, f(\mathbf{w};\mathbf{x}))\right]. \tag{5}$$

Then the testing error is dedined as

$$\mathrm{E}_{\mathcal{A},\mathcal{S}}[F(\mathbf{w}_{\mathcal{S}})] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})]. \tag{6}$$

We notice that there are several works (Hardt et al., 2015; Yuan et al., 2019b) study the testing error result of SGD for non-convex setting under different conditions such as bounded stochastic gradient $\|\nabla_{\mathbf{w}}\ell(\mathbf{y}, f(\mathbf{w};\mathbf{x}))\| \leq G$ and decaying learning rate $\eta_t \leq \frac{c}{t}$ with a constant $c > 0$, where t is the optimization iteration. In this paper, we are not interested in establishing fast rate under different conditions, but we want to explore the generalization ability of TSLA with the fewest possible modifications when building a bridge between theory and practice. For example, weight decay is a widely used trick when training deep neural networks. With the use of weight decay, the empirical risk minimization in practice becomes $\min_{\mathbf{w}\in\mathcal{W}}\{\widehat{F}_{\mathcal{S}}(\mathbf{w}) := F_{\mathcal{S}}(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|^2\}$. We present the ERB of TSLA in the following Theorem 5, whose proofs can be found in the Appendix F.

Theorem 5. *Under Assumption 1, assume that $\ell(\mathbf{y}, f(\mathbf{w}, \mathbf{x}))$ is L-smooth and B-Lipschitz, suppose $\widehat{F}_{\mathcal{S}}(\mathbf{w})$ satisfies Assumption 2 and $\min_{\mathbf{w}\in\mathcal{W}} \widehat{F}_{\mathcal{S}}(\mathbf{w}) \leq F_{\mathcal{S}}(\mathbf{w}^*_S) + \frac{\lambda}{2}\|\mathbf{w}_t\|^2$ with $\lambda = 2L$, where $\mathbf{w}_t$ is the intermediate solution in the second stage of Algorithm 2 by running with $\theta = \frac{1}{1+\delta}$, $\eta_1 < \frac{1}{3L}$, $T_1 = \log\left(\frac{2\mu F_{\mathcal{S}}(\mathbf{w}_0)(1+\delta)}{2\delta\sigma^2}\right)/(\eta_1\mu)$, $\eta_2 = O(1/\sqrt{n})$ and $T_2 = O(\delta n)$ then the testing error $\mathrm{E}_{R,\mathcal{A},\mathcal{S}}[F(\mathbf{w}_R)] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})] \leq O(1/\sqrt{n})$, where $\mathcal{A}$ is TSLA.*

Remark. Theorem 5 shows that in order to have an bound of testing error in the order of $O(1/\sqrt{n})$, TSLA needs to run $T = O(\delta n)$ iterations. For the standard analysis of SGD with/without LSR (please see the remarks in Appendix F) under the same conditions, SGD without LSR needs to run $T = O(n)$ iterations to obtain an $O(1/\sqrt{n})$ bound of testing error. Thus, if δ is small enough, the

Table 1: Comparison of Testing Accuracy for Different Methods (mean $\pm$ standard deviation, in %).

	Stanford Dogs		CUB-2011	
Algorithm*	Top-1 accuracy	Top-5 accuracy	Top-1 accuracy	Top-5 accuracy
baseline	82.31 $\pm$ 0.18	97.76 $\pm$ 0.06	75.31 $\pm$ 0.25	93.14 $\pm$ 0.31
LSR	82.80 $\pm$ 0.07	97.41 $\pm$ 0.09	76.97 $\pm$ 0.19	92.73 $\pm$ 0.12
TSLA(20)	83.15 $\pm$ 0.02	97.91 $\pm$ 0.08	76.62 $\pm$ 0.15	93.60 $\pm$ 0.18
TSLA(30)	83.89 $\pm$ 0.16	98.05 $\pm$ 0.08	77.44 $\pm$ 0.19	93.92 $\pm$ 0.16
TSLA(40)	**83.93** $\pm$ 0.13	98.03 $\pm$ 0.05	77.50 $\pm$ 0.20	93.99 $\pm$ 0.11
TSLA(50)	83.91 $\pm$ 0.15	**98.07** $\pm$ 0.06	**77.57** $\pm$ 0.21	93.86 $\pm$ 0.14
TSLA(60)	83.51 $\pm$ 0.11	97.99 $\pm$ 0.06	77.25 $\pm$ 0.29	**94.43** $\pm$ 0.18
TSLA(70)	83.38 $\pm$ 0.09	97.90 $\pm$ 0.09	77.21 $\pm$ 0.15	93.31 $\pm$ 0.12
TSLA(80)	83.14 $\pm$ 0.09	97.73 $\pm$ 0.07	77.05 $\pm$ 0.14	93.05 $\pm$ 0.08

*TSLA(s): TSLA drops off LSR after epoch s.

testing error of TSLA is better than that of SGD without LSR, otherwise, SGD without LSR has better testing error than TSLA. For SGD with LSR, when $\delta > \Omega(1/\sqrt{n})$, the bound of testing error for LSR can not be bounded by $O(1/\sqrt{n})$; when $\delta \leq \Omega(1/\sqrt{n})$, the bound for LSR can be bounded by $O(1/\sqrt{n})$ in $T = O(\sqrt{n})$ iterations. This shows that TSLA bas better testing error than LSR.

5 Experiments

To further evaluate the performance of the proposed TSLA method, we trained deep neural networks on three benchmark data sets, CIFAR-100 (Krizhevsky & Hinton, 2009), Stanford Dogs (Khosla et al., 2011) and CUB-2011 (Wah et al., 2011), for image classification tasks. CIFAR-100 [3] has 50,000 training images and 10,000 testing images of 32×32 resolution with 100 classes. Stanford Dogs data set [4] contains 20,580 images of 120 breeds of dogs, where 100 images from each breed is used for training. CUB-2011 [5] is a birds image data set with 11,788 images of 200 birds species. The ResNet-18 model (He et al., 2016) is applied as the backbone in the experiments.

We compare the proposed TSLA incorporating with SGD (TSLA) with two baselines, SGD with LSR (LSR) and SGD without LSR (baseline). The mini-batch size of training instances for all methods is 256 as suggested by (He et al., 2019; 2016). The momentum parameter is fixed as 0.9. We will include more details of experimental settings in Appendix A.

