Asymmetric Valleys: Beyond Sharp and Flat Local Minima

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite the non-convex nature of their loss functions, deep neural networks are known to generalize well when optimized with stochastic gradient descent (SGD). Recent work conjectures that SGD with proper configuration is able to find wide and flat local minima, which are correlated with good generalization performance. In this paper, we observe that local minima of modern deep networks are more than being flat or sharp. Instead, at a local minimum there exist many asymmetric directions such that the loss increases abruptly along one side, and slowly along the opposite side – we formally define such minima as *asymmetric valleys*. Under mild assumptions, we first prove that for asymmetric valleys, a solution biased towards the flat side generalizes better than the exact empirical minimizer. Then, we show that performing weight averaging along the SGD trajectory implicitly induces such biased solutions. This provides theoretical explanations for a series of intriguing phenomena observed in recent work [25, 5, 51]. Finally, extensive empirical experiments with modern deep networks are conducted to validate our assumptions and analyze the intriguing properties of asymmetric valleys.

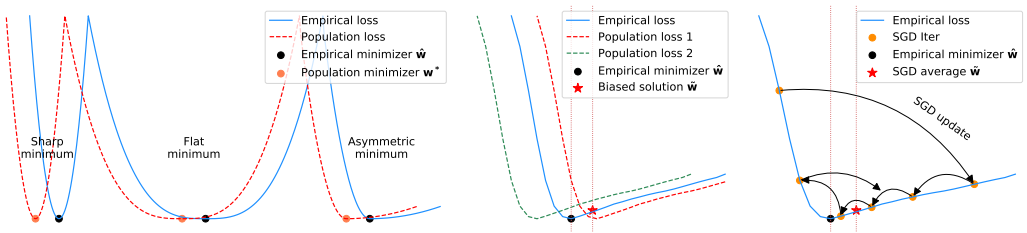
1 Introduction

The loss landscape of neural networks has attracted great research interests in the deep learning community [9, 10, 32, 12, 15, 43, 36]. A deeper understanding of the loss landscape is important for designing better optimization algorithms, and helps to answer the question of when and how a deep network can achieve good generalization performance. One hypothesis that draws attention recently is that the local minima of neural networks can be characterized by their flatness, and it is conjectured that sharp minima tend to generalize worse than the flat ones [32]. A plausible explanation is that a flat minimizer of the training loss can achieve lower generalization error if the test loss is shifted from the training loss due to random perturbations. Figure 1(a) gives an illustration for this argument.

Although being supported by plenty of empirical observations [32, 25, 34], the definition of flatness was recently challenged in [11], which shows that one can construct arbitrarily sharp minima through weight re-parameterization without affecting the generalization performance. Moreover, recent evidences suggest that the minima of modern deep networks are connected with simple paths with low generalization error [12, 13]. It is empirically found that the minima found by large batch training and small batch training are shown to be connected by a path without any "bumps" [43]. In other words, a "sharp minimum" and a "flat minimum" may in fact belong to a same minimum in high dimensional space. Therefore, the notion of flat and sharp minima seems to be an oversimplification of the empirical loss surface.

In this paper, we expand the notion of flat and sharp minima by introducing the concept of *asymmetric valleys*. We observe that the loss surfaces of many neural networks are locally asymmetric. In specific, there exist many directions such that the loss increases abruptly along one side, and grows rather slowly along the opposite side (see Figure 1(b) as an illustration). We formally define this kind of

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.



(a) (b) (c)

Figure 1: **(a)** An illustration of sharp, flat and asymmetric minima. If there exists a shift from empirical loss to population loss, flat minimum is more robust than sharp minimum. **(b)** For asymmetric valleys, if there exists a random shift, the solution $\tilde{\boldsymbol{w}}$ biased towards the flat side is more robust than the minimizer $\hat{\boldsymbol{w}}^*$. **(c)** SGD tends to stay longer on the flat side of asymmetric valleys, therefore SGD averaging automatically produces a bias towards the flat side.

local minima as asymmetric valleys. As we will show in Section 6, asymmetric valleys generate interesting illusions in high dimensional space. For example, located in the same valley shown in Figure 1(b), $\tilde{\boldsymbol{w}}$ may appear to be a wider and flatter minimum than $\hat{\boldsymbol{w}}$ as the former is farther away from the sharp side.

