DTN: A Learning Rate Scheme with Convergence Rate of $\mathcal{O}(1/t)$ for SGD

Lam M. Nguyen [1] **Phuong Ha Nguyen** [2] **Dzung T. Phan** [1] **Jayant R. Kalagnanam** [1] **Marten van Dijk** [2]

Abstract

We propose a novel diminishing learning rate scheme, coined **D**ecreasing-**T**rend-**N**ature (DTN), which allows us to prove fast convergence of the Stochastic Gradient Descent (SGD) algorithm to a first-order stationary point for *smooth general convex* and *some class of nonconvex* including *neural network* applications for classification problems. We are the first to prove that SGD with diminishing learning rate achieves a convergence rate of $\mathcal{O}(1/t)$ for these problems. Our theory applies to *neural network* applications for classification problems in a straightforward way.

1. Introduction

Machine learning (ML) has become a very popular, fast developing and powerful tool for data analytics, partly due to breakthroughs achieved by applying deep learning to several big-data applications over the last years. One of the most active areas of research related to machine learning is the development of optimization algorithms to train deep neural networks (DNNs) as well as minimize empirical risk losses. The optimization problem for training many ML models, including DNNs, using a training set $\{(x_i, y_i)\}_{i=1}^m$ of m samples can be formulated as a finite-sum minimization problem as follows

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ F(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} f(\mathbf{w}; x_i, y_i) \right\}. \tag{1}$$

The objective is to minimize a loss function with respect to model parameters $\mathbf{w}$, such as, for example, weights of all layers of a deep neural network. This problem is known as empirical risk minimization and it covers a wide range of convex and nonconvex problems from the ML domain, including, but not limited to, logistic regression, multi-kernel learning, conditional random fields and neural networks.

[1]IBM Research, Thomas J. Watson Research Center, Yorktown Heights, NY, USA. [2]Department of Electrical and Computer Engineering, University of Connecticut, USA. Correspondence to: Lam M. Nguyen <LamNguyen.MLTD@ibm.com>.

In this paper we are interested in deriving a new *diminishing learning rate* scheme for the stochastic gradient descent (SGD) algorithm for solving the following more general problem. We consider the stochastic optimization problem with respect to some distribution $\mathcal{D}$:

$$\min_{\mathbf{w}\in\mathbb{R}^d} \left\{ F(\mathbf{w}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[f(\mathbf{w}; x, y)] \right\}, \tag{2}$$

where F has a Lipschitz continuous gradient and f has a *finite lower bound* for every (x, y).

At each iteration, SGD randomly updates iterates as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t; x_t, y_t),\ t \geq 0,$$

for a realization of a random variable (x_t, y_t), where η_t is the learning rate. Thanks to its simplicity in implementation and efficiency in dealing with large scale datasets, stochastic gradient descent, originally introduced in (Robbins & Monro, 1951), has become the method of choice for solving not only (1) but also (2) when m is large. With a fixed learning rate $\eta_t = \eta$, the method may provide fast initial improvement, after which it oscillates within a region containing a solution (Bottou et al., 2018; Chee & Toulis, 2018). The algorithm is only guaranteed to converge to within some range of the optimal value, i.e., we have

$$\lim_{t\to\infty} \mathbb{E}[F(\mathbf{w}_t) - F_*] \leq \delta,$$

where F_* is the optimal value of the problem and $\delta = \mathcal{O}(\eta)$ even for strongly convex cases. Hence, SGD can fail to converge to a solution. It is known that behavior of SGD is strongly dependent on the chosen learning rate and on the variance of the stochastic gradients. To overcome this issue, there are two main lines of research that have been proposed in literature: variance reduction methods and diminishing learning rate schemes. These algorithms guarantee to converge to the optimal value, see related work at the end of this section.

SGD and its variants such as AdaGrad (Duchi et al., 2011), RMSProp (Tieleman & Hinton, 2012), Adam (Kingma & Ba, 2014), ADADELTA (Zeiler, 2012), etc., are widely used in dealing with large scale datasets. Recently, many works have tried to provide convergence theory for deep neural networks (see e.g. (Li & Yuan, 2017; Boob & Lan, 2017; Zou et al., 2018), etc). Nevertheless, there are only a very limited

number of theoretical works in deep learning that explain why the above methods are efficient for training neural networks. These studies teach us that SGD and its variants have weaker theoretical convergence rates compared to variance reduction methods. Still, due to their efficiency, SGD and its variants are a competitive choice when considering training large-scale and online problems, especially in neural network applications. Motivated by the work of (Nguyen et al., 2018a) and due to the efficiency of SGD-like algorithms for training neural networks, especially for classification problems, we realize that it may be true *deep neural networks have some special properties that SGD (and its variants) can take advantage of.*

We introduce a new notion, the so-called D-*slow* condition (see (7)), quantifying the growth rate of the gradient in relation to the function value. We show that D-slow is a special property that is satisfied by a feed forward neural network with softmax cross entropy loss functions. Besides assuming the D-slow condition, our analysis relies on the assumption of the existence of a lower bound $\hat{f}$ for f, which widely holds true in practice. For example, in supervised learning such as least squares regression, logistic regression, and deep learning the loss functions are usually non-negative, thus we can take $\hat{f} = 0$. The assumption is also valid in other domains such as the source localization and tracking problem in wireless sensor networks (Blatt et al., 2007).

Given the above assumptions, we introduce a novel diminishing learning rate scheme called ***D**ecreasing-**T**rend-**N**ature (DTN) update* defined by

$$\eta_{t+1} = \frac{2\eta_t}{1 + \sqrt{1 + 4\alpha' \eta_t^2}} \tag{3}$$

with initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha'}}$, for some $\alpha' > 0$, where α' is called the *DTN parameter*. The DTN update scheme for the learning rate allows us to prove $\mathcal{O}(1/t)$ convergence rate.

Contributions. We summarize our key contributions as follows.

- We propose a novel diminishing learning rate scheme called ***D**ecreasing-**T**rend-**N**ature update* given by (3). Given the D-slow condition in (7), we show a $\mathcal{O}(1/t)$ convergence rate of SGD with DTN updates to a first-order stationary point for smooth nonconvex problems (Theorem 1). We show the D-slow condition holds for feed forward neural network with softmax cross entropy loss functions (Section 3). We notice that $\mathcal{O}(1/t)$ convergence rate matches the convergence rate of Gradient Descent for these problems.
- We show that the D-slow condition always holds for smooth general convex problems. As a consequence we inherit the $\mathcal{O}(1/t)$ convergence rate to a first-order stationary point for the general convex case (Theorem 2).
- We present an asymptotic analysis of η_t for the proposed DTN scheme and show that $\eta_t = \Theta(1/\sqrt{t})$ (Corollary 3). Up to the best of our knowledge, this is the first work able to provide a convergence guarantee for this order of learning rate without needing to satisfy the standard condition $\sum_{t=0}^{\infty} \eta_t^2 < \infty$ in the general convex and nonconvex cases.

Related Work. In recent years, there is an abundance of attempts dedicated to designing efficient variance reduction algorithms: SAG/SAGA (Schmidt et al., 2016; Defazio et al., 2014), SDCA (Shalev-Shwartz & Zhang, 2013), MISO (Mairal, 2013), SVRG/S2GD (Johnson & Zhang, 2013; Konečný & Richtárik, 2013), SARAH (Nguyen et al., 2017a), etc. Variance reduction methods were first analyzed for strongly convex problems of form (1) for asymptotic convergence to an optimal solution. Due to recent interest in deep neural networks, *nonconvex* problems of form (1) have been studied and analyzed by considering a number of different approaches including many variants of variance reduction techniques and show improved convergence results over SGD (see e.g. (Reddi et al., 2016; Lei et al., 2017; Nguyen et al., 2017b; Allen-Zhu, 2017; Fang et al., 2018), etc.) Theoretically, they are among the best in terms of algorithmic complexity and improve over regular SGDs. However, these methods reduce the variance of the stochastic gradient estimates by either computing a full gradient after a certain number of iterations or by storing the past gradients, both of which can be expensive. Therefore, in many cases such as training DNNs with huge datasets they are often not preferred over the standard or momentum SGDs.

Diminishing learning rate methods have good theoretical convergence properties. They can avoid oscillatory behavior provided that the learning rate diminishes sufficiently fast but still maintains substantial progress. A general requirement is that $\sum_{t=0}^{\infty} \eta_t = \infty$ and $\sum_{t=0}^{\infty} \eta_t^2 < \infty$. There are limited explicit rules in the literature, one typical example is $\eta_t = a/(b+t)$, where $a > 0$ and $b \geq 0$. Some results show a convergence rate of $\mathcal{O}(1/t)$ for SGD in the strongly convex case (see e.g. (Moulines & Bach, 2011; Bottou et al., 2018; Nguyen et al., 2018c)). In terms of convergence rate, the standard result for SGD is $\mathcal{O}\left(1/\sqrt{t}\right)$ for general convex (see e.g. (Nemirovsky & Yudin, 1983; Nemirovski et al., 2009)) and nonconvex under bounded variance assumption (see e.g. (Ghadimi & Lan, 2013)) with fixed learning rate. In this paper, we propose a recursive diminishing learning rate rule $\eta_t = \phi(\eta_{t-1})$ (DTN) which achieves the convergence rate of $\mathcal{O}(1/t)$ for general convex cases and a class of nonconvex problems arising from a popular deep neural network architecture. We are the first to prove/achieve

a convergence rate of $\mathcal{O}(1/t)$ for SGD in general convex and nonconvex practical/realistic cases with diminishing learning rate.

Here we note that very recently, a convergence rate $\mathcal{O}(1/t)$ for SGD with fixed learning rate is obtained in (Vaswani et al., 2018) under two strong assumptions: *strong growth condition (SGC)* (Schmidt & Roux, 2013) and *weak growth condition (WGC)*. A necessary condition for satisfying these conditions is that any stationary point of $F(\mathbf{w})$ is also a stationary point of $f(\mathbf{w}; x, y)$ for every (x, y). It follows that minimizing $F(\mathbf{w})$ can be reduced to the problem of minimizing a component $f(\mathbf{w}; x, y)$ for any (x, y). This assumption is not satisfied by most realistic machine learning problems.

2. General Framework for Fast Convergence of SGD

We show that the D-slow condition (see (7)) and the DTN diminishing learning rate scheme (see (3)) constitute a general framework for proving the convergence of $\mathcal{O}(1/t)$ for SGD in the non-convex (see Section 2.2) and general convex (see Section 2.3) cases.

Specifically, we are interested in finding a stationary point of problem (2), where F has a Lipschitz continuous gradient with constant $L > 0$ (L-smooth) and f has a finite lower bound $\hat{f}$ for every (x, y). We aim to find an ϵ-accurate solution $w_{\mathcal{T}}$ with satisfying criterion $\mathbb{E}[\|\nabla F(\mathbf{w}_{\mathcal{T}})\|^2] \leq \epsilon$.

Algorithm 1 describes the SGD algorithm for solving (2). For applying SGD in the general form (2), the assumption of unbiased gradient estimators, i.e.,

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\nabla f(\mathbf{w}; x, y)] = \nabla F(\mathbf{w}),\ \forall \mathbf{w} \in \mathbb{R}^d,$$

is requested. For simplicity, we will drop the notation $\mathbb{E}_{(x,y)\sim\mathcal{D}}$ and keep using only notation $\mathbb{E}$ to represent the expectation from now.

Algorithm 1 Stochastic Gradient Descent (SGD) Method

Initialize: $\mathbf{w}_0$
Iterate:
for $t = 0, 1, \ldots$ **do**
  Choose a step size (i.e., learning rate) $\eta_t > 0$.
  Generate a realization of a random variable (x_t, y_t).
  Compute a stochastic gradient $\nabla f(\mathbf{w}_t; x_t, y_t)$.
  Update the new iterate $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t; x_t, y_t)$.
end for

We use the following definitions in this paper.

Definition 1 (L-smooth)**.** *A function ϕ is L-smooth if there exists a constant $L > 0$ such that for $\forall \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d$,*

$$\|\nabla\phi(\mathbf{w}) - \nabla\phi(\mathbf{w}')\| \leq L\|\mathbf{w} - \mathbf{w}'\|. \tag{4}$$

If ϕ is L-smooth, then by (Nesterov, 2004), for any $\mathbf{w}$, $\mathbf{w}' \in \mathbb{R}^d$,

$$\phi(\mathbf{w}) \leq \phi(\mathbf{w}') + \nabla\phi(\mathbf{w}')^T(\mathbf{w} - \mathbf{w}') + \frac{L}{2}\|\mathbf{w} - \mathbf{w}'\|^2. \tag{5}$$

Definition 2 (D-slow)**.** *A function ϕ, $\phi(\mathbf{w}) \geq c$, for $\forall \mathbf{w} \in \mathbb{R}^d$, is D-slow over c with $c \in \mathbb{R}$ if there exists a constant $D > 0$ such that, for every $\mathbf{w} \in \mathbb{R}^d$,*

$$\|\nabla\phi(\mathbf{w})\|^2 \leq 4D(\phi(\mathbf{w}) - c). \tag{6}$$

In this paper, we consider a function ϕ for which $\phi(\mathbf{w}) \geq \hat{f}$, for $\forall \mathbf{w} \in \mathbb{R}^d$. Hence, by understanding $c = \hat{f}$, we will simply say that function ϕ is *D-slow*. We also say that a function ϕ is *D-slow for a set $\mathcal{W}$* if (6) holds for every $\mathbf{w} \in \mathcal{W}$.