5.1 Stanford Dogs and CUB-2011

We separately train ResNet-18 (He et al., 2016) up to 90 epochs over two data sets Stanford Dogs and CUB-2011. We use weight decay with the parameter value of 10^{-4}. For all algorithms, the initial learning rates for FC are set to be 0.1, while that for the pre-trained backbones are 0.001 and 0.01 for Standford Dogs and CUB-2011, respectively. The learning rates are divided by 10 every 30 epochs. For LSR, we fix the value of smoothing strength $\theta = 0.4$ for the best performance, and the label $\widehat{\mathbf{y}}$ used for label smoothing is set to be a uniform distribution over all K classes, i.e., $\widehat{\mathbf{y}} = \frac{\mathbf{1}}{K}$. The same values of the smoothing strength θ and the same $\widehat{\mathbf{y}}$ are used during the first stage of TSLA. For TSLA, we drop off the LSR (i.e., let $\theta = 0$) after s epochs during the training process, where $s \in \{20, 30, 40, 50, 60, 70, 80\}$.

We first report the highest top-1 and top-5 accuracy on the testing data sets for different methods. All top-1 and top-5 accuracy are averaged over 5 independent random trails with their standard deviations. The results of the comparison are summarized in Table 1, where the notation "TSLA(s)" means that the TSLA algorithm drops off LSR after epoch s. It can be seen from Table 1 that under an appropriate hyperparameter setting the models trained using TSLA outperform that trained using LSR and baseline, which supports the convergence result in Section 4.2. We notice that the best top-1 accuracy of TSLA are TSLA(40) and TSLA(50) for Stanford Dogs and CUB-2011,

[3] https://www.cs.toronto.edu/~kriz/cifar.html
[4] http://vision.stanford.edu/aditya86/ImageNetDogs/
[5] http://www.vision.caltech.edu/visipedia/CUB-200.html

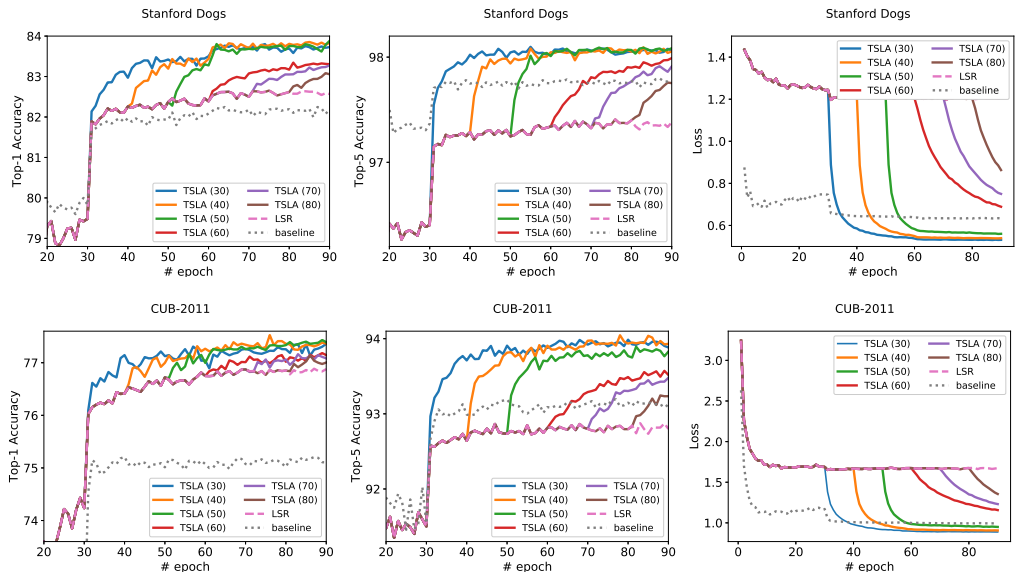

Figure 2: Testing Top-1, Top-5 Accuracy and Loss on ResNet-18 over Stanford Dogs and CUB-2011. TSLA(s) means TSLA drops off LSR after epoch s.

respectively, meaning that the performance of TSLA(s) is not monotonic over the dropping epoch s. For CUB-2011, the top-1 accuracy of TSLA(20) is smaller than that of LSR. This result matches the convergence analysis of TSLA showing that it can not drop off LSR too early. For top-5 accuracy, we found that TSLA(80) is slightly worse than baseline. This is because of dropping LSR too late so that the update iterations (i.e., T_2) in the second stage of TSLA is too small to converge to a good solution. We also observe that LSR is better than baseline regarding top-1 accuracy but the result is opposite as to top-5 accuracy.

We then plot the averaged top-1 accuracy, averaged top-5 accuracy, and averaged loss among 5 trails of different methods in Figure 2. We remove the results for TSLA(20) since it dropped off LSR too early as mentioned before. The figure shows TSLA improves the top-1 and top-5 testing accuracy immediately once it drops off LSR. Although TSLA may not converge if it drops off LSR too late, see TSLA(60), TSLA(70), and TSLA(80) from the third column of Figure 2, it still has the best performance compared to LSR and baseline. TSLA(30), TSLA(40), and TSLA(50) can converge to lower objective levels, comparing to LSR and baseline.

5.2 CIFAR-100

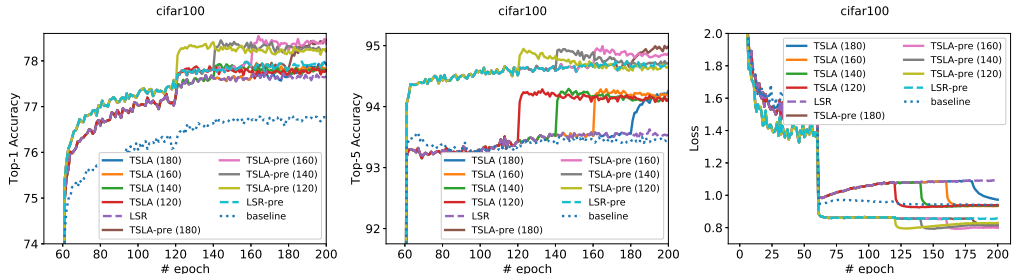

Figure 3: Testing Top-1, Top-5 Accuracy and Loss on ResNet-18 over CIFAR-100. TSLA(s)/TSLA-pre(s) meansTSLA/TSLA-pre drops off LSR/LSR-pre after epoch s.

