An Alternative View: When Does SGD Escape Local Minima?

Robert Kleinberg [1] **Yuanzhi Li** [2] **Yang Yuan** [1]

Abstract

Stochastic gradient descent (SGD) is widely used in machine learning. Although being commonly viewed as a fast but not accurate version of gradient descent (GD), it always finds better solutions than GD for modern neural networks. In order to understand this phenomenon, we take an alternative view that SGD is working on the convolved (thus smoothed) version of the loss function. We show that, even if the function f has many bad local minima or saddle points, as long as for every point x, the weighted average of the gradients of its neighborhoods is one point convex with respect to the desired solution x^*, SGD will get close to, and then stay around x^* with constant probability. Our result identifies a set of functions that SGD provably works, which is much larger than the set of convex functions. Empirically, we observe that the loss surface of neural networks enjoys nice one point convexity properties locally, therefore our theorem helps explain why SGD works so well for neural networks.

1. Introduction

Nowadays, stochastic gradient descent (SGD), as well as its variants (Adam (Kingma & Ba, 2014), Momentum (Sutskever et al., 2013), Adagrad (Duchi et al., 2011), etc.) have become the de facto algorithms for training neural networks. SGD runs iterative updates for the weights x_t: $x_{t+1} = x_t - \eta v_t$, where η is the step size [1]. v_t is the stochastic gradient that satisfies $\mathbb{E}[v_t] = \nabla f(x_t)$, and is usually computed using a mini-batch of the dataset.

In the regime of convex optimization, SGD is proved to be a nice tradeoff between accuracy and efficiency: it requires more iterations to converge, but fewer gradient evaluations per iteration. Therefore, for the standard empirical risk minimizing problems with n points and smoothness L, to get to ϵ-close to x^*, GD needs $O(Ln/\epsilon)$ gradient evaluations (Nesterov, 2014), but SGD with reduced variance only needs $O(n \log \frac{1}{\epsilon} + \frac{L}{\epsilon})$ gradient evaluations (Johnson & Zhang, 2013; Defazio et al., 2014; Schmidt et al., 2016; Allen-Zhu & Yuan, 2016). In these scenarios, noise is a by-product of cheap gradient computation, and does not help training.

By contrast, for non-convex optimization problems like training neural networks, noise seems crucial. It is observed that with the help of noisy gradients, SGD does not only converge faster, but also converge to a better solution compared with GD (Keskar et al., 2017). To formally understand this phenomenon, people have analyzed the role of noise in various settings. For example, it is proved that noise helps to escape saddle points (Ge et al., 2015; Jin et al., 2017), gives better generalization (Hardt et al., 2015; Mou et al., 2017), and also guarantees polynomial hitting time of good local minima under some assumptions (Zhang et al., 2017).

However, it is still unclear why SGD could converge to better local minima than GD. Empirically, in additional to the gradient noise, the step size is observed to be a key factor in optimization. More specifically, small step size helps refine the network and converge to a local minimum, while large step size helps escape the current local minimum and go towards a better one (Huang et al., 2017; Loshchilov & Hutter, 2017). Thus, standard training schedule for modern networks uses large step size first, and shrinks it later (He et al., 2016; Huang et al., 2016). While using large step sizes to escape local minima matches with intuition, the existing analysis on SGD for non-convex objectives always considers the small-step-size settings (Ge et al., 2015; Jin et al., 2017; Hardt et al., 2015; Zhang et al., 2017).

See Figure 1 for an illustration. Consider the scenario that for some x_t, instead of pointing to the solution (not shown), its negative gradient points to a bad local minimum $x_\circ$, so following the full gradient we will arrive $y_t \triangleq x_t - \eta \nabla f(x_t)$. Fortunately, since we are running SGD, the actual direction we take is $-\eta v_t = -\eta(\nabla f(x_t) + \omega_t)$, where ω_t is the noise with $\mathbb{E}[\omega_t] = 0, \omega_t \sim W(x_t)$[2]. As we show in Figure 1, if we take a large η, we may get out of the basin region with

[1]Department of Computer Science, Cornell University [2]Department of Computer Science, Princeton University. Correspondence to: Yang Yuan <yangyuan@cs.cornell.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

[1]In this paper, we use step size and learning rate interchangeably.

[2]$W(x_t)$ is data dependent.

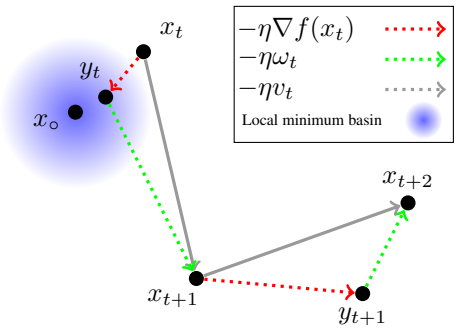


Figure 1: SGD path $x_t \to x_{t+1}$ can be decomposed into $x_t \to y_t \to x_{t+1}$. If the local minimum basin has small diameter, the gradient at x_{t+1} will point away from the basin.

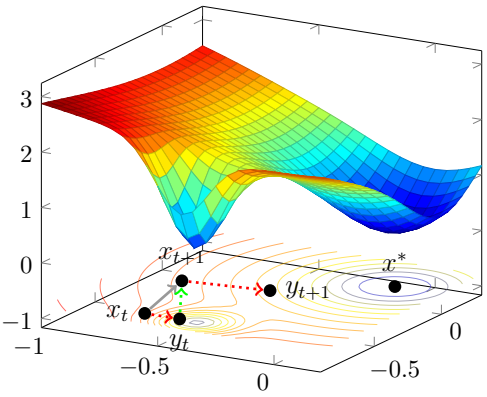


Figure 2: 3D version of Figure 1: SGD could escape a local minimum within one step.

the help of noise, i.e., from y_t to x_{t+1}. Here, getting out of the basin means the negative gradient at x_{t+1} no longer points to $x_\circ$ (See also Figure 2).