We notice that if the value c is the optimal function value for $\phi(\mathbf{w})$, then this implies $\nabla\phi(\mathbf{w}) = 0$. The D-slow property gives rise to the first-order optimality condition for the global minimum. In this paper, we are considering a particular class of nonconvex functions which satisfy the D-slow property of Definition 2. Later, in Section 3, we will show that for classification problems in deep learning applications for feed-forward neural networks, the softmax cross entropy loss functions are D-slow functions.

2.1. D-slow Condition

We recall that $\hat{f}$ is a lower bound of f. Clearly, if f is D-slow over $\hat{f}$ for every realization of (x, y), then F in (2) is also D-slow over $\hat{f}$. Moreover, it has the following property by taking the expectation.

$$\mathbb{E}[\|\nabla f(\mathbf{w}; x, y)\|^2] \leq 4D(F(\mathbf{w}) - \hat{f}). \tag{7}$$

We call the condition in (7) as *D-slow condition*.

We first prove that the D-slow condition is more acceptable than the *strong growth condition* (SGC) and *weak growth condition* (WGC). After this, we show it is more reasonable than the *bounded gradient assumption*. We observe that the bounded gradient assumption is intensively used to prove the convergence of SGD in many scenarios. Hence, the D-slow condition significantly improves the quality of the presented SGD analysis.

We recall that $F(\mathbf{w}) \geq \hat{f}$, for $\forall \mathbf{w} \in \mathbb{R}^d$. Hence, if $F(\mathbf{w}) = \hat{f}$, then this implies $f(\mathbf{w}; x, y) = \hat{f}$ and therefore $\|\nabla f(\mathbf{w}; x, y)\|^2 = 0$ for every (x, y) since the losses are attaining the global minimum value. We notice that $\hat{f}$ is not necessarily an optimal minimum value of F. It is clear that $\|\nabla F(\mathbf{w})\|^2 = 0$ does not necessarily imply $\mathbb{E}[\|\nabla f(\mathbf{w}; x, y)\|^2] = 0$. *Hence, our condition does not*

violate the critical issue of SGC and WGC as explained in the previous section.

Note that the bounded gradient assumption, i.e. there exists a $G > 0$ such that $\mathbb{E}[\|\nabla f(\mathbf{w}; x, y)\|^2] \leq G$, for $\forall \mathbf{w} \in \mathbb{R}^d$, is a very common assumption for nonconvex problems. Given this assumption we have

$$\begin{aligned}\mathbb{E}[\|\nabla f(\mathbf{w}; x, y)\|^2] \leq G &\leq G + 4D(F(\mathbf{w}) - \hat{f}) \\ &= 4D\left[F(\mathbf{w}) - \left(\hat{f} - \frac{G}{4D}\right)\right] \\ &= 4D[F(\mathbf{w}) - \hat{f}_{new}]. \end{aligned} \tag{8}$$

We notice that $\hat{f}$ is a lower bound of f, so $\hat{f}_{new}$ is also a (new) lower bound of f since $\hat{f}_{new} = \hat{f} - \frac{G}{4D} \leq \hat{f}$. We observe that the bounded gradient assumption implies the D-slow condition in (8) but not vice versa since F can be unbounded from above. Therefore, the D-slow condition is weaker than the bounded gradient assumption.

2.2. Non-convex Convergence Analysis

We explain our fundamental results for the non-convex case based on the D-slow condition and diminishing learning rate scheme.

Lemma 1 (Non-convex)**.** *Suppose that F is L-smooth and the D-slow condition in (7) holds. Consider SGD (Algorithm 1) with initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha LD}}$ for some $\alpha > 0$, and diminishing learning rate*

$$\eta_{t+1} = \frac{2\eta_t}{1 + \sqrt{1 + 4\alpha LD\eta_t^2}} \tag{9}$$

for $t \geq 0$ (note that $\eta_{t+1} < \eta_t$). Then, the average

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2]$$

is dominated by

$$\frac{\left(\frac{1}{\eta_0} - \alpha\eta_0 LD\right)[F(\mathbf{w}_0) - \hat{f}]}{t+1} + \frac{(2+\alpha)LD}{t+1}\sum_{j=0}^{t}\eta_j\mathbb{E}[F(\mathbf{w}_j) - \hat{f}].$$

The convergence rate can now be described in the following theorem.

Theorem 1 (Non-convex)**.** *Suppose that F is L-smooth and the D-slow condition in (7) holds. Consider SGD (Algorithm 1) with initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha'}}$ for some $\alpha' > (1 + \sqrt{5})LD$, and diminishing learning rate given by (9), where $\alpha' = \alpha LD$. Then, there exists a number $N > 0$ such that the average*

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2]$$

is dominated by

$$\frac{\left(\frac{1}{\eta_0} - \alpha'\eta_0\right)[F(\mathbf{w}_0) - \hat{f}]}{t+1} + \frac{(2LD + \alpha')N}{t+1} = \mathcal{O}\left(\frac{1}{t}\right).$$

The standard convergence result of SGD in the non-convex case with fixed learning rate is $\mathcal{O}\left(1/\sqrt{t}\right)$ and requires a *bounded variance* (or *bounded gradient*) assumption, that is, there exists a $G > 0$ such that, $\mathbb{E}[\|\nabla f(\mathbf{w}; x, y) - \nabla F(\mathbf{w})\|^2] \leq G$ (or $\mathbb{E}[\|\nabla f(\mathbf{w}; x, y)\|^2] \leq G$), for $\forall \mathbf{w} \in \mathbb{R}^d$. We are able to achieve the convergence rate of $\mathcal{O}\left(1/t\right)$ for SGD with the weaker D-slow condition in (7) for our novel diminishing learning rate scheme. In Section 3 we will show that this condition holds for feed-forward neural networks applications.

We notice that the learning rates and the constant N depend on α' and η_0. It is nontrivial to find the values of α' and η_0 that minimize the bound in Theorem 1. Therefore, tuning α' and η_0 in order to get a better bound will be necessary in practice.

We now have the following complexity result.

Corollary 1 (Non-convex)**.** *Consider the same conditions in Theorem 1. Then, in order to achieve an ϵ-accurate solution*

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \epsilon,$$

we need $\mathcal{O}\left(1/\epsilon\right)$ iterations.

2.3. General Convex Convergence Analysis

As shown in the previous section, the D-slow condition together with the diminishing learning rate scheme in (9) help SGD achieve an $\mathcal{O}(1/t)$ convergence rate. We now show that the D-slow property is implied by convexity. We start by defining convex functions.

Definition 3 (Convex)**.** *A function ϕ is convex if $\forall \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d$,*

$$\phi(\mathbf{w}) - \phi(\mathbf{w}') \geq \nabla\phi(\mathbf{w}')^T(\mathbf{w} - \mathbf{w}'). \tag{10}$$

If f is L-smooth and convex for every realization of (x, y), then by (Nesterov, 2004), we have for $\forall \mathbf{w} \in \mathbb{R}^d$,

$$\begin{aligned}\|\nabla f(\mathbf{w}; x, y)\|^2 &\leq 2L[f(\mathbf{w}; x, y) - f(\mathbf{w}_*; x, y)] \\ &\leq 2L[f(\mathbf{w}; x, y) - \hat{f}],\end{aligned}$$

since f has lower bound $\hat{f}$ for every (x, y); where $f(\mathbf{w}_*; x, y)$ is the optimal value of $f(\mathbf{w}; x, y)$. This simply implies that f is D-slow over $\hat{f}$ for every (x, y) with $D = \frac{L}{2}$. Hence, the D-slow condition in (7) is satisfied as well.

We note that smoothness and convexity of non-negative f (i.e. $\hat{f} = 0$) for every realization of (x, y) is satisfied in many ML applications such as (but not limited to) logistic regression, least squares regression, etc.

All results from the previous section easily follow by substituting $D = \frac{L}{2}$ in Lemma 1 and Theorem 1:

Theorem 2 (General Convex)**.** *Suppose that f is L-smooth, convex and has lower bound $\hat{f}$ for every realization of (x, y). Consider SGD (Algorithm 1) with initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha'}}$ for some $\alpha' > (1+\sqrt{5})L^2/2$, and diminishing learning rate given by (9), where $\alpha' = \alpha LD = \alpha L^2/2$. Then, exactly as in Theorem 1 the average*

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] = \mathcal{O}\left(\frac{1}{t}\right).$$

From this we obtain the complexity of SGD.

Corollary 2 (General Convex)**.** *Consider the same conditions in Theorem 2. Then, in order to achieve an ϵ-accurate solution*

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \epsilon,$$

we need $\mathcal{O}(1/\epsilon)$ iterations.

2.4. Asymptotic Behavior of the DTN Scheme

Notice that the DTN recurrence

$$\eta_{t+1} = \frac{2\eta_t}{1+\sqrt{1+4\alpha'\eta_t^2}}, \tag{11}$$

with $\alpha' > 0$, after rescaling with $\bar{\eta}_t = \sqrt{\alpha'}\eta_t$ gives

$$\bar{\eta}_{t+1} = \frac{2\bar{\eta}_t}{1+\sqrt{1+4\bar{\eta}_t^2}}.$$

For this reason, without loss of generality, we only need to analyze the recursion for η_t with $\alpha' = 1$. Its asymptotic behavior is given by the following lemma.

Lemma 2. *Let*

$$c_t = \frac{1}{\eta_t}, \tag{12}$$

where η_t is defined in (11) with $\alpha' = 1$. For

$$a \leq \frac{2c_0}{\sqrt{(c_0+c_0^{-1})^2+4}} + \frac{1}{(c_0+c_0^{-1})^2+4}, \tag{13}$$

we have for all $t \geq 0$,

$$\sqrt{at+c_0^2} \leq c(t) \leq \sqrt{\frac{4}{a}t + (c_0+c_0^{-1})^2}, \tag{14}$$

where $c(t)$ is an increasing function with $c_t = c(t)$. We notice that the right hand side in (13) is ≤ 2 for all $c_0 > 0$.

We numerically verify the result by considering a simple simulation with $\eta_0 = 1$ and η_t updated by (11) with $\alpha' = 1$. Figure 1 shows the behavior of η_t and its bounds (i.e. the inverse of the bounds in (14)) with a chosen as in (13). We notice that $c_0 = 1/\eta_0$. The bounds clearly match our analysis.

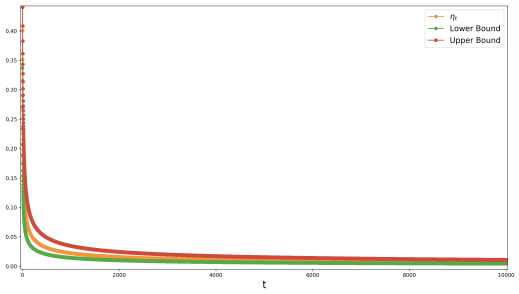

Figure 1: Asymptotic behavior of η_t and its bounds

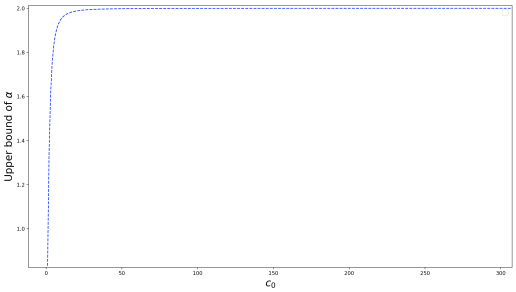

Figure 2: The upper bound on a with respect to c_0

We observe that for sufficiently large c_0, we can choose $a \approx 2$ in (13). This is shown in Figure 2 (with the upper bound of a defined in (13) for the y-axis, and the values of c_0 for the x-axis). We conclude the following result.

Corollary 3. *For DTN recurrence (11),*

$$\eta_t = \frac{1}{c(t)} = \Theta(1/\sqrt{t}) \tag{15}$$

which is well approximated by $1/\sqrt{2t+c_0^2}$ for large c_0, i.e., small η_0.

Notice that the derivation holds for any starting point c_0 and since c_T is increasing, c_T is large for large T and we may conclude that for $t \geq T$,

$$\eta_t \approx \frac{1}{\sqrt{2(t-T)+c_T^2}}.$$

3. Deep Learning Theory

In this section we provide theory for deep learning applications. We show that the softmax cross entropy loss function for classification problems is D-slow. The D-slow condition (equation (7)) leads to a proof of the fast convergence of SGD as shown in Section 2.2.