The total epochs of training ResNet-18 (He et al., 2016) on CIFAR-100 is set to be 200.The weight decay with the parameter value of 5×10^{-4} is used. We use 0.1 as the initial learning rates for all algorithms and divide them by 10 every 60 epochs as suggested in (He et al., 2016; Zagoruyko & Komodakis, 2016). For LSR and the first stage of TSLA, the value of smoothing strength θ is fixed as $\theta = 0.1$, which shows the best performance for LSR. We use two different labels $\widehat{\mathbf{y}}$ to smooth the

one-hot label, the uniform distribution over all labels and the distribution predicted by an ImageNet pre-trained model which is downloaded directly from PyTorch [6] (Paszke et al., 2019). For TSLA, we try to drop off the LSR after s epochs during the training process, where $s \in \{120, 140, 160, 180\}$.

Table 2: Comparison of Testing Accuracy for Different Methods (mean $\pm$ standard deviation, in %).

	CIFAR-100	
Algorithm*	Top-1 accuracy	Top-5 accuracy
baseline	76.87 $\pm$ 0.04	93.47 $\pm$ 0.15
LSR	77.77 $\pm$ 0.18	93.55 $\pm$ 0.11
TSLA(120)	77.92 $\pm$ 0.21	94.13 $\pm$ 0.23
TSLA(140)	77.93 $\pm$ 0.19	94.11 $\pm$ 0.22
TSLA(160)	77.96 $\pm$ 0.20	94.19 $\pm$ 0.21
TSLA(180)	78.04 $\pm$ 0.27	94.23 $\pm$ 0.15
LSR-pre	78.07 $\pm$ 0.31	94.70 $\pm$ 0.14
TSLA-pre(120)	78.34 $\pm$ 0.31	94.68 $\pm$ 0.14
TSLA-pre(140)	78.39 $\pm$ 0.25	94.73 $\pm$ 0.11
TSLA-pre(160)	**78.55** $\pm$ 0.28	94.83 $\pm$ 0.08
TSLA-pre(180)	78.53 $\pm$ 0.23	**94.96** $\pm$ 0.23

*TSLA(s)/TSLA-pre(s): TSLA/TSLA-pre drops off LSR/LSR-pre after epoch s.

All top-1 and top-5 accuracy on the testing data set are averaged over 5 independent random trails with their standard deviations. We summarize the results in Table 2, where LSR-pre and TSLA-pre indicate LSR and TSLA use the label $\widehat{\mathbf{y}}$ based on the ImageNet pre-trained model. The results show that LSR-pre/TSLA-pre has a better performance than LSR/TSLA. The reason might be that the pre-trained model-based prediction is closer to the ground truth than the uniform prediction and it has lower variance (smaller δ). Then, TSLA (LSR) with such pre-trained model-based prediction converges faster than TSLA (LSR) with uniform prediction, which verifies our theoretical findings in Section 4.2 (Section 4.1). This observation also empirically tells us the selection of the prediction function $\widehat{\mathbf{y}}$ used for smoothing label is the key to the success of TSLA as well as LSR. Among all methods, the performance of TSLA-pre is the best. For top-1 accuracy, TSLA-pre(160) outperforms all other algorithms, while for top-5 accuracy, TSLA-pre(180) has the best performance.

Finally, we observe from Figure 3 that both TSLA and TSLA-pre converge, while TSLA-pre converges to the lowest objective value. Similarly, the top-1 and top-5 accuracies show the improvements of TSLA and TSLA-pre at the point of dropping off LSR.

We conduct an **ablation study** for the smoothing parameter θ and the dropping epoch s in TSLA(s). We follow the same settings in Subsection 5.2 but use different values of θ in LSR and TSLA. Specifically, $\theta \in \{0.2, 0.4, 0.9\}$. We use the uniform distribution over all labels to smooth the one-hot label. The results are summarized in Table 3, showing that the different values of θ and s can affect the performances of LSR and TSLA. With the increase of θ, the performances become worse. However, with different values of θ, TSLA can always outperform LSR when an appropriate dropping epoch s is selected. Besides, TSLA(180) has the best performance for each value of θ.

Table 3: Comparison of Testing Top-1 Accuracy for Different θ of LSR and TSLA (in %).

θ	0.2	0.4	0.9
LSR	77.75	77.72	76.40
TSLA(120)	77.92	77.68	76.05
TSLA(140)	78.06	77.61	76.27
TSLA(160)	78.09	77.54	76.37
TSLA(180)	78.10	77.89	76.60

*TSLA(s): TSLA drops off LSR after epoch s.

6 Conclusions

In this paper, we have studied the power of LSR in training deep neural networks by analyzing SGD with LSR in different non-convex optimization settings. The convergence results show that an appropriate LSR with reduced label variance can help speed up the convergence. We have proposed a simple yet efficient strategy so-called TSLA whose basic idea is to switch the training from smoothed label to one-hot label. Integrating TSLA with SGD, we observe from its improved convergence result that TSLA benefits from LSR in the first stage and essentially converges faster in the second stage. Our theoretical result also shows that TSLA has better testing error than that of LSR. Throughout extensive experiments, we have shown that TSLA improves the generalization accuracy of deep models on several benchmark data sets.

[6] https://pytorch.org/docs/stable/torchvision/models.html

REFERENCES

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252, 2019.

Hessam Bagherinezhad, Maxwell Horton, Mohammad Rastegari, and Ali Farhadi. Label refinery: Improving imagenet classification through label progression. *arXiv preprint arXiv:1805.02641*, 2018.

Zachary Charles and Dimitris Papailiopoulos. Stability and generalization of learning algorithms that converge to global optima. In *International Conference on Machine Learning*, pp. 745–754, 2018.

Jan Chorowski and Navdeep Jaitly. Towards better decoding and language model integration in sequence to sequence models. *Proc. Interspeech 2017*, pp. 523–527, 2017.

Qianggang Ding, Sifan Wu, Hao Sun, Jiadong Guo, and Shu-Tao Xia. Adaptive regularization of labels. *arXiv preprint arXiv:1908.05474*, 2019.

Timothy Dozat. Incorporating nesterov momentum into adam. 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

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

Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Math. Program.*, 156(1-2):59–99, 2016.

Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2): 267–305, 2016.