To formalize this intuition, instead of analyzing the sequence $x_t \to x_{t+1}$, let us look at the sequence $y_t \to y_{t+1}$, where y_t is defined to be $x_t - \eta\nabla f(x_t)$, as in the preceding paragraph. The SGD algorithm never computes these vectors y_t, but we are only using them as an analysis tool. From the equation $x_{t+1} = y_t - \eta\omega_t$ we obtain the following update rule relating y_{t+1} to y_t.

$$y_{t+1} = y_t - \eta\omega_t - \eta\nabla f(y_t - \eta\omega_t) \tag{1}$$

The random vector $\eta\omega_t$ in (1) has expectation 0, so if we take the expectation of both sides of (1), we get $\mathbb{E}_{\omega_t}[y_{t+1}] = y_t - \eta\nabla\mathbb{E}_{\omega_t}[f(y_t - \eta\omega_t)]$. Therefore, if we define g_t to be the function $g_t(y) = \mathbb{E}_{\omega_t}[f(y - \eta\omega_t)]$, which is simply the original function f convolved with the η-scaled gradient noise, then the sequence y_t is approximately doing gradient descent on the sequence of functions (g_t).

This alternative view helps to explain why SGD converges to a good local minimum, even when f has many other sharp local minima. Intuitively, sharp local minima are eliminated by the convolution operator that transforms f to g_t, since convolution has the effect of smoothing out short-range fluctuations. This reasoning ensures that SGD converges to a good local minimum under much weaker conditions, because instead of imposing convexity or one-point convexity requirements on f itself, we only require those properties to hold for the smoothed functions obtained from f by convolution. We can formalize the foregoing argument using the following assumption.

Assumption 1 (Main Assumption)**.** *For a fixed point x^*[3], noise distribution $W(x)$, step size η, the function f is c-one point strongly convex with respect to x^* after convolved with noise. That is, for any x, y in domain $\mathbb{D}$ s.t. $y = x - \eta\nabla f(x)$,*

$$\langle -\nabla\mathbb{E}_{\omega\in W(x)} f(y - \eta\omega), x^* - y\rangle \geq c\|x^* - y\|_2^2 \tag{2}$$

For point y, since the direction $x^* - y$ points to x^*, by having positive inner product with $x^* - y$, we know the direction $-\eta\nabla f(y_t - \eta\omega_t)$ in (1) approximately points to x^* in expectation (See more discussion on one point convexity in Appendix). Therefore, y_t will converge to x^* with decent probability:

Theorem 1 (Main Theorem, Informal)**.** *Assume f is smooth, for every $x \in \mathbb{D}$, $W(x)$ s.t., $\max_{\omega\sim W(x)} \|\omega\|_2 \leq r$. Also assume η is bounded by a constant, and Assumption 1 holds with x^*, η, and c. For $T_1 \geq \tilde{O}(\frac{1}{\eta c})$[4], and any $T_2 > 0$, with probability at least $1/2$, we have $\|y_t - x^*\|_2^2 \leq O(\log(T_2)\frac{\eta r^2}{c})$ for any t s.t., $T_1 + T_2 \geq t \geq T_1$.*

Notice that our main theorem not only says SGD will get close to x^*, but also says with constant probability, SGD will stay close to x^* for the future T_2 steps. As we will see in Section 5, we observe that Assumption 1 holds along the SGD trajectory for the modern neural networks when the noise comes from real data mini-batches. Moreover, the SGD trajectory matches with our theory prediction in practice.

Our main theorem can also help explain why SGD could escape "sharp" local minima and converge to "flat" local minima in practice (Keskar et al., 2017). Indeed, the sharp local minima have small loss value and small diameter, so after convolved with the noise kernel, they easily disappear, which means Assumption 1 holds. However, flat local minima have large diameter, so they still exists after convolution.

[3]Notice that x^* is not necessarily the global optimal in the original function f due to the convolution operator.

[4]We use $\tilde{O}$ to hide log terms here.

In that case, our main theorem says, it is more likely that SGD will converge to flat local minima, instead of sharp local minima.

1.1. Related Work

Previously, people already realized that the noise in the gradient could help SGD to escape saddle points (Ge et al., 2015; Jin et al., 2017) or achieve better generalization (Hardt et al., 2015; Mou et al., 2017). With the help of noise, SGD can also be viewed as doing approximate Bayesian inference (Mandt et al., 2017) or variational inference (Chaudhari & Soatto, 2017). Besides, it is proved that SGD with extra noise could "hit" a local minimum with small loss value in polynomial time under some assumptions (Zhang et al., 2017). However, the extra noise is too big to guarantee convergence, and that model cannot deal with escaping sharp local minima.

Escaping sharp local minima for neural network is important, because it is conjectured (although controversial (Dinh et al., 2017)) that flat local minima may lead to better generalization (Hochreiter & Schmidhuber, 1995; Keskar et al., 2017; Chaudhari et al., 2016). It is also observed that the correct learning rate schedule (small or large) is crucial for escaping bad local minima (Huang et al., 2017; Loshchilov & Hutter, 2017). Furthermore, solutions that are farther away from the initialization may lead to wider local minima and better generalization (Hoffer et al., 2017). Under a Bayesian perspective, it is shown that the noise in stochastic gradient could drive SGD away from sharp minima, which decides the optimal batch size (Smith & Le, 2017). There are also explanations for why small batch methods prefers flat minima while large batch methods are not, by investigating the canonical quadratic sums problem (Patel, 2017).

To visualize the loss surface of neural network, a common practice is projecting it onto a one dimensional line (Goodfellow & Vinyals, 2014), which was observed to be convex. For the simple two layer neural network, a local one point strongly convexity property provably holds under Gaussian input assumption (Li & Yuan, 2017).

2. Motivating Example

Let us first see a simple example in Figure 3. We use $\mathrm{F}_{r,c}$ to denote the sub-figure at row r and column c. The function f at $\mathrm{F}_{1,1}$ is a approximately convex function, but very spiky. Therefore, GD easily gets stuck at various local minima, see $\mathrm{F}_{2,1}$. However, we want to get rid of those spurious local minima, and get a point near $x^* = 0$.