3.1. Feed Forward Neural Network

A Feed Forward Network (FFN) having n layers uses n linear transformations, for $i \in \{0, 1, \ldots, n-1\}$,

$$L^i \ : \ x \in \mathbb{R}^{d_i} \longrightarrow xW^i + b^i \in \mathbb{R}^{d_{i+1}},$$

together with n nonlinear functions

$$z \in \mathbb{R}^{d_{i+1}} \longrightarrow \sigma^i(z) \in \mathbb{R}^{d_{i+1}} \text{ for } i \in \{0, 1, \ldots, n-1\}.$$

This notation provides a general description. In practical examples the nonlinear functions $\sigma^i(z)$ calculate $(\sigma(z_1), \ldots, \sigma(z_{d_{i+1}}))$ where $\sigma(.)$ is called an activation function over the real numbers; the only exception may be for the "last layer" where $\sigma^{n-1}(z)$ can be the softmax function which outputs a vector that represents a probability vector[1].

We introduce an operator $[a, e]$ as follows: For $x \in \mathbb{R}^{d_a}$,

$$x^{[a,a]} = x,$$

i.e., $[a, a]$ is the identity operator which maps a vector x to itself. By using induction in e we define

$$x^{[a,e+1]} = \sigma^e(L^e(x^{[a,e]})).$$

The FFN outputs $x^{[0,n]}$ for input data $x \in \mathbb{R}^{d_0}$.

Operator $[a, e]$ describes the transformation by the layers in the FFN that are responsible for transformations $\sigma^i \circ L^i$ for $i \in \{a, \ldots, e-1\}$. We define

$$w^{[a,e]} = (W^a, b^a, \ldots, W^{e-1}, b^{e-1})$$

which completely describe transformations $\sigma^i \circ L^i$ for $i \in \{a, \ldots, e-1\}$. Notice that $[e, n] \circ [a, e] = [a, n]$, or equivalently

$$x^{[a,n]} = (x^{[a,e]})^{[e,n]}.$$

The goal of training the FFN with respect to training data[2]

$$(x, y) \in \mathbb{R}^{d_0} \times \{1, \ldots, d_e\}$$

is to find a vector $w_*^{[0,n]}$ which minimizes the objective function

$$F(w^{[0,n]}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(x^{[0,n]}, y)],$$

where $x^{[0,n]}$ is defined with respect to $w^{[0,n]}$, (x, y) is training data chosen from some distribution $\mathcal{D}$, and $\mathcal{L}(.,.)$ is a loss function. By realizing that $x^{[0,n]}$ is uniquely defined by x together with $w^{[0,n]}$, we express the dependence of $\mathcal{L}(x^{[0,n]}, y)$ on $w^{[0,n]}$ explicitly by the (component) function

$$f(w^{[0,n]}; x, y) = \mathcal{L}(x^{[0,n]}, y).$$

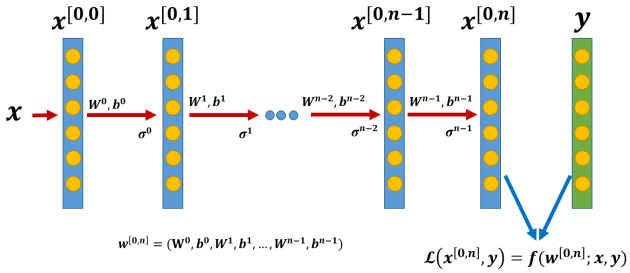

Figure 3: Diagram of a Feed Forward Neural Network

Figure 3 depicts a diagram of a FFN using our notation.

In practice we have

$$\sigma^a(z) = (\sigma(z_1), \ldots, \sigma(z_{d_{i+1}})) \text{ for } 0 \le a \le n-2,$$

where $\sigma(.)$ is an (activation) function over the real numbers. The only exception is for the "last layer" where

$$\sigma^{n-1}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

is given by the softmax function which outputs a vector that represents a probability vector. Loss function $\mathcal{L}$ will represent the cross entropy between the estimated probability vector $x^{[0,n]}$ (after applying softmax) and the "true" probability vector $\mathbf{y}$ defined as

$$\mathbf{y}_j = \delta_{j,y},$$

which says that with probability 1 data x is labelled with y:

$$\mathcal{L}(x^{[0,n]}, y) = -\sum_j \mathbf{y}_j \ln x_j^{[0,n]} = -\ln x_y^{[0,n]}.$$

We remind the reader that $f(w^{[0,n]}; x, y) = \mathcal{L}(x^{[0,n]}, y)$ and $w^{[0,n]} = (W^0, b^0, \ldots, W^{n-1}, b^{n-1})$. We have the following result.

Lemma 3. *Let $\sigma^a(z) = (\sigma(z_1), \ldots, \sigma(z_{d_{i+1}}))$ for $0 \le a \le n-2$ with $|\sigma'(z)| \le \rho$ for all z where $\sigma(.)$ is an activation function over the real numbers. Assume $\sigma^{n-1}(z)$ is the softmax function and assume loss function $\mathcal{L}$ measures cross entropy. Then, we have*

$$\begin{aligned} &\|\nabla f(w^{[0,n]}; x, y)\|^2 \\ &\le \sum_{a=0}^{n-1}(1 + \|x^{[0,a]}\|^2) \left(\prod_{j=a+1}^{n-1} \rho \|W^j\|_F\right)^2 \\ &\quad \cdot 4 f(w^{[0,n]}; x, y), \end{aligned} \tag{16}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

[1] Note that all vectors in this section are "row" vectors

[2] We assume that training data x is labeled by one out of d_e possible classes.

Remark 1 (Bounded derivative for activation functions)**.** *The condition $|\sigma'(z)| \leq \rho = 1$, for all z, holds for well-known activation functions that appear frequently in deep learning practice including but not limited to ReLU, sigmoid, tanh, etc. Softmax cross entropy is also well-known for measuring the loss for classification problems with neural networks.*

Remark 2 (Bounded inputs)**.** *The inputs are always assumed to be bounded. In practice, people usually pre-process the inputs into the range of $[0, 1]$ or $\|x\| = 1$, etc.*

We define $\mathcal{W} = \{w : \|w\| \leq B\}$ for some $B > 0$.

Remark 3 (Bounded weights)**.** *Bounded weights $\mathbf{w}_t \in \mathcal{W}$ can easily be guaranteed in SGD by neglecting updated iterations that violate this condition. In other words, if $\mathbf{w}_{t+1} \in \mathcal{W}$ then proceed as usual and if the computed $\mathbf{w}_{t+1} \notin \mathcal{W}$, then neglect this update and try again (by randomly choosing a different training data (x, y)).*

We know that if the input x is bounded and the weights $w^{[0,n]}$ are bounded to $\mathcal{W}$, then we can also bound $\|x^{[0,a]}\|^2$ by using induction in a. Therefore, there must exist a constant $D > 0$, such that

$$\sum_{a=0}^{n-1}(1 + \|x^{[0,a]}\|^2)\left(\prod_{j=a+1}^{n-1}\rho\|W^j\|_F\right)^2 \leq D,$$

which implies the D-slow property for $\mathcal{W}$. We conclude the following remark.

Remark 4. *For classification problems with feed-forward neural network using the softmax cross-entropy loss function, function f, for every (x, y), is D-slow (Definition 2) over the lower bound $\hat{f} = 0$ (non-negativity property of cross entropy loss functions) for any set $\mathcal{W} = \{w : \|w\| \leq B\}$ with fixed $B > 0$ (where $B \neq \infty$).*

Hence, the result on the iteration complexity of Corollary 1 applies. For the total complexity measured by the number of gradient evaluations, we need to know how often SGD needs to neglect an update. This is an open problem, but intuitively it makes sense that neglecting updates will not happen much once convergence kicks in – also, in practice one usually normalizes data into a small range and chooses small weights as a starting point. In practice, we fine tune the initial learning rate η_0 and the DTN parameter α' with respect to the original SGD without neglecting updates – in simulations the effect of a large B is not noticed. This is because the bound in (16) only represents a theoretical bound. In practice D can be small. This is shown in Section 4.3 for an example of bound D with respect to actual SGD updates.

4. Numerical Experiments

We notice that in the proofs of our theoretical framework we need $\alpha' > (1 + \sqrt{5})LD$. In practice, however, we tune α' and it can be chosen rather small as shown in this section. For our numerical experiments, we consider the finite sum minimization problem (as a special case of the problem (2)) with m samples. We have conducted experiments for SGD (Algorithm 1) with the DTN learning rate scheme. In order to achieve good performance, one has to properly tune DTN parameter α' and initial learning rate η_0. Figure 4 (convex) and Figure 5 (nonconvex) show a sensitivity study of α', explained below. We experimented with 10 runs and reported the average results.

4.1. Convex - Logistic Regression

We consider logistic regression problems with

$$f_i(\mathbf{w}) = \log(1 + \exp(-y_i\langle x_i, \mathbf{w}\rangle)), \tag{17}$$

where $\{x_i, y_i\}_{i=1}^m$ is the training data. We conducted experiments to demonstrate the advantage in performance of the SGD algorithm with the DTN scheme for convex problems on popular datasets including *covtype* ($m = 406,708$ training data) and *ijcnn1* ($m = 91,701$ training data) from LIBSVM (Chang & Lin, 2011).

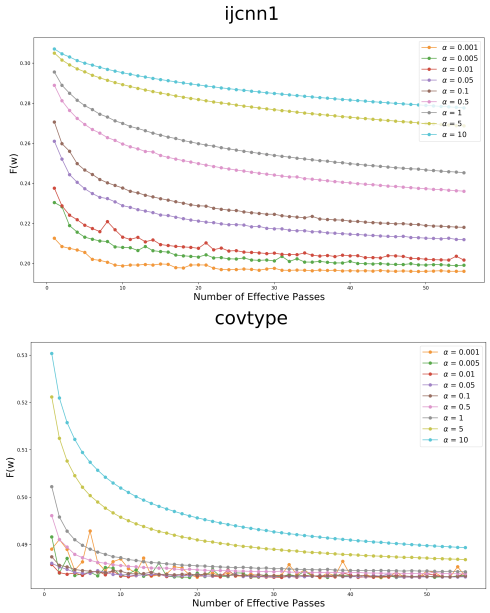

Figure 4: Comparisons of $F(\mathbf{w})$ (convex) on different values of α' on *ijcnn1* and *covtype* datasets

Figure 4 shows the comparisons of the convex objective function $F(\mathbf{w})$ for different values of $\alpha' = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$ with $\eta_0 = 1$. We depicted the value of $F(\mathbf{w})$ for the y-axis and "number of effective passes" (or number of epochs, where an epoch is the equivalent of m gradient evaluations) for the x-axis.

4.2. Non-convex - Neural Networks

We performed numerical experiments with neural networks with two fully connected hidden layers of 300 and 100 nodes, followed by a fully connected output layer which feeds into the softmax cross entropy loss. We use Tensorflow (Abadi et al., 2015) to train deep learning model with batch size 32 on MNIST ($m = 60,000$ training data) (LeCun et al., 1998) and CIFAR-10 ($m = 50,000$ training data) (Krizhevsky & Hinton, 2009), which are standard datasets for neural networks. Both datasets have 10 classes, that is, 10 softmax output nodes in the network, and are normalized to interval $[0, 1]$ as a simple data pre-processing.

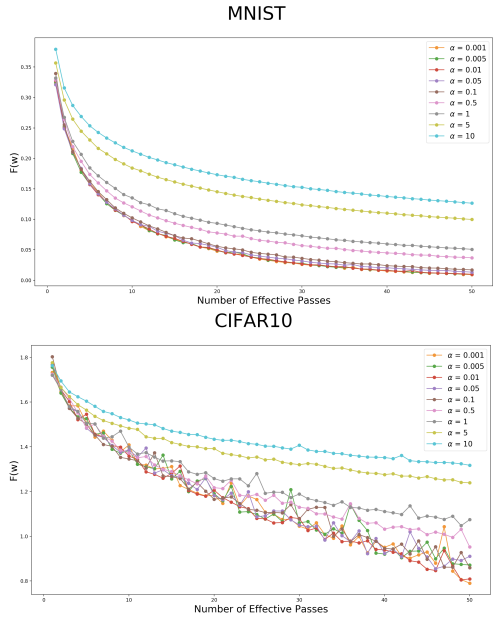

Figure 5: Comparisons of $F(\mathbf{w})$ (nonconvex) on different values of α' on *MNIST* and *CIFAR-10* datasets

Figure 5 shows comparisons of the nonconvex objective function $F(\mathbf{w})$ for different values of $\alpha' = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$ with $\eta_0 = 0.01$. We observe that the values of α' need to be chosen properly. We provide additional experiments for comparisons with other algorithms in supplementary material.