Morgane Goibert and Elvis Dohmatob. Adversarial robustness via adversarial label-smoothing. *arXiv preprint arXiv:1906.11567*, 2019.

Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 558–567, 2019.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. 2012a.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS Deep Learning Workshop*, 2014.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012b.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 795–811. Springer, 2016.

Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, volume 2, 2011.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Technical report, University of Tronto*, 2009.

Xiaoyu Li, Zhenxun Zhuang, and Francesco Orabona. Exponential step sizes for non-convex optimization. *arXiv preprint arXiv:2002.05273*, 2020a.

Xingjian Li, Haoyi Xiong, Haozhe An, Dejing Dou, and Chengzhong Xu. Colam: Co-learning of deep neural networks and soft labels via alternating minimization. *arXiv preprint arXiv:2004.12443*, 2020b.

Zhize Li and Jian Li. A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. In *Advances in Neural Information Processing Systems*, pp. 5564–5574, 2018.

Michal Lukasik, Srinadh Bhojanapalli, Aditya Krishna Menon, and Sanjiv Kumar. Does label smoothing mitigate label noise? *arXiv preprint arXiv:2003.02819*, 2020.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems*, pp. 4696–4705, 2019.

Yurii Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27:372–376, 1983.

Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course. 1998.

Yurii Nesterov. *Introductory lectures on convex optimization : a basic course*. Applied optimization. Kluwer Academic Publ., 2004. ISBN 1-4020-7553-7.

Toan Q Nguyen and Julian Salazar. Transformers without tears: Improving the normalization of self-attention. *arXiv preprint arXiv:1910.05895*, 2019.

Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu. Towards robust detection of adversarial examples. In *Advances in Neural Information Processing Systems*, pp. 4579–4589, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019. URL https://pytorch.org/docs/stable/torchvision/models.html.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

Boris Teodorovich Polyak. Gradient methods for minimizing functionals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 3(4):643–653, 1963.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Chaomin Shen, Yaxin Peng, Guixu Zhang, and Jinsong Fan. Defending against adversarial attacks by suppressing the largest eigenvalue of fisher information matrix. *arXiv preprint arXiv:1909.06137*, 2019.

Patrice Y Simard, Yann A LeCun, John S Denker, and Bernard Victorri. Transformation invariance in pattern recognition—tangent distance and tangent propagation. In *Neural networks: tricks of the trade*, pp. 239–274. Springer, 1998.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Shuo Wang, Zhaopeng Tu, Shuming Shi, and Yang Liu. On the inference calibration of neural machine translation. *arXiv preprint arXiv:2005.00963*, 2020.

Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. Spiderboost and momentum: Faster variance reduction algorithms. In *Advances in Neural Information Processing Systems*, pp. 2403–2413, 2019.

Lingxi Xie, Jingdong Wang, Zhen Wei, Meng Wang, and Qi Tian. Disturblabel: Regularizing cnn on the loss layer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4753–4762, 2016.

Yan Yan, Tianbao Yang, Zhe Li, Qihang Lin, and Yi Yang. A unified analysis of stochastic momentum methods for deep learning. In *International Joint Conference on Artificial Intelligence*, pp. 2955–2961, 2018.

Li Yuan, Francis EH Tay, Guilin Li, Tao Wang, and Jiashi Feng. Revisit knowledge distillation: a teacher-free framework. *arXiv preprint arXiv:1909.11723*, 2019a.

Zhuoning Yuan, Yan Yan, Rong Jin, and Tianbao Yang. Stagewise training accelerates convergence of testing error over sgd. In *Advances in Neural Information Processing Systems*, pp. 2604–2614, 2019b.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *British Machine Vision Conference*, 2016.

Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Albert Zeyer, Kazuki Irie, Ralf Schlüter, and Hermann Ney. Improved training of end-to-end attention models for speech recognition. *Proc. Interspeech 2018*, pp. 7–11, 2018.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8697–8710, 2018.

A ADDITIONAL EXPERIMENTS

All experiments are implemented using PyTorch[7] with version 1.7.1 on a machine with 8 NVIDIA P100 GPUs. The code is available at

https://drive.google.com/drive/folders/1QLvSE6iB2YN7UBpRMKB_kKQGZzK7M4Qr?usp=sharing

As mentioned in previsous sections, the value of δ plays a very important role on the theoretical results. In this section, we provide the estimation of δ for three data sets Stanford Dogs, CUB-2011 and CIFAR-100 that used in the experiments. Our results empirically show that $\delta < 1$ for all three data sets. Specifically, we computed the sample variances (denoted by σ_k^2 and $\widehat{\sigma}_k^2$, $k = 10, 20, \ldots$) of two variance σ^2 and $\widehat{\sigma}^2$ every 10 epochs, then based on the definitions in (4), the etimation (denoted by $\widetilde{\delta}$) of δ was taken as

$$\widetilde{\delta} = \max_k \frac{\widehat{\sigma}_k^2}{\sigma_k^2}.$$

All results of estimated δ are listed in Table 4, empirically showing that $\widetilde{\delta} < 1$. We also plot the estimated values of δ across iterations over three different data sets in Figure 4, showing that $\widetilde{\delta} < 1$ for all data sets across iterations.

Table 4: The estimated value of δ for different data sets

dataset	Stanford Dogs	CUB-2011	CIFAR-100	CIFAR-100 (pretrained)
$\widetilde{\delta}$	0.16	0.13	0.55	0.38

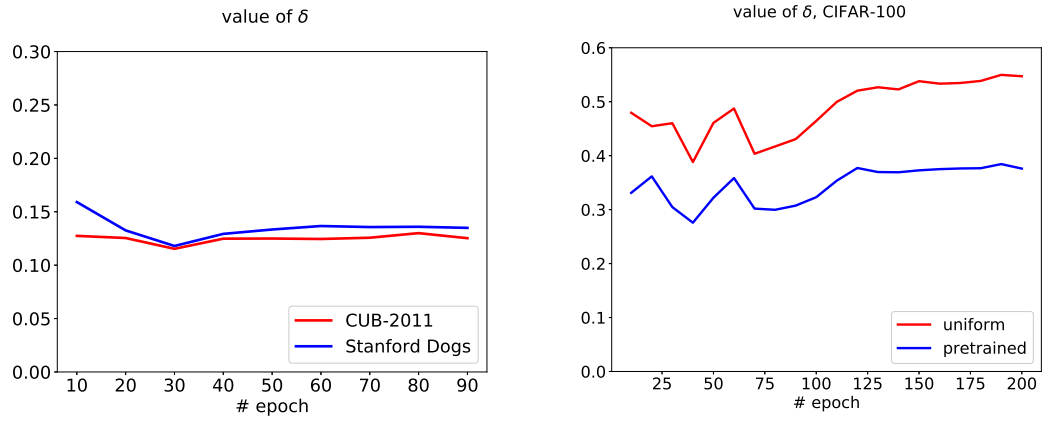

Figure 4: The estimated values of δ for different data sets.