Asymmetric valleys also introduce novel insights to generalization. Folklore says when the exact minimizer is flat, it tends to generalize better as it is more stable with respect to loss surface perturbations [32]. Instead of following this argument, we show that in asymmetric valleys, the solution biased towards the flat side of the valley generalizes better than the exact minimizer, under mild assumptions. This result has at least two interesting implications: (1) converging to *which* local minimum (if there are many) may not be critical for modern deep networks. However, it matters a lot *where* the solution locates; and (2) the solution with lowest *a priori* generalization error is not necessarily the minimizer of the training loss.

Given that a biased solution is preferred for asymmetric valleys, an immediate question is how we can find such solutions in practice. It turns out that simply averaging the weights along the SGD trajectory, naturally leads to the desired solutions. We give a theoretical analysis to support this argument, see Figure 1(c) for an illustration. Our result nicely complements a series of recent empirical observations, which demonstrated that averaged SGD has better performance over plain SGD, for various scenarios including supervised/unsupervised/low-precision training [25, 5, 51].

In addition, we provide empirical analysis to verify our theoretical results and support our claims. For example, we show that asymmetric valleys are indeed prevalent in modern deep networks, and solutions with lower generalization error has bias towards the flat side of the valley.

2 Related Work

Neural network landscape. Neural network landscape analysis is an active and exciting area [16, 34, 15, 40, 49, 10, 43]. For example, [12, 13] observed that essentially all local minima are connected together with simple paths. In [22], cyclic learning rate was used to explore multiple local optima along the training trajectory for model ensembling. There are also appealing visualizations for the neural network landscape [34].

Sharp and flat minima. The discussion of sharp and flat local minima dates back to [20], and recently regains its popularity. For example, Keskar et al. [32] proposed that large batch SGD finds sharp minima, which leads to poor generalization. In [8], an entropy regularized SGD was introduced to explicitly searching for flat minima. It was later pointed out that large batch SGD can yield comparable performance when the learning rate or the number of training iterations are properly set [21, 17, 47, 35, 46, 26]. Moreover, [11] showed that from a given flat minimum, one could construct another minimum with arbitrarily sharp directions but equally good performance. In this paper, we argue that the description of sharp or flat minima is an oversimplification. There may simultaneously exist steep directions, flat directions, and asymmetric directions for the same minimum.

SGD optimization and generalization. As the de facto optimization tool for deep networks, SGD and its variants are extensively studied in the literature. For example, it is shown that they could escape saddle points or sharp local minima under reasonable assumptions [14, 28–30, 50, 1–3, 33]. For convex functions [41] or strongly convex but non-smooth functions [42], SGD averaging is shown to give better convergence rate. In addition, it can also achieve higher generalization performance for

Lipschitz functions in theory [44, 7], or for deep networks in practice [22, 25, 5, 51]. Discussions on the generalization bound of neural networks can be found in [6, 39, 37, 31, 38, 4, 52]. We show that SGD averaging has implicit bias on the flat sides of the minima. Previously, it was shown that SGD has other kinds of implicit bias as well [48, 27, 18].

3 Asymmetric Valleys

In this section, we give a formal definition of asymmetric valley, and empirically show that it is prevalent in the loss landscape of modern deep neural networks.

Preliminaries. In supervised learning, we seek to optimize $\boldsymbol{w}^* \triangleq \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \mathsf{L}(\boldsymbol{w})$, where $\mathsf{L}(\boldsymbol{w}) \triangleq \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[f(\boldsymbol{x};\boldsymbol{w})] \in \mathbb{R}^d \to \mathbb{R}$ is the population loss, $\boldsymbol{x} \in \mathbb{R}^m$ is the input sampled from distribution $\mathcal{D}$, $\boldsymbol{w} \in \mathbb{R}^d$ denotes the model parameter, and $f \in \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}$ is the loss function. Since the data distribution $\mathcal{D}$ is usually unknown, instead of optimizing L directly, we often use SGD to find the empirical risk minimizer $\hat{\boldsymbol{w}}^*$ for a set of random samples $\{\boldsymbol{x}_i\}_{i=1}^n$ from $\mathcal{D}$ (a.k.a. training set): $\hat{\boldsymbol{w}}^* \triangleq \arg\min_{\boldsymbol{w}\in\mathbb{R}^d} \hat{\mathsf{L}}(\boldsymbol{w})$, where $\hat{\mathsf{L}}(\boldsymbol{w}) \triangleq \frac{1}{n}\sum_{i=1}^n f(\boldsymbol{x}_i;\boldsymbol{w})$.