If we take the alternative view that SGD works on the convolved version of f ($\mathrm{F}_{1,2}$, $\mathrm{F}_{1,3}$, $\mathrm{F}_{1,4}$), we find that those functions are much smoother and contain few local minima. However, the gradient noise here is a double-edged sword. On one hand, if the noise is small, the convolved f is still somewhat non-convex, then SGD may find a few bad local minima as shown in $\mathrm{F}_{2,2}$. On the other hand, if the noise is too large, the noise dominates the gradient, and SGD will act like random walk, see $\mathrm{F}_{2,4}$.

$\mathrm{F}_{2,3}$ seems like a nice tradeoff, as all trials converges to a local region near 0, but the region is too big (most points are in $[-1.5, 1.5]$). In order to get closer to 0, we may "restart" SGD with a point in $[-1.5, 1.5]$, using smaller noise level 0.15. Recall in $\mathrm{F}_{2,2}$, SGD fails because the convolved f has a few non-convex regions ($\mathrm{F}_{1,2}$), so SGD may find spurious local minima. However, those local minima are outside $[-1.5, 1.5]$. The convolved f in $\mathrm{F}_{1,2}$ restricted in $[-1.5, 1.5]$ is pretty convex, so if we start a point in this region, SGD converges to a smaller local region centered at 0, see $\mathrm{F}_{3,2}$.

We may do this iteratively, with even smaller noise levels and smaller initialization regions, and finally we will get pretty close to 0 with decent probability, see $\mathrm{F}_{3,3}$ and $\mathrm{F}_{3,4}$.

3. Main Theorem

Definition 1 (Smoothness). *Function $f \in \mathbb{R}^d \to \mathbb{R}$ is L-smooth, if for any $x, y \in \mathbb{R}^d$,*

$$f(y) \le f(x) + \langle f'(x), y - x \rangle + \frac{L}{2}\|y - x\|_2^2$$

Assume that we are running SGD on the sequence $\{x_t\}$. Recall the update rule (1) for y_t. Our main theorem says that $\{y_t\}$ is converging to x^* and will stay around x^* afterwards.

Theorem 1 (Main Theorem). *Assume f is L-smooth, for every $x \in \mathbb{D}$, $W(x)$ s.t., $\max_{\omega \sim W(x)} \|\omega\|_2 \le r$. For a fixed target solution x^*, if there exists constant $c, \eta > 0$, such that Assumption 1 holds with x^*, η, c, and $\eta < \min\{\frac{1}{2L}, \frac{c}{L^2}, \frac{1}{2c}\}$, $\lambda \triangleq 2\eta c - \eta^2 L^2$, $b \triangleq \eta^2 r^2 (1 + \eta L)^2$. Then for any fixed $T_1 \ge \frac{\log(\lambda \|y_0 - x^*\|_2^2 / b)}{\lambda}$ and $T_2 > 0$, with probability at least $1/2$, we have $\|y_T - x^*\|_2^2 \le \frac{20b}{\lambda}$ and $\|y_t - x^*\|_2^2 \le O\left(\frac{\log(T_2) b}{\lambda}\right)$ for all t s.t., $T_1 + T_2 \ge t \ge T_1$.*

We defer the proof to Section 4.

Remark. For fixed c, there exists a lower bound on η to satisfy Assumption 1, so η cannot be arbitrarily small. However, the main theorem says within $T_1 + T_2$ steps, SGD will stay in a local region centered at x^* with diameter $O\left(\frac{\log(T_2) b}{\lambda}\right)$, which is essentially $\tilde{O}(\eta r^2 / c)$ that scales with η. In order to get closer to x^*, a common trick in practice is to restart SGD with smaller step size η' within the local region. If f inside this region has better geometric properties (which is usually true), one gets better convergence guarantee:

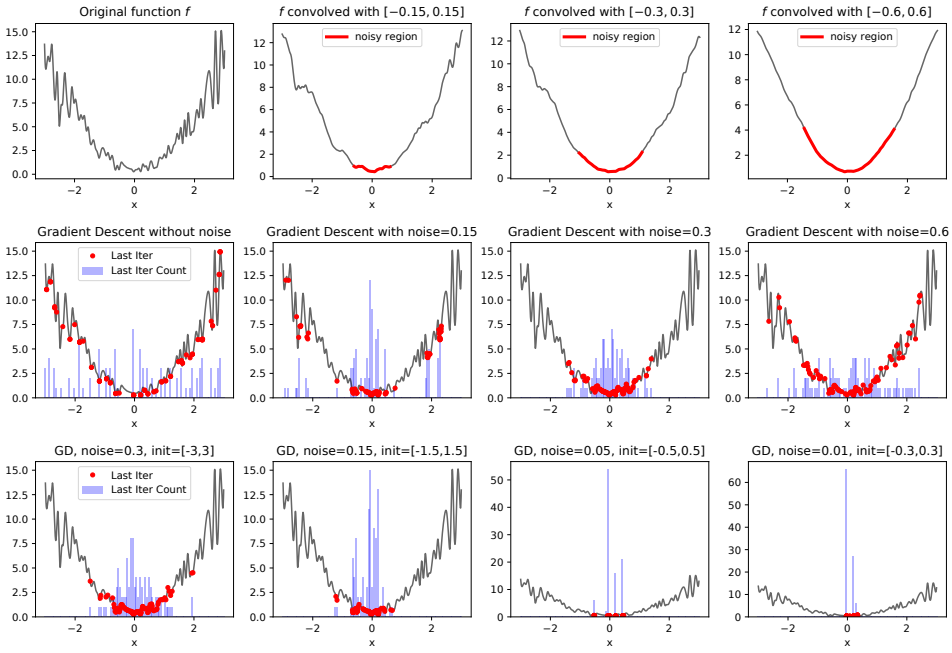


Figure 3: Running SGD on a spiky function f. **Row 1:** f gets smoother after convolving with uniform random noise. **Row 2:** Run SGD with different noise levels. Every figure is obtained with 100 trials with different random initializations. Red dots represent the last iterates of these trials, while blue bars represent the cumulative counts. GD without noise easily gets stuck at various local minima, while SGD with appropriate noise level converges to a local region. **Row 3:** In order to get closer to x^*, one may run SGD in multiple stages with shrinking learning rates.