4.3. Discussion on the constant D

The D-slow condition (7) is used in our general analysis for nonconvex problems. However, it is sufficient to assume that (7) only holds for the iterations generated by Algorithm 1 rather than for $\forall \mathbf{w} \in \mathbb{R}^d$. We also note that the bound on D in Section 3 is quite loose and is only a theoretical demonstration in order to show that the D-slow property is satisfied for some D. In practice, with properly choosing α' and the initial learning rate η_0, we can achieve a small value of D. We remind the reader that the lower bound $\hat{f}$ for neural network nonconvex problems is 0. Figure 6 measures the value of $D_t = \frac{1}{m}\sum_{i=1}^{m} \|\nabla f_i(\mathbf{w}_t)\|^2/(4F(\mathbf{w}_t))$ for the y-axis where $\{w_t\}$ are generated by the SGD algorithm with DTN on the MNIST dataset.

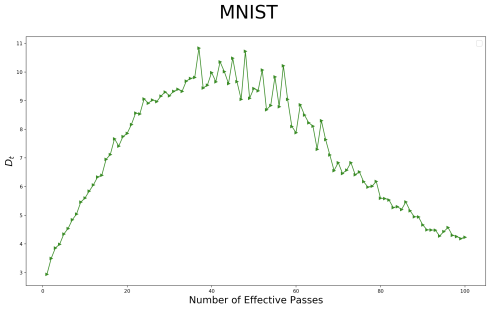

Figure 6: D_t on *MNIST* dataset

5. Conclusion and Future Work

For general convex problems and for a class of nonconvex problems which includes neural networks for classification problems we have proved

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] = \mathcal{O}\left(\frac{1}{t}\right)$$

convergence rate for SGD with $\eta_t = \Theta(1/\sqrt{t})$ defined by the DTN learning rate scheme. In our analysis we assume the existence of a lower bound for f, which widely holds true in practice (in particular, for non-negative loss functions).

We notice that $\|\nabla F(\mathbf{w}_k)\|^2 \leq 2L[F(\mathbf{w}_k) - F(\mathbf{w}_*)]$ but not vice versa for general convex. Hence, the criteria $\|\nabla F(\mathbf{w}_k)\|^2 \leq \epsilon$ is weaker than $F(\mathbf{w}_k) - F(\mathbf{w}_*) \leq \epsilon$ in this case. We leave it as an open problem to analyze $F(\mathbf{w}_k) - F(\mathbf{w}_*) \leq \epsilon$ under reasonable assumptions.

Our convergence analysis is based on the D-slow property. We have shown that this holds for smooth general convex and nonconvex neural networks with softmax cross entropy losses. It is worthwhile to investigate the D-slow property for general nonconvex problems.

Also for future work, we believe that it should be possible to extend our theory to Convolutional Neural Network or Recurrent Neural Network architectures; these are frequently used in deep learning applications. Applying the DTN learning rate scheme to other algorithms may also be useful. In the supplementary material we show numerically (without theoretical convergence analysis) the advantage of SGD with Momentum *using DTN* for training neural networks.

The convergence rate depends on the DTN parameter α' and the initial learning rate η_0. Therefore, the choice of α' and η_0 in order to achieve best performance is also an interesting question for future work.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

Allen-Zhu, Z. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. *arXiv preprint arXiv:1702.00763*, 2017.

Bertsekas, D. P. Incremental gradient, subgradient, and proximal methods for convex optimization: A survey, 2015.

Blatt, D., Hero, A. O., and Gauchman, H. A convergent incremental gradient method with a constant step size. *SIAM Journal on Optimization*, 18(1):29–51, February 2007. ISSN 1052-6234.

Boob, D. and Lan, G. Theoretical properties of the global optimizer of two layer neural network. *arXiv preprint arXiv:1710.11241*, 2017.

Bottou, L., Curtis, F. E., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

Chang, C.-C. and Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

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

Defazio, A., Bach, F., and Lacoste-Julien, S. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *NIPS*, pp. 1646–1654, 2014.

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

Fang, C., Li, C. J., Lin, Z., and Zhang, T. Spider: Near-optimal non-convex optimization via stochastic path integrated differential estimator. *arXiv preprint arXiv:1807.01695*, 2018.

Ghadimi, S. and Lan, G. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013. doi: 10.1137/120880811. URL http://dx.doi.org/10.1137/120880811.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, pp. 315–323, 2013.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Konečný, J. and Richtárik, P. Semi-stochastic gradient descent methods. *arXiv:1312.1666*, 2013.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

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

Lei, L., Ju, C., Chen, J., and Jordan, M. I. Non-convex finite-sum optimization via SCSG methods. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 2348–2358. Curran Associates, Inc., 2017.

Li, Y. and Yuan, Y. Convergence analysis of two-layer neural networks with relu activation. In *Advances in Neural Information Processing Systems*, pp. 597–607, 2017.

Mairal, J. Optimization with first-order surrogate functions. In *ICML*, pp. 783–791, 2013.

Moulines, E. and Bach, F. R. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 24*, pp. 451–459. Curran Associates, Inc., 2011.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Nemirovsky, A. S. and Yudin, D. B. Problem complexity and method efficiency in optimization. 1983.

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

Nguyen, L., Liu, J., Scheinberg, K., and Takáč, M. SARAH: A novel method for machine learning problems using stochastic recursive gradient. *ICML*, 2017a.

Nguyen, L., Nguyen, N., Phan, D., Kalagnanam, J., and Scheinberg, K. When does stochastic gradient algorithm work well? *arXiv:1801.06159*, 2018a.

Nguyen, L., Nguyen, P. H., Richtarik, P., Scheinberg, K., Takac, M., and van Dijk, M. New convergence aspects of stochastic gradient algorithms. *arXiv preprint arXiv:1811.12403*, 2018b.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takác, M. Stochastic recursive gradient algorithm for nonconvex optimization. *CoRR*, abs/1705.07261, 2017b.

Nguyen, L. M., Nguyen, P. H., van Dijk, M., Richtarik, P., Scheinberg, K., and Takac, M. SGD and Hogwild! convergence without the bounded gradients assumption. *ICML*, 2018c.

Qian, N. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

Reddi, S. J., Hefny, A., Sra, S., Póczos, B., and Smola, A. J. Stochastic variance reduction for nonconvex optimization. In *ICML*, pp. 314–323, 2016.

Robbins, H. and Monro, S. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3): 400–407, 1951.

Rudin, W. *Principles of mathematical analysis*. McGraw-Hill Book Co., New York, third edition, 1976. ISBN 0-07-085613-3. International Series in Pure and Applied Mathematics.

Schmidt, M. and Roux, N. L. Fast convergence of stochastic gradient descent under a strong growth condition. *arXiv preprint arXiv:1308.6370*, 2013.

Schmidt, M., Le Roux, N., and Bach, F. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, pp. 1–30, 2016.

Shalev-Shwartz, S. and Zhang, T. Stochastic dual coordinate ascent methods for regularized loss. *Journal of Machine Learning Research*, 14(1):567–599, 2013.

Tieleman, T. and Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. Technical report, 2012.

van Dijk, M., Nguyen, L. M., Nguyen, P. H., and Phan, D. Characterization of convex objective functions and optimal expected convergence rates for sgd. *arXiv preprint arXiv:1810.04100*, 2018.

Vaswani, S., Bach, F., and Schmidt, M. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. *arXiv preprint arXiv:1810.07288*, 2018.

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

Zou, D., Cao, Y., Zhou, D., and Gu, Q. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888*, 2018.

DTN: A Learning Rate Scheme with Convergence Rate of $\mathcal{O}(1/t)$ for SGD Supplementary Material

A. Proof of Lemma 1; Proof of Theorem 1; Proof of Lemma 2

A.1. Proof of Lemma 1

Lemma 1. *Suppose that F is L-smooth and the D-slow condition in (7) holds. Consider SGD (Algorithm 1) with initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha LD}}$ for some $\alpha > 0$, and diminishing learning rate*

$$\eta_{t+1} = \frac{2\eta_t}{1 + \sqrt{1 + 4\alpha LD\eta_t^2}}$$

for $t \geq 0$ (note that $\eta_{t+1} < \eta_t$). Then, the average

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \frac{\left(\frac{1}{\eta_0} - \alpha\eta_0 LD\right)[F(\mathbf{w}_0) - \hat{f}]}{t+1} + \frac{(2+\alpha)LD}{t+1}\sum_{j=0}^{t}\eta_j\mathbb{E}[F(\mathbf{w}_j) - \hat{f}].$$

Proof. Let $\mathcal{F}_t = \sigma(\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_t)$ be the σ-algebra generated by $\mathbf{w}_0$, $\mathbf{w}_1$, $\ldots$, $\mathbf{w}_t$. From L-smooth property of F and $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t\nabla f(\mathbf{w}_t; x_t, y_t)$, we have

$$\mathbb{E}[F(\mathbf{w}_{t+1})|\mathcal{F}_t] \overset{(5)}{\leq} F(\mathbf{w}_t) - \eta_t\|\nabla F(\mathbf{w}_t)\|^2 + \frac{\eta_t^2 L}{2}\mathbb{E}[\|\nabla f(\mathbf{w}_t; x_t, y_t)\|^2|\mathcal{F}_t]$$

Taking the expectations, we have

$$\begin{aligned}\mathbb{E}[F(\mathbf{w}_{t+1})] &\leq \mathbb{E}[F(\mathbf{w}_t)] - \eta_t\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2] + \frac{\eta_t^2 L}{2}\mathbb{E}[\|\nabla f(\mathbf{w}_t; x_t, y_t)\|^2] \\ &\overset{(7)}{\leq} \mathbb{E}[F(\mathbf{w}_t)] - \eta_t\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2] + 2\eta_t^2 LD\mathbb{E}[F(\mathbf{w}_t) - \hat{f}]\end{aligned}$$

Hence, adding $-\hat{f}$ to both sides, we have

$$\begin{aligned}\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}] &\leq \mathbb{E}[F(\mathbf{w}_t) - \hat{f}] - \eta_t\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2] + 2\eta_t^2 LD\mathbb{E}[F(\mathbf{w}_t) - \hat{f}] \\ &= (1 - \alpha\eta_t^2 LD)\mathbb{E}[F(\mathbf{w}_t) - \hat{f}] + (2+\alpha)\eta_t^2 LD\mathbb{E}[F(\mathbf{w}_t) - \hat{f}] - \eta_t\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2],\end{aligned} \tag{18}$$

for any $\alpha > 0$. Hence,

$$\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2] \leq \left(\frac{1}{\eta_t} - \alpha\eta_t LD\right)\mathbb{E}[F(\mathbf{w}_t) - \hat{f}] - \frac{1}{\eta_t}\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}] + (2+\alpha)\eta_t LD\mathbb{E}[F(\mathbf{w}_t) - \hat{f}]. \tag{19}$$

We would like to have

$$\frac{1}{\eta_t} = \frac{1}{\eta_{t+1}} - \alpha\eta_{t+1}LD, \tag{20}$$

which is equivalent to

$$(\alpha LD\eta_t)\eta_{t+1}^2 + \eta_{t+1} - \eta_t = 0.$$

Let us solve the above quadratic equation for η_{t+1} in term of η_t. We have

$$\Delta = 1 + 4\alpha LD\eta_t^2 > 0.$$

We want to have $\eta_{t+1} > 0$. Hence,

$$\eta_{t+1} = \frac{-1 + \sqrt{1 + 4\alpha LD\eta_t^2}}{2\alpha LD\eta_t} = \frac{2\eta_t}{(1 + \sqrt{1 + 4\alpha LD\eta_t^2})} < \eta_t.$$

Applying (20) and summing $j = 0, \ldots, t$ and taking the average to (19), we have

$$\begin{aligned}
\frac{1}{t+1}\sum_{j=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_j)\|^2] &\leq \frac{1}{t+1}\left(\frac{1}{\eta_0} - \alpha\eta_0 LD\right)\mathbb{E}[F(\mathbf{w}_0) - \hat{f}] - \frac{1}{t+1}\left(\frac{1}{\eta_{t+1}} - \alpha\eta_{t+1}LD\right)\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}] \\
&\quad + \frac{(2+\alpha)LD}{t+1}\sum_{j=0}^{t}\eta_j\mathbb{E}[F(\mathbf{w}_j) - \hat{f}] \\
&\leq \frac{1}{t+1}\left(\frac{1}{\eta_0} - \alpha\eta_0 LD\right)\mathbb{E}[F(\mathbf{w}_0) - \hat{f}] + \frac{(2+\alpha)LD}{t+1}\sum_{j=0}^{t}\eta_j\mathbb{E}[F(\mathbf{w}_j) - \hat{f}],
\end{aligned}$$

where the last inequality follows since $\eta_{t+1} < \eta_0 \leq \frac{1}{\sqrt{\alpha LD}}$ with decreasing η_t, for $t \geq 0$, and $\mathbb{E}[F(\mathbf{w}_{t+1})] \geq \hat{f}$. For any given $\mathbf{w}_0$, we achieve the desired result.