In practice, it is numerically infeasible to find or test the exact local minimizer $\hat{\boldsymbol{w}}^*$. Fortunately, our theoretical results only depend on a good enough solution rather than an exact local minimum, as we will formally define in Section 4. For simplicity, we still refer to such solutions as "local minima", although our analysis generalizes to "solutions found by SGD".

3.1 Definition of asymmetric valley

Before formally introducing asymmetric valleys, we first define asymmetric directions.

Definition 1 (Asymmetric direction)**.** *Given constants $p > 0, \overline{r} > \underline{r} > 0, c > 1$, a direction $\boldsymbol{u}$ is $(\overline{r}, \underline{r}, p, c)$-asymmetric with respect to point $\boldsymbol{w} \in \mathbb{R}^d$ and loss function $\hat{\mathsf{L}}$, if $\nabla_l \hat{\mathsf{L}}(\boldsymbol{w} + l\boldsymbol{u}) < p$, and $\nabla_l \hat{\mathsf{L}}(\boldsymbol{w} - l\boldsymbol{u}) > cp$ for $l \in (\underline{r}, \overline{r})$.*

In the above definition, $\boldsymbol{u} \in \mathbb{R}^d$ is a unit vector representing a direction such that the points on this direction passing $\boldsymbol{w} \in \mathbb{R}^d$ can be written as $\boldsymbol{w} + l\boldsymbol{u}$ for $l \in (-\infty, \infty)$. Intuitively, the loss landscape in the interval $(-\overline{r}, -\underline{r})$ is "sharp", while it is "flat" in the region $(\underline{r}, \overline{r})$. Note that we purposely leave out the region $(-\underline{r}, \underline{r})$ without making further assumptions on it to comply with the fact that the second order derivatives of the loss function is usually continuous. It is impractical to assume the slope of the loss function change abruptly at the point $l = 0$.

As a concrete example, Figure 2 shows an asymmetric direction for a local minimum in ResNet-110 trained on the CIFAR-10 dataset. We verified that it is a $(2.0, 0.6, 0.03, 15)$-asymmetric direction, which means in the region $(-2.0, -0.6) \cup (0.6, 2.0)$ the gradients are asymmetric with a relative ratio of $c = 15$.

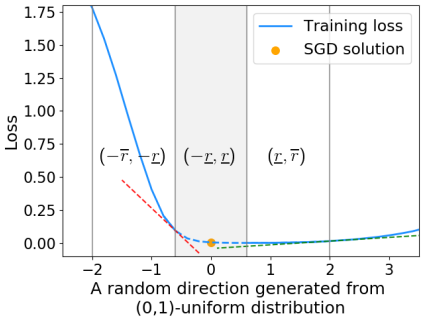


Figure 2: An asymmetric direction of a solution on the loss landscape of ResNet-110 trained on CIFAR-10.

With this Definition 1, we now formally define the *asymmetric valley*[1].

Definition 2 (Asymmetric valley)**.** *Given constants $p, \overline{r} > \underline{r} > 0, c > 1$, a solution $\hat{\boldsymbol{w}}^*$ of $\hat{\mathsf{L}} \in \mathbb{R}^d \to \mathbb{R}$ is a $(\overline{r}, \underline{r}, p, c)$-asymmetric valley, if there exists at least one direction $\boldsymbol{u}$ such that $\boldsymbol{u}$ is $(\overline{r}, \underline{r}, p, c)$-asymmetric with respect to $\hat{\boldsymbol{w}}^*$ and $\hat{\mathsf{L}}$.*

3.2 Asymmetric valleys in deep networks

Empirically, by taking random directions with value $(0, 1)$ in each dimension, we can find an asymmetric direction for a given solution $\boldsymbol{w}^*$ with decent probability. We perform experiments with widely used deep networks, i.e.,ResNet-56, ResNet-110, ResNet-164 [19], VGG-16 [45] and DenseNet-100 [23], on the CIFAR-10, CIFAR-100, SVHN and STL-10 image classification datasets. For each model on each dataset, we conduct 5 independent runs. The results show that we can *always* find asymmetric directions with certain specification $(\overline{r}, \underline{r}, p, c)$ with $c > 2$, which means all the

[1]Here we abuse the name "valley", since $\hat{\boldsymbol{w}}^*$ is essentially a point at the center of a valley.

solutions that SGD found are located in asymmetric valleys. Asymmetric valleys widely exist in both simple and complex models, see Appendix A, Appendix E and Appendix F.