B TECHNICAL LEMMA

Recall that the optimization problem is

$$\min_{\mathbf{w} \in \mathcal{W}} F_{\mathcal{S}}(\mathbf{w}) := \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{y}_i, f(\mathbf{w}; \mathbf{x}_i)) \tag{7}$$

[7] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems, 32: 8026–8037, 2019. https://pytorch.org

where the cross-entropy loss function ℓ is given by

$$\ell(\mathbf{y}, f(\mathbf{w};\mathbf{x})) = \sum_{i=1}^{K} -y_i \log\left(\frac{\exp(f_i(\mathbf{w};\mathbf{x}))}{\sum_{j=1}^{K}\exp(f_j(\mathbf{w};\mathbf{x}))}\right). \tag{8}$$

If we set

$$p(\mathbf{w};\mathbf{x}) = (p_1(\mathbf{w};\mathbf{x}),\ldots,p_K(\mathbf{w};\mathbf{x})) \in \mathbb{R}^K, \quad p_i(\mathbf{w};\mathbf{x}) = -\log\left(\frac{\exp(f_i(\mathbf{w};\mathbf{x}))}{\sum_{j=1}^{K}\exp(f_j(\mathbf{w};\mathbf{x}))}\right), \tag{9}$$

the problem (7) becomes

$$\min_{\mathbf{w}\in\mathcal{W}} F_{\mathcal{S}}(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n}\langle \mathbf{y}_i, p(\mathbf{w};\mathbf{x}_i)\rangle. \tag{10}$$

Then the stochastic gradient with respective to $\mathbf{w}$ is

$$\nabla\ell(\mathbf{y}, f(\mathbf{w};\mathbf{x})) = \langle \mathbf{y}, \nabla p(\mathbf{w};\mathbf{x})\rangle. \tag{11}$$

Lemma 1. *Under Assumption 1 (i), we have*

$$\mathrm{E}\left[\left\|\nabla\ell(\mathbf{y}_t^{LS}, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)\right\|^2\right] \le (1-\theta)\sigma^2 + \theta\delta\sigma^2.$$

Proof. By the facts of $\mathbf{y}_t^{\mathrm{LS}} = (1-\theta)\mathbf{y}_t + \theta\widehat{\mathbf{y}}_t$ and the equation in (11), we have

$$\nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t)) = (1-\theta)\nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t)) + \theta\nabla\ell(\widehat{\mathbf{y}}_t, f(\mathbf{w}_t;\mathbf{x}_t)).$$

Therefore,

$$\begin{aligned}
&\mathrm{E}\left[\left\|\nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)\right\|^2\right] \\
=&\mathrm{E}\left[\left\|(1-\theta)[\nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)] + \theta[\nabla\ell(\widehat{\mathbf{y}}_t, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)]\right\|^2\right] \\
\overset{(a)}{\le}&(1-\theta)\mathrm{E}\left[\left\|\nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)\right\|^2\right] + \theta\mathrm{E}\left[\left\|\nabla\ell(\widehat{\mathbf{y}}_t, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)\right\|^2\right] \\
\overset{(b)}{\le}&(1-\theta)\sigma^2 + \theta\delta\sigma^2,
\end{aligned}$$

where (a) uses the convexity of norm, i.e., $\|(1-\theta)\mathbf{a} + \theta\mathbf{b}\|^2 \le (1-\theta)\|\mathbf{a}\|^2 + \theta\|\mathbf{b}\|^2$; (b) uses assumption 1 (i) and the definitions in (4), and Assumption 1 (i). □

C Proof of Theorem 3

Proof. By the smoothness of objective function $F(\mathbf{w})$ in Assumption 1 (ii) and its remark, we have

$$\begin{aligned}
&F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_t) \\
\le&\langle \nabla F_{\mathcal{S}}(\mathbf{w}_t), \mathbf{w}_{t+1} - \mathbf{w}_t\rangle + \frac{L}{2}\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2 \\
\overset{(a)}{=}&-\eta\left\langle \nabla F_{\mathcal{S}}(\mathbf{w}_t), \nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t))\right\rangle + \frac{\eta^2 L}{2}\left\|\nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t))\right\|^2 \\
\overset{(b)}{=}&-\frac{\eta}{2}\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2 + \frac{\eta}{2}\left\|\nabla F_{\mathcal{S}}(\mathbf{w}_t) - \nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t))\right\|^2 + \frac{\eta(\eta L - 1)}{2}\left\|\nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t))\right\|^2 \\
\overset{(c)}{\le}&-\frac{\eta}{2}\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2 + \frac{\eta}{2}\left\|\nabla F_{\mathcal{S}}(\mathbf{w}_t) - \nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t))\right\|^2,
\end{aligned} \tag{12}$$

where (a) is due to the update of $\mathbf{w}_{t+1}$; (b) is due to $\langle \mathbf{a}, -\mathbf{b}\rangle = \frac{1}{2}\left(\|\mathbf{a}-\mathbf{b}\|^2 - \|\mathbf{a}\|^2 - \|\mathbf{b}\|^2\right)$; (c) is due to $\eta \le \frac{1}{L}$. Taking the expectation over $(\mathbf{x}_t, \mathbf{y}_t^{\mathrm{LS}})$ on the both sides of (12), we have

$$\begin{aligned}
&\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_t)\right] \\
\le&-\frac{\eta}{2}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta}{2}\mathrm{E}\left[\left\|\nabla F_{\mathcal{S}}(\mathbf{w}_t) - \nabla\ell(\mathbf{y}_t^{\mathrm{LS}}, f(\mathbf{w}_t;\mathbf{x}_t))\right\|^2\right] \\
\le&-\frac{\eta}{2}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta}{2}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right).
\end{aligned} \tag{13}$$

where the last inequality is due to Lemma 1. Then inequality (13) implies

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \leq & \frac{2F_{\mathcal{S}}(\mathbf{w}_0)}{\eta T} + (1-\theta)\sigma^2 + \theta\delta\sigma^2 \\ \overset{(a)}{=} & \frac{2F_{\mathcal{S}}(\mathbf{w}_0)}{\eta T} + \frac{2\delta}{1+\delta}\sigma^2 \\ \overset{(b)}{\leq} & \frac{2F_{\mathcal{S}}(\mathbf{w}_0)}{\eta T} + 2\delta\sigma^2,\end{aligned}$$

where (a) is due to $\theta = \frac{1}{1+\delta}$; (b) is due to $\frac{1}{1+\delta} \leq 1$. □