□

A.2. Proof of Theorem 1

Lemma 4. *Suppose that F is L-smooth and the D-slow condition in (7) holds. Consider SGD (Algorithm 1) with the initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha LD}}$, for any $\alpha > 1 + \sqrt{5}$, and diminishing learning rate for $t \geq 0$,*

$$\eta_{t+1} = \frac{2\eta_t}{1 + \sqrt{1 + 4\alpha LD\eta_t^2}}.$$

(Note that $\eta_{t+1} < \eta_t$.) Then, we have

$$\frac{\eta_{t+1}\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}]}{\eta_t\mathbb{E}[F(\mathbf{w}_t) - \hat{f}]} < 1. \tag{21}$$

Proof. From (18) in the proof of Lemma 1, we have

$$\begin{aligned}
\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}] &\leq \mathbb{E}[F(\mathbf{w}_t) - \hat{f}] - \eta_t\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2] + 2\eta_t^2 LD\mathbb{E}[F(\mathbf{w}_t) - \hat{f}] \\
&\leq (1 + 2\eta_t^2 LD)\mathbb{E}[F(\mathbf{w}_t) - \hat{f}],
\end{aligned} \tag{22}$$

and

$$\eta_{t+1} = \frac{2\eta_t}{(1 + \sqrt{1 + 4\alpha LD\eta_t^2})}. \tag{23}$$

We want to show that

$$\frac{2\eta_t}{(1 + \sqrt{1 + 4\alpha LD\eta_t^2})} < \frac{\eta_t}{1 + 2\eta_t^2 LD}, \tag{24}$$

which is equivalent to (note that $\eta_t > 0$)

$$\begin{aligned} 2 + 4\eta_t^2 LD &< 1 + \sqrt{1 + 4\alpha LD\eta_t^2} \\ 1 + 4\eta_t^2 LD &< \sqrt{1 + 4\alpha LD\eta_t^2} \\ 1 + 8\eta_t^2 LD + 16\eta_t^4 L^2 D^2 &< 1 + 4\alpha LD\eta_t^2 \\ 8\eta_t^2 LD + 16\eta_t^4 L^2 D^2 &< 4\alpha LD\eta_t^2 \\ 2 + 4\eta_t^2 LD &< \alpha \\ \eta_t^2 &< \frac{\alpha - 2}{4LD}. \end{aligned}$$

We have $\eta_t^2 \leq \eta_0^2 \leq \frac{1}{\alpha LD}$ since η_t is decreasing. Hence, if $\alpha > 1 + \sqrt{5}$, we have

$$\alpha^2 - 2\alpha - 4 > 0 \Leftrightarrow \frac{1}{\alpha} < \frac{\alpha - 2}{4}.$$

Hence, if $\alpha > 1 + \sqrt{5}$, we have

$$\eta_t^2 \leq \frac{1}{\alpha LD} < \frac{\alpha - 2}{4LD},$$

which implies (24), that is

$$\eta_{t+1} = \frac{2\eta_t}{(1 + \sqrt{1 + 4\alpha LD\eta_t^2})} < \frac{\eta_t}{1 + 2\eta_t^2 LD},$$

which is equivalent to

$$1 + 2\eta_t^2 LD < \frac{\eta_t}{\eta_{t+1}}.$$

Applying to (22), we have

$$\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}] < \frac{\eta_t}{\eta_{t+1}} \mathbb{E}[F(\mathbf{w}_t) - \hat{f}],$$

which is equivalent to

$$\frac{\eta_{t+1}\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f}]}{\eta_t \mathbb{E}[F(\mathbf{w}_t) - \hat{f}]} < 1.$$

This completes the proof.

□

Lemma 5. *Suppose that F is L-smooth and the D-slow condition in (7) holds. Consider SGD (Algorithm 1) with the initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha LD}}$, for any $\alpha > 1 + \sqrt{5}$, and diminishing learning rate for $t \geq 0$,*

$$\eta_{t+1} = \frac{2\eta_t}{1 + \sqrt{1 + 4\alpha LD\eta_t^2}}.$$

(Note that $\eta_{t+1} < \eta_t$.) Then, there must exist $N > 0$ such that

$$\sum_{j=0}^{t} \eta_j \mathbb{E}[F(\mathbf{w}_j) - \hat{f}] \leq N < \infty.$$

Furthermore,

$$\frac{1}{t+1} \sum_{k=0}^{t} \mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \frac{\left(\frac{1}{\eta_0} - \alpha\eta_0 LD\right)[F(\mathbf{w}_0) - \hat{f}]}{t+1} + \frac{(2+\alpha)LD}{t+1} N = \mathcal{O}\left(\frac{1}{t}\right).$$

Proof. From Lemma 1, we have

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \frac{\left(\frac{1}{\eta_0}-\alpha\eta_0 LD\right)[F(\mathbf{w}_0)-\hat{f}]}{t+1} + \frac{(2+\alpha)LD}{t+1}\sum_{j=0}^{t}\eta_j\mathbb{E}[F(\mathbf{w}_j)-\hat{f}].$$

We need to show that there exists $N > 0$ such that

$$\sum_{j=0}^{t}\eta_j\mathbb{E}[F(\mathbf{w}_j)-\hat{f}] \leq \sum_{j=0}^{\infty}\eta_j\mathbb{E}[F(\mathbf{w}_j)-\hat{f}] = N < \infty, \tag{25}$$

in order to achieve

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] = \mathcal{O}\left(\frac{1}{t}\right).$$

In order to prove (25), using the ratio test for a series (Rudin, 1976), we need to show that

$$\frac{\eta_{t+1}\mathbb{E}[F(\mathbf{w}_{t+1})-\hat{f}]}{\eta_t\mathbb{E}[F(\mathbf{w}_t)-\hat{f}]} < 1.$$

It follows by Lemma 4. Therefore, we have

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \frac{\left(\frac{1}{\eta_0}-\alpha\eta_0 LD\right)[F(\mathbf{w}_0)-\hat{f}]}{t+1} + \frac{(2+\alpha)LD}{t+1}N = \mathcal{O}\left(\frac{1}{t}\right).$$

□

For simplifying the notations in Lemma 5, we can replace by $\alpha' = \alpha LD$. Hence, with $\alpha' > (1+\sqrt{5})LD$, we could prove Theorem 1. We recall Theorem 1 as follows.

Theorem 1. *Suppose that F is L-smooth and the D-slow condition in (7) holds. Consider SGD (Algorithm 1) with initial learning rate $0 < \eta_0 \leq \frac{1}{\sqrt{\alpha'}}$ for some $\alpha' > (1+\sqrt{5})LD$, and diminishing learning rate given by (9), where $\alpha' = \alpha LD$. Then, there exists a number $N > 0$ such that the average*

$$\frac{1}{t+1}\sum_{k=0}^{t}\mathbb{E}[\|\nabla F(\mathbf{w}_k)\|^2] \leq \frac{\left(\frac{1}{\eta_0}-\alpha'\eta_0\right)[F(\mathbf{w}_0)-\hat{f}]}{t+1} + \frac{(2LD+\alpha')N}{t+1} = \mathcal{O}\left(\frac{1}{t}\right).$$

A.3. Proof of Lemma 2

Lemma 2. *Let*

$$c_t = \frac{1}{\eta_t},$$

where η_t defined in (11) with $\alpha' = 1$. For

$$a \leq \frac{2c_0}{\sqrt{(c_0+c_0^{-1})^2+4}} + \frac{1}{(c_0+c_0^{-1})^2+4}, \tag{26}$$

we have for all $t \geq 0$,

$$\sqrt{at+c_0^2} \leq c(t) \leq \sqrt{\frac{4}{a}t + (c_0+c_0^{-1})^2},$$

where $c(t)$ is an increasing function with $c_t = c(t)$. We notice that the right hand side in (26) is ≤ 2 for all $c_0 > 0$.

Proof. Notice that

$$c_{t+1} = \frac{1}{2}(c_t + \sqrt{c_t^2+4}) = c_t + \frac{\sqrt{c_t^2+4}-\sqrt{c_t^2}}{2}.$$

The Taylor series expansion of $\sqrt{x}$ around c_t^2 shows that

$$\frac{\sqrt{c_t^2+4}-\sqrt{c_t^2}}{2} = \frac{\sqrt{c_t^2}+\frac{1}{2\sqrt{z_t^2}}\cdot 4^1 - \sqrt{c_t^2}}{2} = \frac{1}{z_t},$$

where $\frac{1}{2\sqrt{z_t^2}}\cdot 4^1$ is the Lagrange remainder for some

$$c_t^2 \le z_t^2 \le c_t^2 + 4. \tag{27}$$

Substituting this into our recursive formula for c_{t+1} yields

$$c_{t+1} = c_t + \frac{1}{z_t}.$$

By using induction in t we solve

$$c_t = c_0 + \sum_{i=0}^{t-1}\frac{1}{z_i}.$$

This proves that c_t is increasing in t. Now we are ready to provide lower and upper bounds by using (27):

$$\begin{aligned} c_t &\ge c_0 + \sum_{i=0}^{t-1}\frac{1}{\sqrt{c_i^2+4}}, \text{ and} \\ c_t &\le c_0 + \sum_{i=0}^{t-1}\frac{1}{\sqrt{c_i^2}}. \end{aligned}$$

Let us extend c_t to an increasing function $c(t)$ with $c_t = c(t)$.

We use $c(t)$ to bound the sums in the lower and upper bounds of c_t:

$$\sum_{i=0}^{t-1}\frac{1}{\sqrt{c_i^2+4}} \ge \frac{1}{\sqrt{c(t-1)^2+4}} + \int_{x=0}^{t-1}\frac{dx}{\sqrt{c(x)^2+4}}$$

and

$$\sum_{i=0}^{t-1}\frac{1}{\sqrt{c_i^2}} \le \frac{1}{\sqrt{c_0^2}} + \int_{x=0}^{t-1}\frac{dx}{\sqrt{c(x)^2}}.$$

We conclude

$$\begin{aligned} &c_0 + \frac{1}{\sqrt{c(t-1)^2+4}} + \int_{x=0}^{t-1}\frac{dx}{\sqrt{c(x)^2+4}} \\ &\le c(t) \le c_0 + \frac{1}{c_0} + \int_{x=0}^{t-1}\frac{dx}{\sqrt{c(x)^2}}. \end{aligned}$$

Assume as induction hypothesis that

$$\sqrt{ax+c_0^2} \le c(x) \le \sqrt{bx + (c_0 + c_0^{-1})^2}$$

for $0 \le x \le t-1$ for some $t \ge 1$. (The base case for $t = 1$ trivially holds since $c(0) = c_0$.) Then,

$$\begin{aligned} \int_{x=0}^{t-1}\frac{dx}{\sqrt{c(x)^2}} &\le \int_{x=0}^{t-1}\frac{dx}{\sqrt{ax+c_0^2}} \\ &= 2\frac{\sqrt{a(t-1)+c_0^2}}{a} - 2\frac{c_0}{a} \end{aligned}$$

and similarly

$$\begin{aligned}\int_{x=0}^{t-1}\frac{dx}{\sqrt{c(x)^2}} &\geq \int_{x=0}^{t-1}\frac{dx}{\sqrt{bx+(c_0+c_0^{-1})^2}}\\ &= 2\frac{\sqrt{b(t-1)+(c_0+c_0^{-1})^2}}{b}-2\frac{c_0+c_0^{-1}}{b}\end{aligned}$$

We derive

$$c(t)\leq c_0+\frac{1}{c_0}+2\frac{\sqrt{a(t-1)+c_0^2}}{a}-2\frac{c_0}{a}$$

and

$$c(t) \geq c_0+\frac{1}{\sqrt{b(t-1)+(c_0+c_0^{-1})^2+4}}+2\frac{\sqrt{b(t-1)+(c_0+c_0^{-1})^2}}{b}-2\frac{c_0+c_0^{-1}}{b}.$$

We notice that

$$c_0+\frac{1}{\sqrt{b(t-1)+(c_0+c_0^{-1})^2+4}}+\sqrt{\frac{4}{b}(t-1)+\frac{4}{b^2}(c_0+c_0^{-1})^2}-2\frac{c_0+c_0^{-1}}{b}\geq\sqrt{at+c_0^2}$$

implies $c(t)\geq\sqrt{at+c_0^2}$. We notice that

$$c_0+\frac{1}{c_0}+\sqrt{\frac{4}{a}(t-1)+\frac{4}{a^2}c_0^2}-2\frac{c_0}{a}\leq\sqrt{bt+(c_0+c_0^{-1})^2}$$

implies $c(t)\leq\sqrt{bt+(c_0+c_0^{-1})^2}$. If these inequalities hold, then we have proven the induction hypothesis for $t+1$ and by induction in t we know that

$$\sqrt{at+c_0^2}\leq c(t)\leq\sqrt{bt+(c_0+c_0^{-1})^2}$$

for all $t\geq 0$. We use

$$b=4/a.$$

Then these two conditions become

$$c_0+\frac{1}{\sqrt{\frac{4}{a}(t-1)+(c_0+c_0^{-1})^2+4}}+\sqrt{a(t-1)+\frac{a^2}{4}(c_0+c_0^{-1})^2}-\frac{a}{2}(c_0+c_0^{-1})\geq\sqrt{at+c_0^2}$$

and

$$c_0+\frac{1}{c_0}+\sqrt{\frac{4}{a}(t-1)+\frac{4}{a^2}c_0^2}-2\frac{c_0}{a}\leq\sqrt{\frac{4}{a}t+(c_0+c_0^{-1})^2}.$$

The derivatives with respect to t on the left hand and right hand sides are equal for both inequalities. So, the two conditions are true if they hold for $t=1$. For $t=1$, the inequalities become

$$c_0+\frac{1}{\sqrt{(c_0+c_0^{-1})^2+4}}\geq\sqrt{a+c_0^2}$$

and

$$c_0+\frac{1}{c_0}\leq\sqrt{\frac{4}{a}+(c_0+c_0^{-1})^2}.$$

The second inequality trivially holds. The first inequality is true if

$$a \leq \left(c_0 + \frac{1}{\sqrt{(c_0 + c_0^{-1})^2 + 4}} \right)^2 - c_0^2. \tag{28}$$

Notice (by using straightforward calculus) that the right side is equal to

$$\frac{2c_0}{\sqrt{(c_0 + c_0^{-1})^2 + 4}} + \frac{1}{(c_0 + c_0^{-1})^2 + 4} \leq 2.$$

For large c_0, we can choose $a \approx 2$.