4 Bias and Generalization

As we show in the previous section, in the context of deep learning most local minima in practice are *asymmetric*, i.e., they might be sharp on one direction, but flat on the opposite direction. Therefore, it is interesting to investigate the generalization ability of a solution $\boldsymbol{w}$ in this scenario, which may lead to different results as those obtained under the common symmetric assumption. In this section, we prove that a *biased* solution on the flat side of an asymmetric valley yields lower generalization error than the exact empirical minimizer $\hat{\boldsymbol{w}}^*$ in that valley.

4.1 Theoretical analysis

Before presenting our theorem, we first introduce two mild assumptions. We will show that they empirically hold on modern deep networks in Section 4.2.

The first assumption (Assumption 1) states that there exists a shift between the empirical loss and true population loss. This is a common assumption in the previous works, e.g., [32], but was usually presented in an informal way. Here we define the "shift" in formally. Without loss of generality, we will compare the empirical loss $\hat{\mathsf{L}}$ with $\mathsf{L}' \triangleq \mathsf{L} - \min_{\boldsymbol{w}} \mathsf{L}(\boldsymbol{w}) + \min_{\boldsymbol{w}} \hat{\mathsf{L}}(\boldsymbol{w})$ to remove the "vertical difference" between $\hat{\mathsf{L}}$ and L. Notice that $\min_{\boldsymbol{w}} \mathsf{L}(\boldsymbol{w})$ and $\min_{\boldsymbol{w}} \hat{\mathsf{L}}(\boldsymbol{w})$ are constants and do not affect our generalization guarantee.

Definition 3 ($(\boldsymbol{\delta}, R)$-shift gap)**.** *For $\xi \geq 0$, $\boldsymbol{\delta} \in \mathbb{R}^d$, and fixed functions L and $\hat{\mathsf{L}}$, we define the $(\boldsymbol{\delta}, R)$-shift gap between L and $\hat{\mathsf{L}}$ with respect to a point $\boldsymbol{w}$ as*

$$\xi_{\boldsymbol{\delta}}(\boldsymbol{w}) = \max_{\boldsymbol{v} \in \mathbb{B}(R)} |\mathsf{L}'(\boldsymbol{w} + \boldsymbol{v} + \boldsymbol{\delta}) - \hat{\mathsf{L}}(\boldsymbol{w} + \boldsymbol{v})|$$

where $\mathsf{L}'(\boldsymbol{w}) \triangleq \mathsf{L}(\boldsymbol{w}) - \min_{\boldsymbol{w}} \mathsf{L}(\boldsymbol{w}) + \min_{\boldsymbol{w}} \hat{\mathsf{L}}(\boldsymbol{w})$, and $\mathbb{B}(R)$ is the d-dimensional ball with radius R centered at $\mathbf{0}$.

From the above definition, we know that the two functions match well after the shift $\boldsymbol{\delta}$ if $\xi_{\boldsymbol{\delta}}(\boldsymbol{w})$ is very small. For example, $\xi_{\boldsymbol{\delta}}(\boldsymbol{w}) = 0$ means L is locally identical to $\hat{\mathsf{L}}$ after the shift $\boldsymbol{\delta}$. Since $\hat{\mathsf{L}}$ is computed on a set of random samples from $\mathcal{D}$, the actual shift $\boldsymbol{\delta}$ between $\hat{\mathsf{L}}$ and L is a random variable, ideally with zero expectation[2].

Assumption 1 (Random shift assumption)**.** *For a given population loss L and a random empirical loss $\hat{\mathsf{L}}$, constants $R > 0, \overline{r} \geq \underline{r} > 0, \xi \geq 0$, a vector $\overline{\boldsymbol{\delta}} \in \mathbb{R}^d$ with $\overline{r} \geq \overline{\boldsymbol{\delta}}_i \geq \underline{r}$ for all $i \in [d]$, a minimizer $\hat{\boldsymbol{w}}^*$, we assume that there exists a random variable $\boldsymbol{\delta} \in \mathbb{R}^d$ correlated with $\hat{\mathsf{L}}$ such that $\Pr(\boldsymbol{\delta}_i = \overline{\boldsymbol{\delta}}_i) = \Pr(\boldsymbol{\delta}_i = -\overline{\boldsymbol{\delta}}_i) = \frac{1}{2}$ for all $i \in [d]$, and the $(\boldsymbol{\delta}, R)$-shift gap between L and $\hat{\mathsf{L}}$ with respect to $\hat{\boldsymbol{w}}^*$ is bounded by ξ.*