Corollary 2 (Shrinking Learning Rate)**.** *If the assumptions in Theorem 1 holds, and f restricted in the local region $\mathbb{D}' \triangleq \{x | \|x - x^*\| \le \frac{20b}{\lambda}\}$ satisfy the same assumption with $c' > c, \eta' < \eta$, then if we run SGD with η for the first $T_1 \ge \frac{\log(\frac{\lambda d}{b})}{\lambda}$ steps, and with η' for the next $T_2 \ge \frac{\log(\frac{\lambda}{b'}\frac{20b'}{\lambda})}{\lambda'}$ steps, with probability at least $1/4$, we have $\|y_{T_1+T_2} - x^*\|_2^2 \le \frac{20b'}{\lambda'} < \frac{20b}{\lambda}$.*

This corollary can be easily generalized to shrink the learning rate multiple times.

Our main theorem is based on the important assumption that the step size is bounded. If the step size is too big, even if the whole function f is one point convex (a stronger assumption than Assumption 1), and we run full gradient descent, we may not keep getting closer to x^*, as we show below.

Theorem 3. *For function f, if $\forall x, \langle -\nabla f(x), x^* - x\rangle \le c'\|x^* - x\|_2^2$, and we are at the point x_t. If we run full gradient descent with step size $\eta > \frac{2c'\|x_t - x^*\|_2^2}{\|\nabla f(x_t)\|_2^2}$, we have $\|x_{t+1} - x^*\|_2^2 \ge \|x_t - x^*\|_2^2$.*

Proof. The proof is straightforward and we defer it to Appendix C. □

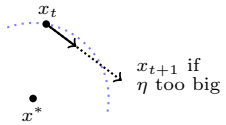


Figure 4: When step size is too big, even the gradient is one point convex, we may still go farther away from x^*.

This theorem can be best illustrated with Figure 4. If η is too big, although the gradient (the arrow) is pointing to the approximately correct direction, x_{t+1} will be farther away from x^* (going outside of the x^*-centered ball).

Although this theorem analyzes the simple full gradient case, SGD is similar. In the high dimensional case, it is natural to assume that most of the noise will be orthogonal to the direction of $x_t - x^*$, therefore with additional noise inside the stochastic gradient, a large step size will drive x_{t+1} away from x^* more easily.

Therefore, our paper provides a theoretical explanation for why picking step size is so important (too big or too small will not work). We hope it could lead to more practical guidelines in the future.

4. Proof for Theorem 1

In the proof, we will use the following lemma.

Theorem 4 (Azuma). *Let $X_1, X_2, \cdots, X_n$ be independent random variables satisfying $|X_i - E(X_i)| \le c_i$, for $1 \le i \le n$. We have the following bound for the sum $X = \sum_{i=1}^n X_i$:*

$$\Pr(|X - E(X)| \ge \lambda) \le 2e^{-\frac{\lambda^2}{2\sum_{i=1}^n c_i^2}}.$$

Our proof has four steps.

Step 1. Since Assumption 1 holds, we show that SGD always makes progress towards x^* in expectation, plus some noise.

Let filtration $\mathcal{F}_t = \sigma\{\omega_0, \cdots, \omega_{t-1}\}$, where $\sigma\{\cdot\}$ denotes the sigma field. Notice that for any $\omega_t \sim W(x_t)$, we have $\mathbb{E}[\omega_t|\mathcal{F}_t] = 0$.

Thus,

$$\begin{aligned}
&\mathbb{E}[\|y_{t+1} - x^*\|_2^2|\mathcal{F}_t] \\
=&\mathbb{E}[\|y_t - \eta\omega_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2|\mathcal{F}_t] \\
=&\mathbb{E}\Big[\|y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2 + \|\eta\omega_t\|_2^2 \\
&- 2\langle\eta\omega_t, y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\rangle|\mathcal{F}_t\Big] \\
\le&\mathbb{E}\Big[\|y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2 + \eta^2 r^2 \\
&- 2\langle\eta\omega_t, -\eta\nabla f(y_t - \eta\omega_t) + \eta\nabla f(y_t) - \eta\nabla f(y_t)\rangle|\mathcal{F}_t\Big] \\
\le&\mathbb{E}\Big[\|y_t - x^*\|_2^2 + \eta^2\|\nabla f(y_t - \eta\omega_t)\|_2^2 \\
&- 2\eta\langle-\nabla f(y_t - \eta\omega_t), x^* - y_t\rangle + \eta^2 r^2 + 2\eta^3 r^2 L|\mathcal{F}_t\Big] \\
\le&\|y_t - x^*\|_2^2 + \mathbb{E}\Big[\eta^2\|\nabla f(y_t - \eta\omega_t)\|_2^2|\mathcal{F}_t\Big] + \eta^2 r^2 \\
&- 2\eta\langle-\nabla\mathbb{E}_{\omega_t\in W(x_t)} f(y_t - \eta\omega_t), x^* - y_t\rangle + 2\eta^3 r^2 L \\
\le&(1 - 2\eta c)\|y_t - x^*\|_2^2 + \eta^2 r^2 + 2\eta^3 r^2 L \\
&+ \mathbb{E}\Big[\eta^2 L^2\|x^* - y_t + \eta\omega_t\|_2^2|\mathcal{F}_t\Big] \\
\le&(1 - 2\eta c)\|y_t - x^*\|_2^2 + \eta^2 r^2 + 2\eta^3 r^2 L + \eta^2 L^2\|x^* - y_t\|_2^2 \\
&+ \eta^4 r^2 L^2 \\
=&(1 - 2\eta c + \eta^2 L^2)\|y_t - x^*\|_2^2 + \eta^2 r^2(1 + \eta L)^2
\end{aligned}$$

Step 2. Since SGD makes progress in every step, after many steps, SGD gets very close to x^* in expectation. By Markov inequality, this event holds with large probability.