Now we use induction in t and conclude that

$$\sqrt{at + c_0^2} \leq c(t) \leq \sqrt{\frac{4}{a}t + (c_0 + c_0^{-1})^2}$$

holds for all t if a is chosen according to (28). This completes the proof. □

B. Deep Learning Theory and Proof of Lemma 3

B.1. Feed Forward Neural Network

We recall the notations that a Feed Forward Network (FFN) having n layers uses n linear transformations, for $i \in \{0, 1, \ldots, n-1\}$,

$$L^i \ : \ x \in \mathbb{R}^{d_i} \longrightarrow xW^i + b^i \in \mathbb{R}^{d_{i+1}},$$

together with n nonlinear functions

$$z \in \mathbb{R}^{d_{i+1}} \longrightarrow \sigma^i(z) \in \mathbb{R}^{d_{i+1}} \text{ for } i \in \{0, 1, \ldots, n-1\}.$$

This notation will provide a general description. In practical examples the nonlinear functions $\sigma^i(z)$ calculate $(\sigma(z_1), \ldots, \sigma(z_{d_{i+1}}))$ where $\sigma(.)$ is called an activation function over the real numbers; the only exception may be for the "last layer" where $\sigma^{n-1}(z)$ may be chosen as the softmax function which outputs a vector that represents a probability vector[3].

We introduce an operator $[a, e]$ as follows: For $x \in \mathbb{R}^{d_a}$,

$$x^{[a,a]} = x,$$

i.e., $[a, a]$ is the identity operator which maps a vector x to itself. By using induction in e we define

$$x^{[a,e+1]} = \sigma^e(L^e(x^{[a,e]})).$$

The FFN outputs $x^{[0,n]}$ for input data $x \in \mathbb{R}^{d_0}$.

Operator $[a, e]$ describes the transformation by the layers in the FFN that are responsible for transformations $\sigma^i \circ L^i$ for $i \in \{a, \ldots, e-1\}$. We define

$$w^{[a,e]} = (W^a, b^a, \ldots, W^{e-1}, b^{e-1})$$

which completely describe transformations $\sigma^i \circ L^i$ for $i \in \{a, \ldots, e-1\}$. Notice that $[e, n] \circ [a, e] = [a, n]$, or equivalently

$$x^{[a,n]} = (x^{[a,e]})^{[e,n]}.$$

The goal of training the FFN with respect to training data[4]

$$(x, y) \in \mathbb{R}^{d_0} \times \{1, \ldots, d_e\}$$

[3] Note that all vectors in this section are "row" vectors

[4] We assume that training data x is labeled by one out of d_e possible classes.

is to find a vector $w_*^{[0,n]}$ which minimizes the objective function

$$F(w^{[0,n]}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(x^{[0,n]}, y)],$$

where $x^{[0,n]}$ is defined with respect to $w^{[0,n]}$, (x, y) is training data chosen from some distribution $\mathcal{D}$, and $\mathcal{L}(.,.)$ is a loss function. By realizing that $x^{[0,n]}$ is uniquely defined by x together with $w^{[0,n]}$, we express the dependence of $\mathcal{L}(x^{[0,n]}, y)$ on $w^{[0,n]}$ explicitly by the (component) function

$$f(w^{[0,n]}; x, y) = \mathcal{L}(x^{[0,n]}, y).$$

We generalize the definition of the component function for all $[a, n]$:

$$f(w^{[a,n]}; x, y) = \mathcal{L}(x^{[a,n]}, y) \text{ where } x \in \mathbb{R}^{d_a}.$$

By using induction in $e \geq a$, we have

$$f(w^{[a,n]}; x, y) = f(w^{[e,n]}; x^{[a,e]}, y).$$

In particular,

$$\begin{aligned} f(w^{[a,n]}; x, y) &= f(w^{[a+1,n]}; x^{[a,a+1]}, y) = f(w^{[a+1,n]}; \sigma^a(L^a(x)), y) \\ &= f(w^{[a+1,n]}; \sigma^a(xW^a + b^a), y). \end{aligned} \tag{29}$$

B.2. Component Gradients

We want to compute bounds on the gradient of component function $f(w^{[a,n]}; x, y)$. Our goal is to understand under which circumstances, i.e., for which inputs, this gradient is small implying that the value of the component function is close to a minimum. This will allow us to understand the shape the minimum of objective function F achieved by $w^{*[0,n]}$.

For $0 \leq a \leq n-1$, we study the partial derivative, see (29),

$$\frac{\partial f(w^{[a,n]}; x, y)}{\partial W^a_{l,s}} = \frac{\partial f(w^{[a+1,n]}; \sigma^a(xW^a + b^a), y)}{\partial W^a_{l,s}}.$$

In the calculation below we use the following notation for functions defined by taking partial derivatives:

$$\begin{aligned} f_i(w^{[a+1,n]}; x, y) &= \frac{\partial f(w^{[a+1,n]}; x, y)}{\partial x_i}, \\ \Phi^a_{i,k}(z) &= \frac{\partial \sigma^a(z)_i}{\partial z_k}, \\ \delta_{k,s} \cdot x_l &= \frac{\partial \sum_j x_j W^a_{j,k}}{\partial W^a_{l,s}} = \frac{\partial (xW^a)_k}{\partial W^a_{l,s}}, \end{aligned}$$

where $\delta_{k,s}$ is the Kronecker delta function with value 1 if indices s and k are equal and with value 0 if $s \neq k$.

Lemma 6. *For $0 \leq a \leq n-1$, let*

$$S^a_s(z, y) = \sum_i f_i(w^{[a+1,n]}; \sigma^a(z), y)\Phi^a_{i,s}(z).$$

Then,

$$\sum_{l,s} \left(\frac{\partial f(w^{[0,n]}; x, y)}{\partial W^a_{l,s}} \right)^2 = \left\{ \sum_s S^a_s(L^a(x^{[0,a]}), y)^2 \right\} \cdot \|x^{[0,a]}\|^2$$

and

$$\sum_s \left(\frac{\partial f(w^{[0,n]}; x, y)}{\partial b^a_s} \right)^2 = \left\{ \sum_s S^a_s(L^a(x^{[0,a]}), y)^2 \right\}.$$

Proof. Let $0 \leq a \leq n-1$. By using the chain rule we infer

$$\begin{aligned}
&\frac{\partial f(w^{[a+1,n]}; \sigma^a(xW^a + b^a), y)}{\partial W^a_{l,s}} \\
= &\sum_{k,i} f_i(w^{[a+1,n]}; \sigma^a(xW^a + b^a), y) \Phi^a_{i,k}(xW^a + b^a) \delta_{k,s} x_l \\
= &\sum_{i} f_i(w^{[a+1,n]}; \sigma^a(xW^a + b^a), y) \Phi^a_{i,s}(xW^a + b^a) x_l \\
= &S^a_s(xW^a + b^a, y) x_l,
\end{aligned}$$

for

$$S^a_s(z, y) = \sum_i f_i(w^{[a+1,n]}; \sigma^a(z), y) \Phi^a_{i,s}(z).$$

In a similar way we can derive

$$\frac{\partial f(w^{[a+1,n]}; \sigma^a(xW^a + b^a), y)}{\partial b^a_s} = S^a_s(xW^a + b^a, y).$$

The above derivation proves

$$\begin{aligned}
\sum_{l,s} \left(\frac{\partial f(w^{[0,n]}; x, y)}{\partial W^a_{l,s}} \right)^2 &= \sum_{l,s} \left(\frac{\partial f(w^{[a,n]}; x^{[0,a]}, y)}{\partial W^a_{l,s}} \right)^2 \\
&= \left\{ \sum_s S^a_s(x^{[0,a]} W^a + b^a, y)^2 \right\} \cdot \left\{ \sum_l (x^{[0,a]}_l)^2 \right\} \\
&= \left\{ \sum_s S^a_s(L^a(x^{[0,a]}), y)^2 \right\} \cdot \|x^{[0,a]}\|^2 \qquad (30)
\end{aligned}$$

and

$$\sum_s \left(\frac{\partial f(w^{[0,n]}; x, y)}{\partial b^a_s} \right)^2 = \left\{ \sum_s S^a_s(L^a(x^{[0,a]}), y)^2 \right\}.$$

□

B.3. Bounding the Norm of S^a_s

Lemma 7. *Let*

$$\gamma^a(x, y) = \sum_s S^a_s(x^{[0,a]} W^a + b^a, y)^2.$$

Then, $\gamma^{n-1}(x, y)$ *is equal to*

$$\sum_k \left(\sum_i \Phi^{n-1}_{i,k}(x^{[0,n-1]} W^{n-1} + b^{n-1}) \cdot \tfrac{\partial \mathcal{L}(x,y)}{\partial x_i}|_{x=x^{[0,n]}} \right)^2$$

and, for $0 \leq a \leq n-2$,

$$\begin{aligned}
\gamma^a(x, y) \leq &\gamma^{a+1}(x, y) \\
&\cdot \left\{ \sum_{s,k} \left(\sum_i W^{a+1}_{i,k} \Phi^a_{i,s}(x^{[0,a]} W^a + b^a) \right)^2 \right\}.
\end{aligned}$$

Proof. In order to find lower and upper bounds for (30) we analyze $S_s^a(L^a(x^{[0,a]}), y)$ for $0 \leq a \leq n-1$. First,

$$S_s^{n-1}(z,y) = \sum_i f_i(w^{[n,n]}; \sigma^{n-1}(z), y)\Phi_{i,s}^{n-1}(z) = \sum_i \Phi_{i,s}^{n-1}(z) \cdot \frac{\partial \mathcal{L}(x,y)}{\partial x_i}|_{x=\sigma^{n-1}(z)}.$$

Second,

$$\begin{aligned}
& S_s^a(z,y) \\
= & \sum_i f_i(w^{[a+1,n]}; \sigma^a(z), y)\Phi_{i,s}^a(z) \\
= & \sum_i \frac{\partial f(w^{[a+1,n]}; x, y)}{\partial x_i}|_{x=\sigma^a(z)} \cdot \Phi_{i,s}^a(z) \\
= & \sum_i \frac{\partial f(w^{[a+2,n]}; \sigma^{a+1}(xW^{a+1}+b^{a+1}), y)}{\partial x_i}|_{x=\sigma^a(z)} \cdot \Phi_{i,s}^a(z) \\
= & \sum_{i,j} f_j(w^{[a+2,n]}; \sigma^{a+1}(xW^{a+1}+b^{a+1}), y)\frac{\partial \sigma^{a+1}(xW^{a+1}+b^{a+1})_j}{\partial x_i}|_{x=\sigma^a(z)} \cdot \Phi_{i,s}^a(z),
\end{aligned}$$

where the last equality follows from the chain rule. From the chain rule we also infer

$$\begin{aligned}
\frac{\partial \sigma^{a+1}(xW^{a+1}+b^{a+1})_j}{\partial x_i} &= \sum_k \frac{\partial \sigma^{a+1}(v)_j}{\partial v_k}|_{v=xW^{a+1}+b^{a+1}} \cdot \frac{\partial (xW^{a+1}+b^{a+1})_k}{\partial x_i} \\
&= \sum_k \Phi_{j,k}^{a+1}(xW^{a+1}+b^{a+1})W_{i,k}^{a+1}.
\end{aligned}$$

Substituting this into the previous equation yields

$$\begin{aligned}
S_s^a(z,y) &= \sum_{i,j,k} f_j(w^{[a+2,n]}; \sigma^{a+1}(\sigma^a(z)W^{a+1}+b^{a+1}), y)\Phi_{j,k}^{a+1}(\sigma^a(z)W^{a+1}+b^{a+1}) \cdot W_{i,k}^{a+1}\Phi_{i,s}^a(z) \\
&= \sum_{i,k} \left\{ \sum_j f_j(w^{[a+2,n]}; \sigma^{a+1}(\sigma^a(z)W^{a+1}+b^{a+1}), y)\Phi_{j,k}^{a+1}(\sigma^a(z)W^{a+1}+b^{a+1}) \right\} \cdot W_{i,k}^{a+1}\Phi_{i,s}^a(z) \\
&= \sum_{i,k} S_k^{a+1}(\sigma^a(z)W^{a+1}+b^{a+1}, y)W_{i,k}^{a+1}\Phi_{i,s}^a(z) \\
&= \sum_k S_k^{a+1}(\sigma^a(z)W^{a+1}+b^{a+1}, y) \cdot \left\{ \sum_i W_{i,k}^{a+1}\Phi_{i,s}^a(z) \right\}.
\end{aligned}$$

We substitute

$$z = x^{[0,a]}W^a + b^a.$$

For this z we have

$$\sigma^a(z) = x^{[0,a+1]}.$$

By realizing that the sum over k expresses an inner product, we can apply Cauchy-Schwartz and obtain inequality

$$\begin{aligned}
& |S_s^a(x^{[0,a]}W^a + b^a, y)| \\
\leq & \sqrt{\sum_k S_k^{a+1}(x^{[0,a+1]}W^{a+1}+b^{a+1}, y)^2} \cdot \sqrt{\sum_k \left(\sum_i W_{i,k}^{a+1}\Phi_{i,s}^a(x^{[0,a]}W^a + b^a) \right)^2}.
\end{aligned}$$

Squaring and summing over s proves the lemma. □

Substituting this upper bound in (30) gives

$$\sum_{l,s}\left(\frac{\partial f(w^{[0,n]};x,y)}{\partial W^a_{l,s}}\right)^2 \leq \gamma^a(x,y)\cdot\|x^{[0,a]}\|^2,$$

and

$$\sum_{s}\left(\frac{\partial f(w^{[0,n]};x,y)}{\partial b^a_{s}}\right)^2 \leq \gamma^a(x,y),$$

with equalities for $a=n-1$.