Clearly, $\boldsymbol{\delta}$ has 2^d possible values for a given shift vector $\overline{\boldsymbol{\delta}}$, each with probability 2^{-d}. Notice that Assumption 1 does not say that the difference between L and $\hat{\mathsf{L}}$ can only be one of the 2^d possible $\boldsymbol{\delta}$. Instead, it says after applying the shift $\boldsymbol{\delta}$, the two functions have bounded L_∞ distance, which is a much milder assumption. It is also worth noting that our Definition 1 can mask out the central interval $(-\underline{r}, \underline{r})$ because we have $\overline{r} \geq \overline{\boldsymbol{\delta}}_i \geq \underline{r}$ in Assumption 1. Therefore, $\underline{r}$ cannot be arbitrarily large, otherwise Assumption 1 does not hold. Our second assumption stated below can be seen as an extension of Definition 2.

Assumption 2 (Locally asymmetric)**.** *For a given population loss $\hat{\mathsf{L}}$, and a minimizer $\hat{\boldsymbol{w}}^*$, there exist orthogonal directions $\boldsymbol{u}^1, \cdots, \boldsymbol{u}^k \in \mathbb{R}^d$ s.t. $\boldsymbol{u}^i$ is $(\overline{r}, \underline{r}, p_i, c_i)$-asymmetric with respect to $\hat{\boldsymbol{w}}^* + \boldsymbol{v} - \langle \boldsymbol{v}, \boldsymbol{u}^i \rangle \boldsymbol{u}^i$ for all $\boldsymbol{v} \in \mathbb{B}(R')$ and $i \in [k]$.*

Assumption 2 states that if $\boldsymbol{u}^i$ is an asymmetric direction at $\hat{\boldsymbol{w}}^*$, then the point $\hat{\boldsymbol{w}}^* + \boldsymbol{v} - \langle \boldsymbol{v}, \boldsymbol{u}^i \rangle \boldsymbol{u}^i$ that deviates from $\hat{\boldsymbol{w}}^*$ along the perpendicular direction of $\boldsymbol{u}^i$, is also asymmetric along the direction of $\boldsymbol{u}^i$. In other words, the *neighborhood* around $\hat{\boldsymbol{w}}^*$ is an asymmetric valley.

[2] It may not be zero, as we are talking about the shift between two loss functions, rather than the difference between empirical/population loss values.

Under the above assumptions, we are ready to state our theorem, which says the empirical minimizer is not necessarily the optimal solution, and a biased solution leads to better generalization. We defer the proof to Appendix B.

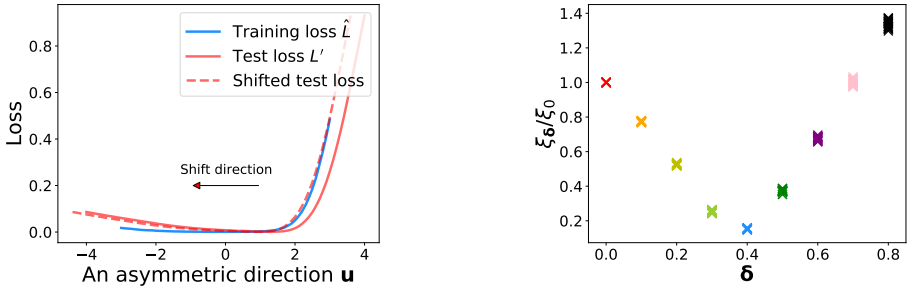
Theorem 1 (Bias leads to better generalization)**.** *For any $\boldsymbol{l} \in \mathbb{R}^k$, if Assumption 1 holds for $R = \|\boldsymbol{l}\|_2$, Assumption 2 holds for $R' = \|\bar{\boldsymbol{\delta}}\|_2 + \|\boldsymbol{l}\|_2$, and $\frac{4\xi}{(c_i-1)p_i} < \boldsymbol{l}_i \leq \max\{\bar{r} - \bar{\boldsymbol{\delta}}_i, \bar{\boldsymbol{\delta}}_i - \underline{r}\}$, then we have*

$$\mathbb{E}_{\boldsymbol{\delta}}\mathsf{L}(\hat{\boldsymbol{w}}^*) - \mathbb{E}_{\boldsymbol{\delta}}\mathsf{L}\left(\hat{\boldsymbol{w}}^* + \sum_{i=1}^{k} \boldsymbol{l}_i \boldsymbol{u}^i\right) \geq \sum_{i=1}^{k} (c_i - 1)\boldsymbol{l}_i p_i/2 - 2k\xi > 0$$