D Convergence Analysis of SGD without LSR ($\theta = 0$)

Theorem 6. *Under Assumption 1, the solutions w_t from Algorithm 1 with $\theta = 0$ satisfy*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \leq \frac{2F_{\mathcal{S}}(\mathbf{w}_0)}{\eta T} + \eta L\sigma^2.$$

In order to have $\mathrm{E}_R[\|\nabla F_{\mathcal{S}}(\mathbf{w}_R)\|^2] \leq \epsilon^2$, it suffices to set $\eta = \min\left(\frac{1}{L}, \frac{\epsilon^2}{2L\sigma^2}\right)$ and $T = \frac{4F_{\mathcal{S}}(\mathbf{w}_0)}{\eta\epsilon^2}$, the total complexity is $O\left(\frac{1}{\epsilon^4}\right)$.

Proof. By the smoothness of objective function $F_{\mathcal{S}}(\mathbf{w})$ in Assumption 1 (ii) and its remark, we have

$$\begin{aligned}& F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_t) \\ \leq & \langle\nabla F_{\mathcal{S}}(\mathbf{w}_t), \mathbf{w}_{t+1} - \mathbf{w}_t\rangle + \frac{L}{2}\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2 \\ \overset{(a)}{=} & -\eta\left\langle\nabla F_{\mathcal{S}}(\mathbf{w}_t), \nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t))\right\rangle + \frac{\eta^2 L}{2}\left\|\nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t))\right\|^2,\end{aligned} \tag{14}$$

where (a) is due to the update of $\mathbf{w}_{t+1}$. Taking the expectation over $(\mathbf{x}_t;\mathbf{y}_t)$ on the both sides of (14), we have

$$\begin{aligned}& \mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_t)\right] \\ \overset{(a)}{\leq} & -\eta\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta^2 L}{2}\mathrm{E}\left[\|\nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t) + \nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \\ \overset{(b)}{=} & -\eta\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta^2 L}{2}\mathrm{E}\left[\|\nabla\ell(\mathbf{y}_t, f(\mathbf{w}_t;\mathbf{x}_t)) - \nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta^2 L}{2}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \\ \overset{(c)}{\leq} & -\frac{\eta}{2}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta^2 L}{2}\sigma^2.\end{aligned} \tag{15}$$

where (a) and (b) use Assumption 1 (i); (c) uses the facts that $\eta \leq \frac{1}{L}$ and Assumption 1 (i). The inequality (15) implies

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \leq \frac{2F_{\mathcal{S}}(\mathbf{w}_0)}{\eta T} + \eta L\sigma^2.$$

By setting $\eta \leq \frac{\epsilon^2}{2L\sigma^2}$ and $T = \frac{4F_{\mathcal{S}}(\mathbf{w}_0)}{\eta\epsilon^2}$, we have $\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$. Thus the total complexity is in the order of $O\left(\frac{1}{\eta\epsilon^2}\right) = O\left(\frac{1}{\epsilon^4}\right)$. □

E Proof of Theorem 4

Proof. Following the similar analysis of inequality (13) from the proof of Theorem 3, we have

$$\begin{aligned}& \mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_t)\right] \\ \leq & -\frac{\eta_1}{2}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta_1}{2}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right).\end{aligned} \tag{16}$$

Using the condition in Assumption 2 we can simplify the inequality from (16) as

$$\begin{aligned}
&\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)\right] \\
\leq &(1-\eta_1\mu)\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_t) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)\right] + \frac{\eta_1}{2}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right) \\
\leq &(1-\eta_1\mu)^{t+1}\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_0) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)\right] + \frac{\eta_1}{2}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right)\sum_{i=0}^{t}(1-\eta_1\mu)^i \\
\leq &(1-\eta_1\mu)^{t+1}\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_0)\right] + \frac{\eta_1}{2}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right)\sum_{i=0}^{t}(1-\eta_1\mu)^i,
\end{aligned}$$

where the last inequality is due to the definition of loss function that $F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*) \geq 0$. Since $\eta_1 \leq \frac{1}{L} < \frac{1}{\mu}$, then $(1-\eta_1\mu)^{t+1} < \exp(-\eta_1\mu(t+1))$ and $\sum_{i=0}^{t}(1-\eta_1\mu)^i \leq \frac{1}{\eta_1\mu}$. As a result, for any T_1, we have

$$\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{T_1}) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)\right] \leq \exp(-\eta_1\mu T_1)F_{\mathcal{S}}(\mathbf{w}_0) + \frac{1}{2\mu}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right). \tag{17}$$

Let $\theta = \frac{1}{1+\delta}$ and $\widehat{\sigma}^2 := (1-\theta)\sigma^2 + \theta\delta\sigma^2 = \frac{2\delta}{1+\delta}\sigma^2$ then $\frac{1}{2\mu}\left((1-\theta)\sigma^2 + \theta\delta\sigma^2\right) = \frac{\delta\sigma^2}{\mu(1+\delta)} \leq F_{\mathcal{S}}(\mathbf{w}_0)$ since δ is small enough. By setting

$$T_1 = \log\left(\frac{2\mu F_{\mathcal{S}}(\mathbf{w}_0)}{\widehat{\sigma}^2}\right)/(\eta_1\mu)$$

we have

$$\mathrm{E}\left[F(\mathbf{w}_{T_1}) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)\right] \leq \frac{\widehat{\sigma}^2}{\mu} \leq \frac{2\delta\sigma^2}{\mu}. \tag{18}$$

After T_1 iterations, we drop off the label smoothing, i.e. $\theta = 0$, then we know for any $t \geq T_1$, following the inequality (15) from the proof of Theorem 6, we have

$$\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{t+1}) - F_{\mathcal{S}}(\mathbf{w}_t)\right] \leq -\frac{\eta_2}{2}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] + \frac{\eta_2^2 L\sigma^2}{2}.$$