B.4. Activation Function

In practice we have

$$\sigma^a(z)=(\sigma(z_1),\ldots,\sigma(z_{d_{i+1}})) \text{ for } 0\leq a\leq n-2,$$

where $\sigma(.)$ is an (activation) function over the real numbers. The only exception is for the "last layer" where

$$\sigma^{n-1}(z)_i=\frac{e^{z_i}}{\sum_j e^{z_j}}$$

is given by the softmax function which outputs a vector that represents a probability vector. Loss function $\mathcal{L}$ will represent the cross entropy between the estimated probability vector $x^{[0,n]}$ (after applying softmax) and the "true" probability vector $\mathbf{y}$ defined as

$$\mathbf{y}_j=\delta_{j,y},$$

which says that with probability 1 data x is labelled with y:

$$\mathcal{L}(x^{[0,n]},y)=-\sum_j \mathbf{y}_j\ln x^{[0,n]}_j=-\ln x^{[0,n]}_y.$$

We obtain the following lemma.

Lemma 8. *Let $\sigma^a(z)=(\sigma(z_1),\ldots,\sigma(z_{d_{i+1}}))$ for $0\leq a\leq n-2$ with $|\sigma'(z)|\leq\rho$ for all z where $\sigma(.)$ is an activation function over the real numbers. Assume $\sigma^{n-1}(z)$ is the softmax function and assume loss function $\mathcal{L}$ measures cross entropy. Then, for $0\leq a\leq n-1$,*

$$\gamma^a(x,y)\leq\left(\prod_{j=a+1}^{n-1}\rho\|W^j\|_F\right)^2\cdot\|x^{[0,n]}-\mathbf{y}\|^2$$

with equality for $a=n-1$.

Proof. Since

$$\Phi^{n-1}_{i,k}(z)=\frac{\sigma^{n-1}(z)_i}{z_k}=\delta_{i,k}\frac{e^{z_i}}{\sum_j e^{z_j}}-\frac{e^{z_i}}{(\sum_j e^{z_j})^2}e^{z_k}=\sigma^{n-1}(z)_i(\delta_{i,k}-\sigma^{n-1}(z)_k)$$

and

$$\frac{\partial\mathcal{L}(x,y)}{\partial x_i}|_{x=x^{[0,n]}}=-\frac{1}{x^{[0,n]}_y}\delta_{i,y},$$

we have

$$\begin{aligned}\gamma^{n-1}(x,y) &= \sum_k\left(\sum_i\Phi^{n-1}_{i,k}(x^{[0,n-1]}W^{n-1}+b^{n-1})\cdot\frac{\partial\mathcal{L}(x,y)}{\partial x_i}|_{x=x^{[0,n]}}\right)^2\\ &= \sum_k\left(\Phi^{n-1}_{y,k}(x^{[0,n-1]}W^{n-1}+b^{n-1})\cdot\frac{-1}{x^{[0,n]}_y}\right)^2\\ &= \sum_k(\delta_{y,k}-x^{[0,n]}_k)^2=\|x^{[0,n]}-\mathbf{y}\|^2.\end{aligned}$$

Since for $0 \leq a \leq n-2$

$$\Phi^a_{i,s}(z) = \frac{\sigma^a(z)_i}{z_s} = \sigma'(z_i)\delta_{i,s},$$

we can also simplify

$$\begin{aligned}\sum_{s,k}\left(\sum_i W^{a+1}_{i,k}\Phi^a_{i,s}(x^{[0,a]}W^a + b^a)\right)^2 &= \sum_{s,k}\left(W^{a+1}_{s,k}\sigma'((x^{[0,a]}W^a + b^a)_s)\right)^2 \\ &= \sum_s\left\{\sum_k (W^{a+1}_{s,k})^2\right\}\cdot\sigma'((L^a(x^{[0,a]})_s)^2.\end{aligned}$$

If we assume

$$\sigma'(z) \leq \rho \text{ for all } z,$$

then the above sum is at most

$$\rho^2\sum_{s,k}(W^{a+1}_{s,k})^2 = \rho^2\|W^{a+1}\|_F^2,$$

where $\|.\|_F$ is the Frobenius norm. □

By combining Lemmas 6, 7, and 8, we get a lower and upper bound on the gradient of the component function with respect to $w^{[0,n]}$:

$$\begin{aligned}(1+\|x^{[0,n-1]}\|^2)\cdot\|x^{[0,n]} - \mathbf{y}\|^2 &\leq \|\nabla f(w^{[0,n]}; x, y)\|^2 \\ &\leq \sum_{a=0}^{n-1}(1+\|x^{[0,a]}\|^2)\left(\prod_{j=a+1}^{n-1}\rho\|W^j\|_F\right)^2\cdot\|x^{[0,n]} - \mathbf{y}\|^2. \qquad (31)\end{aligned}$$

B.5. On the Equivalence of Curve Fitting and Learning

For the loss function we may derive an upper bound and lower bound in terms of $\|x^{[0,n]} - \mathbf{y}\|$:

Lemma 9. *We have*

$$\mathcal{L}(x^{[0,n]}, y) \geq \|x^{[0,n]} - \mathbf{y}\|/(2\sqrt{2}).$$

Proof. We know $x^{[0,n]}$ represents a probability distribution (after the softmax layer) and therefore if $c = 1 - x^{[0,n]}_y$, then $\|x^{[0,n]} - \mathbf{y}\|^2 \leq c^2 + c^2$ (since the sum of the remaining probabilities $x^{[0,n]}_i$, $i \neq y$, is equal to c). This implies

$$\|x^{[0,n]} - \mathbf{y}\|/\sqrt{2} \leq 1 - x^{[0,n]}_y.$$

This allows us to derive the lower bound

$$\begin{aligned}\mathcal{L}(x^{[0,n]}, y) &= -\ln x^{[0,n]}_y = -\ln(1-(1-x^{[0,n]}_y)) = \ln(1/(1-(1-x^{[0,n]}_y))) \\ &\geq \ln(1/(1-\|x^{[0,n]} - \mathbf{y}\|/\sqrt{2})) = \ln(1+\|x^{[0,n]} - \mathbf{y}\|/(\sqrt{2}-\|x^{[0,n]} - \mathbf{y}\|)) \\ &\geq \ln(1+\|x^{[0,n]} - \mathbf{y}\|/\sqrt{2}).\end{aligned}$$

Since $x^{[0,n]}$ and $\mathbf{y}$ are probability distributions $\|x^{[0,n]} - \mathbf{y}\| \leq \sqrt{2}$, in other words, $z = \|x^{[0,n]} - \mathbf{y}\|/\sqrt{2} \leq 1$. Because $\ln(1+z) \geq z/2$ for $0 \leq z \leq 1$, we conclude

$$\mathcal{L}(x^{[0,n]}, y) \geq \|x^{[0,n]} - \mathbf{y}\|/(2\sqrt{2}).$$

□

The lemma shows that minimizing the loss function is equivalent to minimizing $\|x^{[0,n]} - \mathbf{y}\|$, i.e., learning from training data is equivalent to curve fitting the true distribution $\mathbf{y}$ for training data x.

Notice that $x^{[0,n]}$ and $\mathbf{y}$ represent probability vectors, therefore, $\|x^{[0,n]} - \mathbf{y}\| \leq \sqrt{2}$. Hence, together with Lemma 9 we have

$$\|x^{[0,n]} - \mathbf{y}\|^2 \leq \sqrt{2}\|x^{[0,n]} - \mathbf{y}\| \leq 4\mathcal{L}(x^{[0,n]}, y). \tag{32}$$

We remind the reader that $f(w^{[0,n]}; x, y) = \mathcal{L}(x^{[0,n]}, y)$. Hence, applying (32) to (31), we have

$$\|\nabla f(w^{[0,n]}; x, y)\|^2 \leq \sum_{a=0}^{n-1}(1 + \|x^{[0,a]}\|^2)\left(\prod_{j=a+1}^{n-1}\rho\|W^j\|_F\right)^2 \cdot 4f(w^{[0,n]}; x, y).$$

Therefore, we have proved Lemma 3 in Section 3.

Lemma 3. *Let $\sigma^a(z) = (\sigma(z_1), \ldots, \sigma(z_{d_{i+1}}))$ for $0 \leq a \leq n-2$ with $|\sigma'(z)| \leq \rho$ for all z where $\sigma(.)$ is an activation function over the real numbers. Assume $\sigma^{n-1}(z)$ is the softmax function and assume loss function $\mathcal{L}$ measures cross entropy. Then, we have*

$$\|\nabla f(w^{[0,n]}; x, y)\|^2 \leq \sum_{a=0}^{n-1}(1 + \|x^{[0,a]}\|^2)\left(\prod_{j=a+1}^{n-1}\rho\|W^j\|_F\right)^2 \cdot 4f(w^{[0,n]}; x, y),$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

C. Additional Result - Convergence With Probability 1

We have discussed that existing results used bounded variance assumption, i.e, there exists $G > 0$, such that for $\forall \mathbf{w} \in \mathbb{R}^d$,

$$\mathbb{E}[\|\nabla f(\mathbf{w}; x, y) - \nabla F(\mathbf{w})\|^2] \leq G \tag{33}$$

in order to achieve convergence rate of $\mathcal{O}\left(\frac{1}{\sqrt{t}}\right)$ for nonconvex while we are using the condition in (7) in order to achieve convergence rate of $\mathcal{O}\left(\frac{1}{t}\right)$ for nonconvex. Adding $-\|\nabla F(\mathbf{w})\|^2$ to both sides of (7) is equivalent to

$$\mathbb{E}[\|\nabla f(\mathbf{w}; x, y) - \nabla F(\mathbf{w})\|^2] \leq 4D(F(\mathbf{w}) - \hat{f}) - \|\nabla F(\mathbf{w})\|^2, \ \forall \mathbf{w} \in \mathbb{R}^d, \tag{34}$$

since $\mathbb{E}[\|\nabla f(\mathbf{w}; x, y) - \nabla F(\mathbf{w})\|^2] = \mathbb{E}[\|\nabla f(\mathbf{w}; x, y)\|^2] - \|\nabla F(\mathbf{w})\|^2$. We notice that (34) together with the assumption that for $\forall \mathbf{w} \in \mathbb{R}^d$,

$$F(\mathbf{w}) \leq \frac{1}{4D}\|\nabla F(\mathbf{w})\|^2 + \frac{\hat{f}}{4D} + \frac{G}{4D}$$

could imply (33). However, it is hard to say which assumption is stronger or weaker than the other one. Using bounded variance assumption in (33), we achieve the following convergence w.p.1[5] result. (Motivated by (Nguyen et al., 2018c;b) for strongly convex and (van Dijk et al., 2018) for general convex.)

Theorem 3. *Suppose that F is L-smooth and has bounded variance, that is, there exists $G > 0$ such that*

$$\mathbb{E}[\|\nabla f(\mathbf{w}; x, y) - \nabla F(\mathbf{w})\|^2] \leq G, \ \forall \mathbf{w} \in \mathbb{R}^d.$$

Consider SGD (Algorithm 1) with a stepsize sequence such that $0 < \eta_t < \frac{2}{L}$ and

$$\sum_{t=0}^{\infty}\eta_t = \infty \text{ and } \sum_{t=0}^{\infty}\eta_t^2 < \infty.$$

Then, the following holds w.p.1 (almost surely)

$$\|\nabla F(\mathbf{w}_t)\|^2 \to 0.$$

We note that the theorem above does not tell us the convergence rate.