Notice that since $\eta < \frac{c}{L^2}$, we have $\lambda = 2\eta c - \eta^2 L^2 > \eta c > 0$. Recall $b \triangleq \eta^2 r^2(1 + \eta L)^2$, we get:

$$\mathbb{E}[\|y_{t+1} - x^*\|_2^2|\mathcal{F}_t] \le (1 - \lambda)\|y_t - x^*\|_2^2 + b$$

Let $G_t = (1 - \lambda)^{-t}(\|y_t - x^*\|_2^2 - \frac{b}{\lambda})$, we get:

$$\mathbb{E}[G_{t+1}|\mathcal{F}_t] \le G_t$$

That means, G_t is a supermartingale.

We have

$$\mathbb{E}[G_{T_1}|\mathcal{F}_{T_1-1}] \le G_0$$

Which gives

$$\mathbb{E}\left[\|y_{T_1} - x^*\|_2^2 - \frac{b}{\lambda}\Big|\mathcal{F}_{T_1-1}\right] \le (1 - \lambda)^{T_1}\|y_0 - x^*\|_2^2$$

That is,

$$\mathbb{E}[\|y_{T_1} - x^*\|_2^2|\mathcal{F}_{T_1-1}] \le \frac{b}{\lambda} + (1 - \lambda)^{T_1}\|y_0 - x^*\|_2^2$$

Since $T_1 \ge \frac{\log\left(\frac{\lambda\|y_0 - x^*\|_2^2}{b}\right)}{\lambda}$, we get:

$$\mathbb{E}[\|y_{T_1} - x^*\|_2^2|\mathcal{F}_{T_1-1}] \le \frac{2b}{\lambda}$$

By Markov inequality, we know with probability at least 0.9,

$$\|y_{T_1} - x^*\|_2^2 \le \frac{20b}{\lambda} \tag{3}$$

For notational simplicity, for the analysis below we relabel the point y_{T_1} as y_0. Therefore, at time 0 we already have $\|y_0 - x^*\|_2^2 \le \frac{20b}{\lambda}$.

Step 3. Conditioned on the event that we are close to x^*, below we show that if for $t_0 > t \ge 0$, y_t is close to x^*, then y_{t_0} is also close to x^* with high probability.

Let $\zeta = \frac{9T_2}{4}$. Let event $\mathfrak{E}_t = \{\forall\tau \le t, \|y_\tau - x^*\| \le \mu\sqrt{\frac{b}{\lambda}} = \delta\}$, where μ is a parameter satisfies $\mu \ge \max\{8, 42\log^{\frac{1}{2}}(\zeta)\}$. If with probability $\frac{5}{9}$, $\mathfrak{E}_t$ holds for every $t \le T_2$, we are done.

By the previous calculation, we know that ($\mathbb{1}_{\mathfrak{E}_t}$ is the indicator function for $\mathfrak{E}_t$)

$$\mathbb{E}[G_t\mathbb{1}_{\mathfrak{E}_{t-1}}|\mathcal{F}_{t-1}] \le G_{t-1}\mathbb{1}_{\mathfrak{E}_{t-1}} \le G_{t-1}\mathbb{1}_{\mathfrak{E}_{t-2}}$$

So $G_t\mathbb{1}_{\mathfrak{E}_{t-1}}$ is a supermartingale, with the initial value G_0. In order to apply Azuma inequality, we first bound the following term (notice that we use $\mathbb{E}[\omega_t] = 0$ multiple times):

$$\begin{aligned}
&|G_{t+1}\mathbb{1}_{\mathfrak{E}_t} - \mathbb{E}[G_{t+1}\mathbb{1}_{\mathfrak{E}_t}|\mathcal{F}_t]| \\
=&(1 - \lambda)^{-t}|\|y_t - \eta\omega_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2 \\
&- \mathbb{E}[\|y_t - \eta\omega_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2|\mathcal{F}_t]|\mathbb{1}_{\mathfrak{E}_t} \\
\le&(1 - \lambda)^{-t}|2\langle-\eta\omega_t, y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\rangle + \|\eta\omega_t\|_2^2 \\
&+ \|y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2
\end{aligned}$$

$$
\begin{aligned}
&- \mathbb{E}[2\langle -\eta\omega_t, y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\rangle + \|\eta\omega_t\|_2^2 \\
&+ \|y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2 | \mathcal{F}_t] \\
=&(1-\lambda)^{-t}|\|\eta\omega_t\|_2^2 - \mathbb{E}[\|\eta\omega_t\|_2^2|\mathcal{F}_t] \\
&- 2\langle \eta\omega_t, y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\rangle \\
&+ \|y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2 \\
&- \mathbb{E}[2\langle \eta\omega_t, \eta\nabla f(y_t - \eta\omega_t)\rangle \\
&+ \|y_t - \eta\nabla f(y_t - \eta\omega_t) - x^*\|_2^2 | \mathcal{F}_t] \\
\le&(1-\lambda)^{-t}|\eta^2 r^2 + 2\eta r\|y_t - x^*\| + 2\langle \eta\omega_t, \eta\nabla f(y_t - \eta\omega_t)\rangle \\
&+ \|\eta\nabla f(y_t - \eta\omega_t) - \eta\nabla f(y_t) + \eta\nabla f(y_t)\|_2^2 \\
&- \mathbb{E}[\|\eta\nabla f(y_t - \eta\omega_t) - \eta\nabla f(y_t) + \eta\nabla f(y_t)\|_2^2 | \mathcal{F}_t] \\
&+ 2\langle y_t - x^*, \eta\nabla f(y_t - \eta\omega_t) \\
&- E[\eta\nabla f(y_t - \eta\omega_t)|\mathcal{F}_t]\rangle - \mathbb{E}[2\langle \eta\omega_t, \eta\nabla f(y_t - \eta\omega_t)\rangle | \mathcal{F}_t] \\
\le&(1-\lambda)^{-t}|\eta^2 r^2 + 2\eta r\|y_t - x^*\| + 4\eta^2 r\|\nabla f(y_t - \eta\omega_t)\|_2 \\
&+ \eta^2(2\eta^2 r^2 L^2 + 2\langle \nabla f(y_t), \nabla f(y_t - \eta\omega_t) - \nabla f(y_t) \\
&- \mathbb{E}[\nabla f(y_t - \eta\omega_t) - \nabla f(y_t)|\mathcal{F}_t]\rangle) \\
&+ 2\eta\Big\langle y_t - x^*, \nabla f(y_t - \eta\omega_t) - \nabla f(y_t) \\
&- E[\nabla f(y_t - \eta\omega_t) - \nabla f(y_t)|\mathcal{F}_t]\Big\rangle \\
=&(1-\lambda)^{-t}|\eta^2 r^2 + 2\eta r\|y_t - x^*\| + 4\eta^2 rL(\eta r + \|y_t - x^*\|_2) \\
&+ \eta^2\left(2\eta^2 r^2 L^2 + 4L\|y_t - x^*\|_2 \eta r L\right) + 4\eta^2 rL\|y_t - x^*\| \\
\le&(1-\lambda)^{-t}\left(3.5\eta^2 r^2 + 7\eta r\delta\right)
\end{aligned}
$$