Therefore, we get

$$\begin{aligned}
\frac{1}{T_2}\sum_{t=T_1}^{T_1+T_2-1}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \leq &\frac{2}{\eta_2 T_2}\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{T_1}) - F_{\mathcal{S}}(\mathbf{w}_{T_1+T_2-1})\right] + \eta_2 L\sigma^2 \\
\overset{(a)}{\leq} &\frac{2}{\eta_2 T_2}\mathrm{E}\left[F_{\mathcal{S}}(\mathbf{w}_{T_1}) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)\right] + \eta_2 L\sigma^2 \\
\overset{(18)}{\leq} &\frac{4\delta\sigma^2}{\mu\eta_2 T_2} + \eta_2 L\sigma^2,
\end{aligned} \tag{19}$$

where (a) is due to $F_{\mathcal{S}}(\mathbf{w}_{T_1+T_2-1}) \geq F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)$. By setting $\eta_2 = \frac{\epsilon^2}{2L\sigma^2}$ and $T_2 = \frac{8\delta\sigma^2}{\mu\eta_2\epsilon^2}$, we have $\frac{1}{T_2}\sum_{t=T_1}^{T_1+T_2-1}\mathrm{E}\left[\|\nabla F_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \leq \epsilon^2$. □

F Generalization Analysis

In this section, let

$$\widehat{F}_{\mathcal{S}}(\mathbf{w}) = F_{\mathcal{S}}(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|^2 = \frac{1}{n}\sum_{i=1}^{n}\underbrace{\ell(\mathbf{y}_i, f(\mathbf{w};\mathbf{x}_i)) + \frac{\lambda}{2}\|\mathbf{w}\|^2}_{\widehat{\ell}(\mathbf{y}_i, f(\mathbf{w};\mathbf{x}_i))} \tag{20}$$

$$\widehat{F}(\mathbf{w}) = F(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{21}$$

Following by Yuan et al. (2019b), we use the following decomposition of testing error.

$$\mathrm{E}_{\mathcal{A},\mathcal{S}}[F(\mathbf{w}_{\mathcal{S}})] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)] \leq \mathrm{E}_{\mathcal{S}}[\mathrm{E}_{\mathcal{A}}[F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}}^*)]] + \mathrm{E}_{\mathcal{A},\mathcal{S}}[F(\mathbf{w}_{\mathcal{S}}) - F_{\mathcal{S}}(\mathbf{w}_{\mathcal{S}})]. \tag{22}$$

Let define the gradient update rule $\mathcal{G}_{\widehat{\ell},\eta}$ as follows

$$\mathcal{G}_{\widehat{\ell},\eta}(\mathbf{w}) = \mathbf{w} - \eta\nabla_{\mathbf{w}}\widehat{\ell}(\mathbf{y}, f(\mathbf{w}, \mathbf{x})). \tag{23}$$

Then we have the following lemma, which is similar to Lemma 2.5 and Lemma 4.2 in Hardt et al. (2015).

Lemma 2. *Assume that $\ell(\mathbf{y}, f(\mathbf{w}, \mathbf{x}))$ is L-smooth and B-Lipschitz. Let $\mathbf{w}_{t+1} = \mathcal{G}(\mathbf{w}_t)$ and $\mathbf{w}'_{t+1} = \mathcal{G}'(\mathbf{w}'_t)$, then*

$$\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\| = \begin{cases} (1 + \eta L - \eta\lambda)\|\mathbf{w}_t - \mathbf{w}'_t\|, & \mathcal{G} = \mathcal{G}', \\ (1 - \eta\lambda)\|\mathbf{w}_t - \mathbf{w}'_t\| + 2\eta B, & \mathcal{G} \neq \mathcal{G}'. \end{cases}$$

Proof. Recall that $\widehat{\ell}(\mathbf{y}_i, f(\mathbf{w}; \mathbf{x}_i)) = \ell(\mathbf{y}_i, f(\mathbf{w}; \mathbf{x}_i)) + \frac{\lambda}{2}\|\mathbf{w}\|^2$. Then
(1) When $\mathcal{G} = \mathcal{G}'$ and $\ell(\cdot, f(\mathbf{w}; \cdot))$ is L-smooth, then

$$\begin{aligned}\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\| \leq& (1 - \eta\lambda)\|\mathbf{w}_t - \mathbf{w}'_t\| + \eta\|\nabla_{\mathbf{w}}\ell(\mathbf{y}_t, f(\mathbf{w}_t; \mathbf{x}_t)) - \nabla_{\mathbf{w}}\ell(\mathbf{y}_t, f(\mathbf{w}'_t; \mathbf{x}_t))\| \\ \leq& (1 + \eta L - \eta\lambda)\|\mathbf{w}_t - \mathbf{w}'_t\|.\end{aligned}$$

(2) When $\mathcal{G} \neq \mathcal{G}'$ and $\ell(\cdot, f(\mathbf{w}; \cdot))$ is B-Lipschitz, then

$$\begin{aligned}\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\| \leq& (1 - \eta\lambda)\|\mathbf{w}_t - \mathbf{w}'_t\| + \eta\|\nabla_{\mathbf{w}}\ell(\mathbf{y}_t, f(\mathbf{w}_t; \mathbf{x}_t)) - \nabla_{\mathbf{w}}\ell(\mathbf{y}'_t, f(\mathbf{w}'_t; \mathbf{x}'_t))\| \\ \leq& (1 - \eta\lambda)\|\mathbf{w}_t - \mathbf{w}'_t\| + 2\eta B.\end{aligned}$$