Remark on Theorem 1. It is widely known that the empirical minimizer is usually different from the true optimum. However, in practice it is difficult to know how the training loss shifts from the population loss. Therefore, the best we could it to minimize the empirical loss function (with some regularizers). However, Theorem 1 states that in the asymmetric case, we should pick a biased solution even if the shift is unknown. This insight can be distilled into practical algorithms to achieve better generalization, as we will discuss in Section 5.

4.2 Validating assumptions

We conducted a series of experiments with modern deep networks to show that the two assumptions introduced above are generally valid.



(a) Shfit between L' and $\hat{\mathsf{L}}$ (b) $\xi_{\boldsymbol{\delta}}/\xi_0$ vs different shift $\boldsymbol{\delta}$

Figure 3: Shift exists between empirical loss and population loss for ResNet-110 on CIFAR-10.

Verification of Assumption 1. We show that a shift between L and $\hat{\mathsf{L}}$ is quite common in practice, by taking a ResNet-110 trained on CIFAR-10 as an example. Notice that we use test loss to represent L in practice. Since we could not visualize a shift in a high dimensional space, we randomly sample an asymmetric direction $\boldsymbol{u}$ (more results are shown Appendix C) at the SGD solution $\hat{\boldsymbol{w}}^*$. The blue and red curves shown in Figure 3(a) are obtained by calculating $\hat{\mathsf{L}}(\hat{\boldsymbol{w}}^* + l\boldsymbol{u})$ and $\mathsf{L}'(\hat{\boldsymbol{w}}^* + l\boldsymbol{u})$ for $l \in [-3, 3]$, which correspond to the training and test loss, respectively.

We then try different shift values of $\boldsymbol{\delta}$ to "match" the two curves. As shown in Figure 3(a), after applying a horizontal shift $\boldsymbol{\delta} = 0.4$ to the test loss, the two curves overlap almost perfectly. Quantitatively, we can use the *shift gap* defined in Definition 3 to evaluate how well the two curves match each other after shifting. It turns out that $\xi_{\boldsymbol{\delta}=0.4} = 0.03$, which is much lower than $\xi_{\boldsymbol{\delta}=0} = 0.22$ before shifting ($\boldsymbol{\delta}$ has only one dimension here). In Figure 3(b), we plot $\xi_{\boldsymbol{\delta}}/\xi_0$ as a function of $\boldsymbol{\delta}$. Clearly, there exists a $\boldsymbol{\delta}$ that minimizes this ratio, indicating a good match.

We conducted the same experiments for different directions, models and datasets, and similar observations were made. Please refer to Appendix C for more results.

Verification of Assumption 2. This is a mild assumption that can be verified empirically. For example, we take a SGD solution of ResNet-110 on CIFAR-10 as $\hat{\boldsymbol{w}}^*$, and specify an asymmetric direction $\boldsymbol{u}$ for $\hat{\boldsymbol{w}}^*$. We then randomly sample 100 different local adjustments for $\boldsymbol{v} \in \mathbb{B}(25)$. Based on these adjustments, we present the mean loss curves and standard variance zone on the asymmetric direction $\boldsymbol{u}$ for all the points $\hat{\boldsymbol{w}}^* + \boldsymbol{v} - \langle \boldsymbol{v}, \boldsymbol{u} \rangle \boldsymbol{u}$ in Figure 4. As we can see, the variance of these curves are very small, which means all of them are similar to each other. Moreover, we verified that $\boldsymbol{u}$ is $(4, 2, 0.1, 5.22)$-asymmetric with respect to all neighboring points.

5 Averaging Generates Good Bias

In the previous section, we show that when the loss landscape of a local minimum is asymmetric, a solution with bias towards the flat side of the valley has better generalization performance. One immediate question is that how can we obtain such a solution via practical algorithms? Below we show that it can be achieved by simply taking the average of SGD iterates during the course of training. We first analyze the one dimensional case in Section 5.1, and then extend the analysis to the high dimensional case in Section 5.2.