[5] with probability 1

C.1. Proof of Theorem 3

Lemma 10 ((Bertsekas, 2015)). *Let Y_k, Z_k, and W_k, $k = 0, 1, \ldots$, be three sequences of random variables and let $\{\mathcal{F}_k\}_{k\geq 0}$ be a filtration, that is, σ-algebras such that $\mathcal{F}_k \subset \mathcal{F}_{k+1}$ for all k. Suppose that:*

- *The random variables Y_k, Z_k, and W_k are nonnegative, and $\mathcal{F}_k$-measurable.*
- *For each k, we have $\mathbb{E}[Y_{k+1}|\mathcal{F}_k] \leq Y_k - Z_k + W_k$.*
- *There holds, w.p.1,*

$$\sum_{k=0}^{\infty} W_k < \infty.$$

Then, we have, w.p.1,

$$\sum_{k=0}^{\infty} Z_k < \infty \text{ and } Y_k \to Y \geq 0.$$

Theorem 3. *Suppose that F is L-smooth and has bounded variance, that is, there exists $G > 0$ such that*

$$\mathbb{E}[\|\nabla f(\mathbf{w}; x, y) - \nabla F(\mathbf{w})\|^2] \leq G, \ \forall \mathbf{w} \in \mathbb{R}^d.$$

Consider SGD (Algorithm 1) with a stepsize sequence such that $0 < \eta_t < \frac{2}{L}$ and

$$\sum_{t=0}^{\infty} \eta_t = \infty \text{ and } \sum_{t=0}^{\infty} \eta_t^2 < \infty.$$

Then, the following holds w.p.1 (almost surely)

$$\|\nabla F(\mathbf{w}_t)\|^2 \to 0.$$

Proof. Let $\mathcal{F}_t = \sigma(\mathbf{w}_0, \mathbf{w}_1, \ldots, \mathbf{w}_t)$ be the σ-algebra generated by $\mathbf{w}_0$, $\mathbf{w}_1$, $\ldots$, $\mathbf{w}_t$. From L-smooth property of F, we have

$$\begin{aligned}
\mathbb{E}[F(\mathbf{w}_{t+1})|\mathcal{F}_t] &\leq F(\mathbf{w}_t) - \eta_t \|\nabla F(\mathbf{w}_t)\|^2 + \frac{\eta_t^2 L}{2} \mathbb{E}[\|\nabla f(\mathbf{w}_t; x_t, y_t)\|^2 | \mathcal{F}_t] \\
&\leq F(\mathbf{w}_t) - \eta_t \left(1 - \frac{\eta_t L}{2}\right) \|\nabla F(\mathbf{w}_t)\|^2 + \frac{\eta_t^2 L}{2} \mathbb{E}[\|\nabla f(\mathbf{w}_t; x_t, y_t) - \nabla F(\mathbf{w}_t)\|^2 | \mathcal{F}_t] \\
&\leq F(\mathbf{w}_t) - \eta_t \left(1 - \frac{\eta_t L}{2}\right) \|\nabla F(\mathbf{w}_t)\|^2 + \frac{\eta_t^2 L}{2} G,
\end{aligned}$$

where the first inequality follows since $\mathbb{E}[\|\nabla f(\mathbf{w}_t; x_t, y_t) - \nabla F(\mathbf{w}_t)\|^2 | \mathcal{F}_t] = \mathbb{E}[\|\nabla f(\mathbf{w}_t; x_t, y_t)\|^2 | \mathcal{F}_t] - \|\nabla F(\mathbf{w}_t)\|^2$; and the last inequality follows since F has bounded variance. Note that $\left(1 - \frac{\eta_t L}{2}\right) > 0$ since $0 < \eta_t < \frac{2}{L}$.

Adding $-\hat{f}$ to both sides, we have

$$\mathbb{E}[F(\mathbf{w}_{t+1}) - \hat{f} | \mathcal{F}_t] \leq [F(\mathbf{w}_t) - \hat{f}] - \eta_t \left(1 - \frac{\eta_t L}{2}\right) \|\nabla F(\mathbf{w}_t)\|^2 + \frac{\eta_t^2 L}{2} G$$

Since $\frac{LG}{2} \sum_{t=0}^{\infty} \eta_t^2 < \infty$ w.p.1 and $F(\mathbf{w}_t) - \hat{f} \geq 0$, by Lemma 10, we have w.p.1

$$F(\mathbf{w}_t) - \hat{f} \to \hat{F} \geq 0,$$
$$\sum_{t=0}^{\infty} \eta_t \left(1 - \frac{\eta_t L}{2}\right) \|\nabla F(\mathbf{w}_t)\|^2 < \infty.$$

We want to show that $\|\nabla F(\mathbf{w}_t)\|^2 \to 0$ w.p.1. Proving by contradiction, we assume that there exists $\epsilon > 0$ and t_0, s.t., $\|\nabla F(\mathbf{w}_t)\|^2 \geq \epsilon$ for $\forall t \geq t_0$. Hence,

$$\sum_{t=t_0}^{\infty} \eta_t \left(1 - \frac{\eta_t L}{2}\right) \|\nabla F(\mathbf{w}_t)\|^2 \geq \epsilon \sum_{t=t_0}^{\infty} \left(1 - \frac{\eta_t L}{2}\right) \eta_t$$
$$= \epsilon \sum_{t=t_0}^{\infty} \eta_t - \frac{\epsilon L}{2} \sum_{t=t_0}^{\infty} \eta_t^2 = \infty.$$

This is a contradiction. Therefore, $\|\nabla F(\mathbf{w}_t)\|^2 \to 0$ w.p.1.

□

D. Additional Experiments

In this section, we provide the comparisons of the SGD algorithm with the DTN learning rate scheme with other algorithms.

D.1. Convex - Logistic Regression

We first conduct experiments on logistic regression problems in Section 4.1.

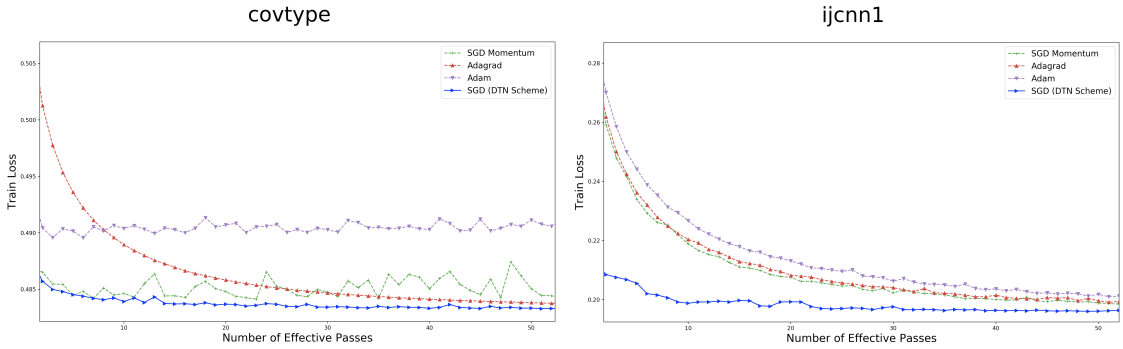

Figure 7: Comparisons of Train Loss among SGD with DTN scheme, SGD Momentum, Adagrad, Adam on *covtype* and *ijcnn1* datasets

Figure 7 shows comparisons among SGD with our DTN scheme, SGD with Momentum (Qian, 1999), Adagrad (Duchi et al., 2011), Adam (Kingma & Ba, 2014). We depicted the value of train loss as $F(\mathbf{w})$ defined in (1) for the y-axis and "number of effective passes" (or number of epochs, where an epoch is the equivalent of m gradient evaluations for the x-axis. We ran SGD Momentum with different learning rate $\{0.1, 0.01, 0.001, 0.0001\}$ with momentum parameter $\{0.5, 0.6, 0.7, 0.8, 0.9\}$; Adagrad with different learning rate $\{0.1, 0.01, 0.001, 0.0001\}$; Adam with default setting as in (Kingma & Ba, 2014), i.e. learning rate 0.001, and choose the best run for each of them. Finally, we ran our learning scheme with different values of DTN parameter $\alpha' = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$ the initial learning rate $\eta_0 = \{0.01, 0.1, 1\}$ and choose the best run. We experimented with 10 runs and reported the average results. We observe that, SGD with the DTN scheme achieves improved overall performance compared to other algorithms in logistic regression problems in Figure 7.

D.2. Non-convex - Neural Networks

We then conduct experiments on neural network architectures in Section 4.2.

We introduce SGD with Momentum (SGD-M) (Qian, 1999) as in Algorithm 2. Experiments show that SGD-M also works with the DTN scheme. The theoretical convergence fo SGD-M with the DTN scheme remains an open question.

We compare the performance of both SGD and SGD-M with the DTN scheme with some well-known algorithm for neural networks including Adagrad (Duchi et al., 2011), Adam (Kingma & Ba, 2014), RMSProp (Tieleman & Hinton, 2012), and ADADELTA (Zeiler, 2012) with default setting parameters in Tensorflow. We ran our learning scheme with different values of DTN parameter $\alpha' = \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10\}$ and the initial learning rate $\eta_0 = \{0.01, 0.1, 1\}$ and choose the best run. Figures 8 and 9 show comparisons of train loss and test accuracy, respectively.

Algorithm 2 SGD with Momentum (SGD-M)

Initialize: $\mathbf{w}_0$, momentum parameter $0 < \mu < 1$
Iterate:
for $t = 0, 1, \ldots$ **do**
Choose a step size (i.e., learning rate) $\eta_t > 0$.
Generate a realization of a random variable (x_t, y_t).
Compute a stochastic gradient $\nabla f(\mathbf{w}_t; x_t, y_t)$.
Update the new iterate

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla f(\mathbf{w}_t; x_t, y_t) + \mu(\mathbf{w}_t - \mathbf{w}_{t-1}).$$

end for

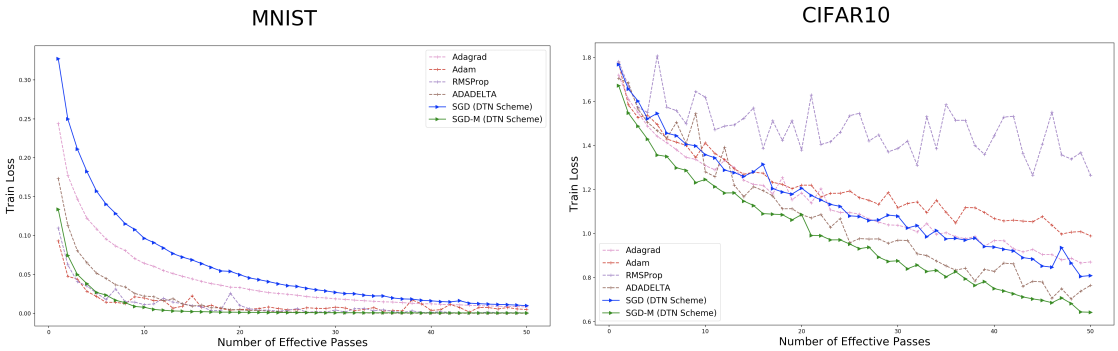

Figure 8: Comparisons of Train Loss among SGD and SGD-M with DTN scheme, Adagrad, Adam, RMSProp, ADADELTA on *MNIST* and *CIFAR-10* datasets

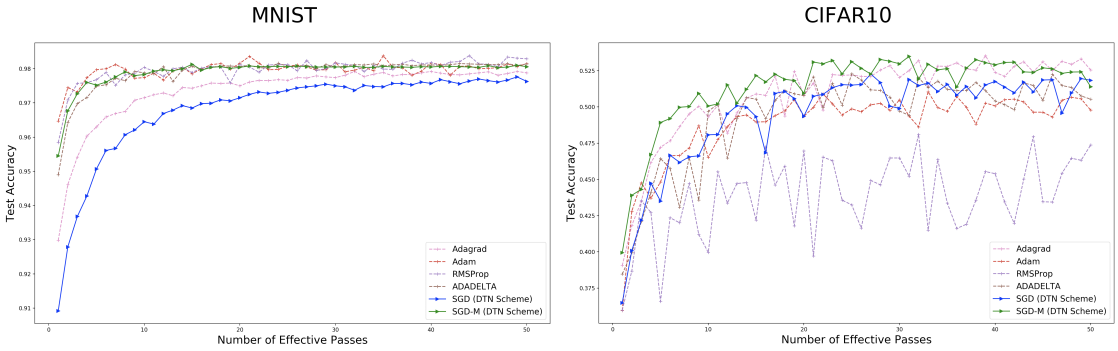

Figure 9: Comparisons of Test Accuracy among SGD and SGD-M with DTN scheme, Adagrad, Adam, RMSProp, ADADELTA on *MNIST* and *CIFAR-10* datasets