Where the last inequality uses the fact that $\eta L \le \frac{1}{2}$ and $\|y_t - x^*\|_2 \le \delta$ (as $\mathbb{1}_{\mathfrak{E}_t}$ holds). Let $M \triangleq 3.5\eta^2 r^2 + 7\eta r\delta$. Let $d_\tau = |G_\tau \mathbb{1}_{\mathfrak{E}_{\tau-1}} - \mathbb{E}[G_\tau \mathbb{1}_{\mathfrak{E}_{\tau-1}}|\mathcal{F}_t]|$, we have

$$
\sum_{\tau=1}^{t} d_\tau^2 = \sum_{\tau=1}^{t}(1-\lambda)^{-2\tau}M^2
$$

$$
r_t = \sqrt{\sum_{\tau=1}^{t} d_\tau^2} = M\sqrt{\sum_{\tau=1}^{t}(1-\lambda)^{-2\tau}}
$$

Apply Azuma inequality (Theorem 4), for any $\zeta > 0$, we know

$$
\begin{aligned}
&\Pr(G_t\mathbb{1}_{\mathfrak{E}_{t-1}} - G_0 \ge \sqrt{2}r_t\log^{\frac{1}{2}}(\zeta)) \\
\le&\exp\left(\frac{-2r_t^2\log(\zeta)}{2\sum_{\tau=1}^{t} d_\tau^2}\right) = \exp^{-\log(\zeta)} = \frac{1}{\zeta}
\end{aligned}
$$

Therefore, with probability $1 - \frac{1}{\zeta}$,

$$
G_t\mathbb{1}_{\mathfrak{E}_{t-1}} \le G_0 + \sqrt{2}r_t\log^{\frac{1}{2}}(\zeta)
$$

Step 4. The inequality above says, if $\mathfrak{E}_{t-1}$ holds, i.e., for all $\tau \le t-1$, $\|y_\tau - x^*\| \le \delta$, then with probability $1 - \frac{1}{\zeta}$, G_t is bounded. If we can show from the upper bound of G_t that $\|y_t - x^*\| \le \delta$ is also true, we automatically get $\mathfrak{E}_t$ holds. In other words, that means if $\mathfrak{E}_{t-1}$ holds, then $\mathfrak{E}_t$ holds with probability $1 - \frac{1}{\zeta}$. Therefore, by applying this claim T_2 times, we get $\mathfrak{E}_{T_2}$ holds with probability $1 - \frac{T_2}{\zeta} = \frac{5}{9}$. Combining with inequality (3), we know with probability at least $1/2$, the theorem statement holds. Thus, it remains to show that $\|y_t - x^*\| \le \delta$.

If $G_t\mathbb{1}_{\mathfrak{E}_{t-1}} \le G_0 + \sqrt{2}r_t\log^{\frac{1}{2}}(\zeta)$, we know

$$
\begin{aligned}
&(1-\lambda)^{-t}\left(\|y_t - x^*\|_2^2 - \frac{b}{\lambda}\right) \\
\le&\|y_0 - x^*\|_2^2 - \frac{b}{\lambda} + \sqrt{2}r_t\log^{\frac{1}{2}}(\zeta)
\end{aligned}
$$

So

$$
\begin{aligned}
&\|y_t - x^*\|_2^2 \\
\le&(1-\lambda)^t\left(\|y_0 - x^*\|_2^2 + \sqrt{2}r_t\log^{\frac{1}{2}}(\zeta)\right) + \frac{b}{\lambda} \\
\le&\|y_0 - x^*\|_2^2 + \sqrt{2}(1-\lambda)^t r_t\log^{\frac{1}{2}}(\zeta) + \frac{b}{\lambda}
\end{aligned}
$$

Notice that

$$
\begin{aligned}
&(1-\lambda)^t r_t = (1-\lambda)^t M\sqrt{\sum_{\tau=1}^{t}(1-\lambda)^{-2\tau}} \\
=&M\sqrt{\sum_{\tau=1}^{t}(1-\lambda)^{2(t-\tau)}} = M\sqrt{\sum_{\tau=0}^{t-1}(1-\lambda)^{2\tau}} \\
\le&M\sqrt{\frac{1}{1-(1-\lambda)^2}} \le \frac{M}{\sqrt{\eta c}}
\end{aligned}
$$

The second last inequality holds because we know $\frac{1}{1-(1-\lambda)^2} = \frac{1}{2\lambda-\lambda^2} \le \frac{1}{\lambda} \le \frac{1}{\eta c}$, since $\lambda = 2\eta c - \eta^2 L^2 \le 2\eta c < 1$, and $\lambda > \eta c$.

That means,

$$
\begin{aligned}
&\|y_t - x^*\|_2^2 \le \|y_0 - x^*\|_2^2 + \frac{\sqrt{2}M}{\sqrt{\eta c}}\log^{\frac{1}{2}}(\zeta) + \frac{b}{\lambda} \\
\le&\frac{\sqrt{2}(3.5\eta^2 r^2 + 7\eta r\delta)}{\sqrt{\eta c}}\log^{\frac{1}{2}}(\zeta) + \frac{21b}{\lambda}
\end{aligned}
$$

It remains to prove the following lemma, which we defer to Appendix B.