Note that weight averaging is a classical algorithm in optimization [41], and recently regained its popularity in the context of deep learning [25, 5, 51]. Our following analysis can be viewed as a theoretical justification of recent algorithms that based on SGD iterates averaging.

5.1 One dimensional case

For asymmetric functions, as long as the learning rate is not too small, SGD will oscillate between the flat side and the sharp side. Below we focus on one round of oscillation, and show that the average of the iterates in each round has a bias on the flat side. Consequently, by aggregating all rounds of oscillation, averaging SGD iterates leads to a bias as well.

For each individual round i, we assume that it starts from the iteration when SGD goes from sharp side to flat side (denoted as w_0^i), and ends at the iteration exactly before the iteration that SGD goes from sharp side to flat side again (denoted as $w_{T_i}^i$). Here T_i denotes the number of iterations in the i-th rounds. The average iterate in the i-th round can be written as $\bar{w} \triangleq \frac{1}{T_i}\sum_{j=0}^{T_i} w_j^i$. For notational simplicity, we will omit the super script i on w_j^i.

The following theorem shows that the expectation of the average has bias on the flat side. To get a formal lower bound on $\bar{w}$, we consider the asymmetric case where $\underline{r} = 0$, and also assume lower bounds for the gradients on the function. We defer the proof to Appendix D.

Theorem 2 (SGD averaging generates a bias)**.** *Assume that a local minimizer $w^* = 0$ is a $(r, 0, a_+, c)$-asymmetric valley, where $b_- \le \nabla \mathsf{L}(w) \le a_- < 0$ for $w < 0$, and $0 < b_+ \le \nabla \mathsf{L}(w) \le a_+$ for $w \ge 0$. Assume $-a_- = ca_+$ for a large constant c, and $\frac{-(b_- - \nu)}{b_+} = c' < \frac{e^{c/3}}{6}$. The SGD updating rule is $w_{t+1} = w_t - \eta(\nabla L(w) + \omega_t)$ where ω_t is the noise and $|\omega_t| < \nu$, and assume $\nu \le a_+$. Then we have*

$$\mathbb{E}[\bar{w}] > c_0 > 0,$$

where c_0 is a constant that only depends on η, a_+, a_-, b_+, b_- and ν.

Theorem 2 can be intuitively explained by Figure 5. If we run SGD on this one dimensional function, it will stay at the flat side for more iterations as the magnitude of the gradient on this side is much smaller. Therefore, the average of the locations is biased towards the flat side.

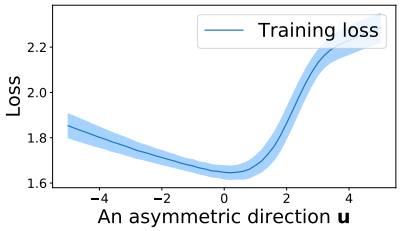


Figure 4: Training loss mean and variance for the neighborhood of $\hat{\boldsymbol{w}}^*$ at the direction of $\boldsymbol{u}$.

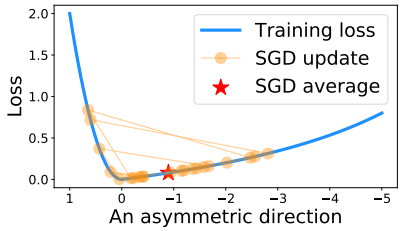


Figure 5: SGD iterates and their average on an asymmetric function.

5.2 High dimensional case

For high dimensional functions, the analysis on averaging SGD iterates would be more complicated compared to that given in the previous subsection. However, if we only care about the bias on a specific direction $\boldsymbol{u}$, we could directly apply Theorem 2 with one additional assumption. Specifically, if the projections of the loss function onto $\boldsymbol{u}$ along the SGD trajectory satisfy the assumptions in Theorem 2, i.e., being asymmetric and the gradient on both sides have upper and lower bounds, then the claim of Theorem 2 directly applies. This is because only the gradient along the direction $\boldsymbol{u}$ will affect the SGD trajectory projected onto $\boldsymbol{u}$, and we could safely omit all other directions.