□

F.1 Proof of Theorem 5

Proof. By Theorem 3.2 of Hardt et al. (2015), we have

$$\begin{aligned}\Delta_{t+1} :=& \mathrm{E}[\|\mathbf{w}_{t+1} - \mathbf{w}'_{t+1}\|] \\ \overset{(a)}{\leq}& \left(1 - \frac{1}{n}\right)(1 + \eta L - \eta\lambda)\Delta_t + \frac{1}{n}((1 - \eta\lambda)\Delta_t + 2\eta_t B) \\ =& \left[1 + \left(1 - \frac{1}{n}\right)\eta L - \eta\lambda\right]\Delta_t + \frac{2\eta B}{n} \\ \overset{(b)}{\leq}& (1 - \eta L)\Delta_t + \frac{2\eta B}{n} \overset{(c)}{\leq} \frac{2\eta B}{n}\sum_{i=0}^{t_0}(1 - \eta L)^i \overset{(d)}{\leq} \frac{2B}{nL},\end{aligned} \tag{24}$$

where (a) uses Lemma 2; (b) uses $\lambda = 2L$; (c) uses $\Delta_{t_0} = 0$; (d) uses $\eta L < 1$. Then by Lemma 3.11 of Hardt et al. (2015) and $\widehat{\ell}$ is $L + \lambda$-smooth, we have

$$\mathrm{E}[\widehat{\ell}(\mathbf{w}_t; \mathbf{z}) - \widehat{\ell}(\mathbf{w}'_t; \mathbf{z})] \leq \frac{t_0}{n} + \frac{2B(L + \lambda)}{nL} \leq \frac{6B + 1}{n}, \tag{25}$$

where the last inequality holds by selecting $t_0 = 1$ and $\lambda = 2L$. Then by Theorem 2.2 of Hardt et al. (2015)

$$\mathrm{E}_{\mathcal{A},\mathcal{S}}[\widehat{F}(\mathbf{w}_t) - \widehat{F}_{\mathcal{S}}(\mathbf{w}_t)] \leq \frac{6B + 1}{n}, \tag{26}$$

which is the generalization error. Next, we will bound the optimization error. Since $\widehat{F}_{\mathcal{S}}$ is $(L + \lambda)$-smooth and satisfies μ-PL condition, then follow the similar analysis of Theorem 4, we have

$$\begin{aligned}\frac{1}{T_2}\sum_{t=T_1}^{T_1+T_2-1}\mathrm{E}_{\mathcal{S}}[\mathrm{E}_{\mathcal{A}}[\widehat{F}_{\mathcal{S}}(\mathbf{w}_t) - \widehat{F}_{\mathcal{S}}(\widehat{\mathbf{w}}_{\mathcal{S}}^*)]] \leq& \frac{1}{2\mu T_2}\sum_{t=T_1}^{T_1+T_2-1}\mathrm{E}\left[\|\nabla\widehat{F}_{\mathcal{S}}(\mathbf{w}_t)\|^2\right] \\ \leq& \frac{2\delta\sigma^2}{\mu^2\eta_2 T_2} + \frac{\eta_2(L + \lambda)\sigma^2}{2\mu},\end{aligned} \tag{27}$$

Then by (22), (26) and (27), we get

$$\frac{1}{T_2}\sum_{t=T_1}^{T_1+T_2-1} \mathrm{E}_{\mathcal{A},\mathcal{S}}[\widehat{F}(\mathbf{w}_t)] - \mathrm{E}_{\mathcal{S}}[\widehat{F}_{\mathcal{S}}(\widehat{\mathbf{w}}^*_{\mathcal{S}})] \leq \frac{2\delta\sigma^2}{\mu^2\eta_2 T_2} + \frac{\eta_2(L+\lambda)\sigma^2}{2\mu} + \frac{6B+1}{n}. \tag{28}$$

By using the conditions that $\min_{\mathbf{w}\in\mathcal{W}} \widehat{F}_{\mathcal{S}}(\mathbf{w}) \leq F_{\mathcal{S}}(\mathbf{w}^*_S) + \frac{\lambda}{2}\|\mathbf{w}_t\|^2$, definitions (20) and (21), we get

$$F(\mathbf{w}_t) - F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}}) \leq \widehat{F}(\mathbf{w}_t) - \widehat{F}_{\mathcal{S}}(\widehat{\mathbf{w}}^*_{\mathcal{S}}). \tag{29}$$

Therefore, we have the following inequality by (28) and (29):

$$\frac{1}{T_2}\sum_{t=T_1}^{T_1+T_2-1} \mathrm{E}_{\mathcal{A},\mathcal{S}}[F(\mathbf{w}_t)] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})] \leq \frac{2\delta\sigma^2}{\mu^2\eta_2 T_2} + \frac{\eta_2(L+\lambda)\sigma^2}{2\mu} + \frac{6B+1}{n}. \tag{30}$$

By setting $\eta_2 = O(1/\sqrt{n})$ and $T_2 = O(\delta n)$ then the testing error $\mathrm{E}_{R,\mathcal{A},\mathcal{S}}[F(\mathbf{w}_R)] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})] \leq O(1/\sqrt{n})$, where $\mathcal{A}$ is TSLA. □

Remark 1. (SGD without LSR) Following the similar analysis of Theorem 5 and similar analysis of Theorem 6, we have the testing error for SGD without LSR is

$$\mathrm{E}_{R,\mathcal{A},\mathcal{S}}[F(\mathbf{w}_R)] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})] \leq \frac{\widehat{F}_{\mathcal{S}}(\mathbf{w}_0)}{\mu\eta T} + \frac{\eta(L+\lambda)\sigma^2}{2\mu} + \frac{6B+1}{n}. \tag{31}$$

By setting $\eta = O(1/\sqrt{n})$ and $T = O(n)$ then $\mathrm{E}_{R,\mathcal{A},\mathcal{S}}[F(\mathbf{w}_R)] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})] \leq O(1/\sqrt{n})$, where $\mathcal{A}$ is SGD without LSR.

Remark 2. (SGD with LSR) Following the similar analysis of Theorem 5 and similar analysis of Theorem 3, we have the testing error for SGD with LSR is

$$\mathrm{E}_{R,\mathcal{A},\mathcal{S}}[F(\mathbf{w}_R)] - \mathrm{E}_{\mathcal{S}}[F_{\mathcal{S}}(\mathbf{w}^*_{\mathcal{S}})] \leq \frac{\widehat{F}_{\mathcal{S}}(\mathbf{w}_0)}{\mu\eta T} + \frac{\delta\sigma^2}{\mu} + \frac{6B+1}{n}, \tag{32}$$

where $\mathcal{A}$ is SGD with LSR. If $\delta > \Omega(1/\sqrt{n})$, the testing error can not be bounded by $O(1/\sqrt{n})$; if $\delta \leq \Omega(1/\sqrt{n})$, the testing error can be bounded by $O(1/\sqrt{n})$ in $T = O(\sqrt{n})$ iterations.