Lemma 5.

$$
\frac{\sqrt{2}(3.5\eta^2 r^2 + 7\eta r\delta)}{\sqrt{\eta c}}\log^{\frac{1}{2}}(\zeta) + \frac{21b}{\lambda} \le \delta^2
$$

Therefore, $\|y_t - x^*\| \le \delta$. Combining the 4 steps together, we have proved the theorem.

5. Empirical Observations

In this section, we explore the loss surfaces of modern neural networks, and show that they enjoy many nice one point convex properties. Therefore, our main theorem could be used for explaining why SGD works so well in practice.

5.1. The SGD trajectory is one point convex

It is well known that the loss surface of neural network is highly non-convex, with numerous local minima. However, we observe that the loss surface is consisted of many one point convex basin region, while each time SGD traverses one of such regions.

See Figure 5a for details. We ran experiments on Resnet (He et al., 2016) (34 layers, ≈ 1.2M parameters), Densenet (Huang et al., 2016) (100 layers, ≈ 0.8M parameters) on cifar10 and cifar100, each for 5 trials with 300 epochs and different initializations. For the start of every epoch x_t in each trial, we compute the inner product between the negative gradient $-\nabla f(x_t)$ and the direction $x_{300}-x_t$. In Figure 5a, we plot the minimum value for every epoch among 5 trials for each setting. Notice that except for the starting period of densenet on Cifar-10, all the other networks in all trials have positive inner products, which shows that the trajectory of SGD (except the starting period) is one point convex with respect to the final solution[5]. In these experiments, we have used the standard step size schedule (0.1 initially, 0.01 after epoch 150, and 0.001 after epoch 225). However, we got the same observation when using smoothly decreasing step sizes (shrink by 0.99 per epoch).

5.2. The neighborhood of the trajectory is one point convex

Having a one point convex trajectory for 5 trials does not suffice to show SGD always has a simple and easy trajectory, due to the randomness of the stochastic gradient. Indeed, by a slight random perturbation, SGD might be in a completely different trajectory that is far from being one point convex to the final solution. However, in this subsection, we show that it is not the case, as the SGD trajectory is one point convex after convolving with uniform ball with radius 0.5. That means, the whole neighborhood of the SGD trajectory is one point convex with respect to the final solution.

In this experiment, we tried Resnet (34 layers, ≈ 1.2M parameters), Densenet (100 layers, ≈ 0.8M parameters) on cifar10 and cifar100. For every epoch in each setting, we take one point and look at its neighborhood with radius 0.5 (upper bound of the length of one SGD step, as we will show below). We take 100 random points inside each neighborhood to verify Assumption 1. More specifically, for every random point w in the neighborhood of x_t, we computer $\langle -\nabla f(w), x_{300} - x_t \rangle$. Figure 5b shows the mean value (solid line), as well as upper and lower bound of the inner product (shaded area). As we can see, the inner products for all epochs in every setting have small variances, and are always positive. Although we could not verify Assumption 1 by computing the exact expectation due to limited computational resources, from Figure 5b and Hoeffding bound (Lemma 6), we conclude that Assumption 1 should hold with high probability.

Lemma 6 (Hoeffding bound (Hoeffding, 1963))**.** *Let $X_1, \ldots, X_n$ be i.i.d. random variables bounded by the interval $[a, b]$. Then* $\Pr\left(\frac{1}{n}\sum_i X_i - \mathbb{E}[X_1] \geq t\right) \leq \exp\left(-\frac{2nt^2}{(b-a)^2}\right)$.

Figure 5c shows the norm of the stochastic gradients, including both the mean value (solid lines), as well as upper and lower bounds (shaped area). For all settings, the stochastic gradients are always less than 5 before epoch 150 with learning rate 0.1, and less than 15 afterwards with learning rate 0.01. Therefore, multiplying step size with gradient norm, we know SGD step length is always bounded by 0.5.

Notice that the gradient norm gets bigger when we get closer to the final solution (after epoch 150). This further explains why shrinking step size is important.

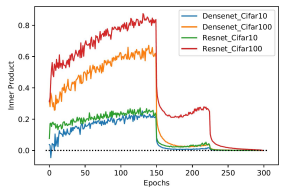
5.3. Loss surface is locally a "slope"

Even with the observation that the whole neighborhood along the SGD trajectory is one point convex with respect to the final solution, there exists a chicken-and-egg concern, as the final target is generated using the SGD trajectory.

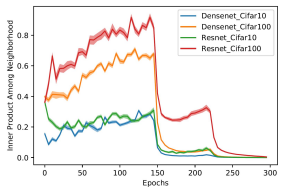
In this subsection, we show that the one point convexity is a pretty "global" property. We were running Resnet and Densenet on Cifar10, but with smaller networks (each with about $10K$ parameters). For each network, if we fix the first 10 epochs, and generate 50 SGD trajectories with different random seeds for 140 epochs and 0.1 learning rate, we get 50 different final solutions (they are pretty far away from each other, with minimum pairwise distance 40). For each network, if we look at the inner product between the negative gradient of **any** epoch of **any** trajectories, and the vector pointing to **any** final solutions, we find that the inner products are almost always positive. (only 0.1% of the inner products are not positive for Densenet, and only 2 out of $343,000$ inner products are not positive for Resnet).

This indicates that the loss surface is "skewed" to the similar direction, and our observation that the whole SGD trajectory is one point convex w.r. to the last point is not a coincidence. Based on our Theorem 1, such loss surface is very friendly to SGD optimization, even with a few exceptional points that

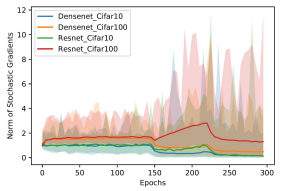
[5] Similar observations were implicitly observed previously (Goodfellow & Vinyals, 2014).



(a) SGD trajectory is locally one point convex.

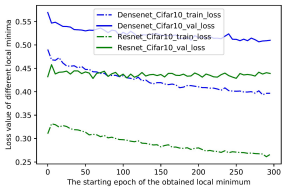


(b) The neighborhood of SGD trajectory is one point convex.



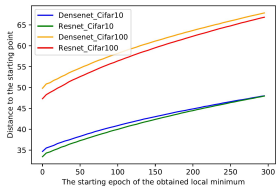
(c) The norm of stochastic gradient

Figure 5: (a). The inner product between the negative gradient and $x_{300} - x_t$ for each epoch $t \geq 5$ is always positive. Every data point is the **minimum** value among 5 trials. (b). Neighborhood of SGD trajectory is also one point convex with respect to x_{300}. (c). Norm of stochastic gradient



(a) Loss value of different local minima on Cifar10

(b) Loss value of different local minima on Cifar100



(c) Distance from the local minima to the initialization

Figure 6: Spectrum of local minima on the loss surface on modern neural networks.

are not one point convex with respect to the final solution.

Notice that in general, it is not possible that all the negative gradients of all points are one point convex with respect to multiple target points. For example, if we take $1D$ interpolation between any two target points, we could easily find points that have negative gradients only pointing to one target point. However, based on our simulation, empirically SGD almost never traverse those regions.

5.4. Spectrum of the local minima

From the previous subsections, we know that the loss surface of neural network has great one point convex properties. It seems that by our Theorem 1, SGD will almost always converge to a few target points (or regions). However, empirically SGD converges to very different target points. In this subsection, we argue that this is because of the learning rate is too big for SGD to converge (Theorem 3). On the other hand, whenever we shrink the learning rate to 0.01, Theorem 1 immediately applies and SGD converges to a local minimum.

In this experiment, we were running smaller version of Resnet and Densenet (each with about $10K$ parameters) on Cifar10 and Cifar100. For each setting, we first train the network with step size 0.1 for 300 epochs, then we pick different epochs as the new starting points for finding nearby local minima using smaller learning rates with additional 150 epochs.

See Figure 6a and Figure 6b. Starting from different epochs, we got local minima with decreasing validation loss and training loss.

To show that these local minima are not from the same region, we also plot the distance of the local minima to the (unique) initialized point. As we can see, as we pick later epochs as the starting points, we get local minima that are farther away from the initialization with better quality (also observed in (Hoffer et al., 2017)).

Furthermore, we observe that for every local minimum, the whole trajectory is always **one point convex** to that local minimum. Therefore, the time for shrinking learning rate decides the quality of the final local minimum. That is, using large step size initially avoids being trapped into a bad local minimum, and whenever we are distant enough from the initialization, we can shrink the step size and converge to a good local minimum (due to one point convexity by Theorem 1).

Acknowledgement

The authors want to thank Zhishen Huang for pointing out a mistake in an early version of this paper, and want to thank Gao Huang, Kilian Weinberger, Jorge Nocedal, Ruoyu Sun, Dylan Foster and Aleksander Madry for helpful discussions. This project is supported by a Microsoft Azure research award and Amazon AWS research award.

References

Allen-Zhu, Z. and Yuan, Y. Improved SVRG for non-strongly-convex or sum-of-non-convex objectives. In *ICML 2016*, volume 48, pp. 1080–1089, 2016.

Chaudhari, P. and Soatto, S. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. *ArXiv e-prints*, October 2017.

Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., and Zecchina, R. Entropy-SGD: Biasing Gradient Descent Into Wide Valleys. *ArXiv e-prints*, November 2016.

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

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp Minima Can Generalize For Deep Nets. *ArXiv e-prints*, March 2017.

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

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points - online stochastic gradient for tensor decomposition. In *COLT 2015*, volume 40, pp. 797–842, 2015.

Goodfellow, I. J. and Vinyals, O. Qualitatively characterizing neural network optimization problems. *CoRR*, abs/1412.6544, 2014.

Hardt, M., Recht, B., and Singer, Y. Train faster, generalize better: Stability of stochastic gradient descent. *ArXiv e-prints*, September 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Hochreiter, S. and Schmidhuber, J. Simplifying neural nets by discovering flat minima. In *Advances in Neural Information Processing Systems 7*, pp. 529–536. MIT Press, 1995.

Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Hoffer, E., Hubara, I., and Soudry, D. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. 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. 1729–1739. Curran Associates, Inc., 2017.

Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L. Densely Connected Convolutional Networks. *ArXiv e-prints*, August 2016.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., and Weinberger, K. Q. Snapshot ensembles: Train 1, get m for free. In *ICLR 2017*, 2017.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. *CoRR*, abs/1703.00887, 2017.

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

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR 2017*, 2017.

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

Li, Y. and Yuan, Y. Convergence analysis of two-layer neural networks with relu activation. In *NIPS 2017*, 2017.

Loshchilov, I. and Hutter, F. SGDR: stochastic gradient descent with restarts. In *ICLR 2017*, 2017.

Mandt, S., Hoffman, M. D., and Blei, D. M. Stochastic Gradient Descent as Approximate Bayesian Inference. *ArXiv e-prints*, April 2017.

Mou, W., Wang, L., Zhai, X., and Zheng, K. Generalization Bounds of SGLD for Non-convex Learning: Two Theoretical Viewpoints. *ArXiv e-prints*, July 2017.

Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Publishing Company, Incorporated, 1 edition, 2014. ISBN 1461346916, 9781461346913.

Patel, V. The Impact of Local Geometry and Batch Size on the Convergence and Divergence of Stochastic Gradient Descent. *ArXiv e-prints*, September 2017.

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

Smith, S. L. and Le, Q. V. A Bayesian Perspective on Generalization and Stochastic Gradient Descent. *ArXiv e-prints*, October 2017.

Sutskever, I., Martens, J., Dahl, G. E., and Hinton, G. E. On the importance of initialization and momentum in deep learning. In *ICML*, pp. 1139–1147, 2013.

Zhang, Y., Liang, P., and Charikar, M. A hitting time analysis of stochastic gradient langevin dynamics. *CoRR*, abs/1702.05575, 2017. URL http://arxiv.org/abs/1702.05575.