We find that this assumption holds empirically. For a given SGD solution, we fix a random asymmetric direction $\boldsymbol{u} \in \mathbb{R}^d$, and sample the loss surface on direction $\boldsymbol{u}$ that passes the t-th epoch of SGD trajectory (denoted as $\boldsymbol{w_t}$), i.e., evaluate $\hat{\mathrm{L}}(\boldsymbol{w_t} + l\boldsymbol{u})$, for $0 \leq t \leq 200$ and $l \in [-15, 15]$. As shown in the Figure 6, after the first 40 epochs, the projected loss surfaces becomes relatively stable. Therefore, we can directly apply Theorem 2 to the direction $\boldsymbol{u}$.

As we will see in Section 6.1, compared with SGD solutions, SGD averaging indeed creates bias along different asymmetric directions, as predicted by our theory.

6 Experimental Observations

In this section, we empirically show that asymmetric valleys create interesting illusions when visualizing high dimensional loss landscape in low dimensional space. In addition, as a refinement of judging the generalization performance by the sharpness/flatness of a local minimum, we show that *where* the solution locates at a local minimum basin is important. We also find that batch normalization [24] seems to be a major cause for asymmetric valleys in deep networks, but the results are deferred to Appendix H due to space limit.

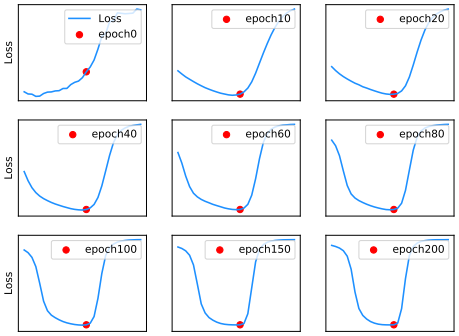


Figure 6: Projection of the training loss surface onto an asymmetric direction $\boldsymbol{u}$

6.1 Experiments with weight averaging

Recently, Izmailov et al. [25] proposed the stochastic weight averaging (SWA) algorithm, which explicitly takes the average of SGD iterates to achieve better generalization. Inspired by their observation that "SWA leads to solutions corresponding to wider optima than SGD", we provide a more refined explanation in this subsection. That is, averaging weights leads to "biased" solutions in an asymmetric valley, which correspond to better generalization.

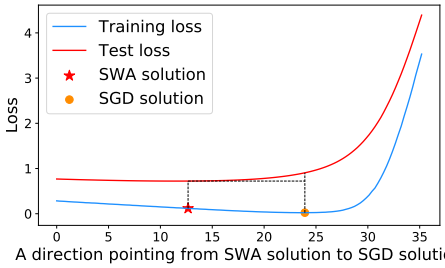


Figure 7: SWA solution and SGD solution interpolation (ResNet-164 on CIFAR-100)

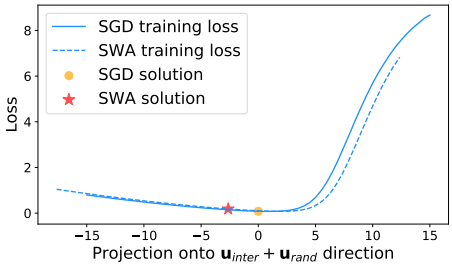


Figure 8: The average of SGD has a bias on flat side (ResNet-110 on CIFAR-100)

Specifically, we run the SWA algorithm (with deceasing learning rate) with popular deep networks, including ResNet-56, ResNet-110, ResNet-164, VGG-16, and DenseNet-100, on various datasets including CIFAR-10, CIFAR-100, SVHN and STL-10, following the configurations in [25]. Then we run SGD with small learning rate *from the SWA solutions* to find a solution located in the same basin (denoted as SGD).

In Figure 7, We draw an interpolation between the solutions obtained by SWA and SGD[3]. One can observe that there is no "bump" between these two solutions, meaning they are located in the same basin. Clearly, the SWA solution is biased towards the flat side, which verifies our theoretical analysis in Section 5. Further, we notice that although the biased SWA solution has higher training loss than the solution found by SGD, it indeed yields lower test loss. This verifies our analysis in Section 4. Similar observations are made on other networks and other datasets, which we present in Appendix E.

Table 1: Training and test accuracy on CIFAR-100.

Network	CIFAR-100	
	train	test
ResNet-110-SWA	94.98%	78.94%
ResNet-110-SGD	97.52%	78.29%
ResNet-164-SWA	97.48%	80.69%
ResNet-164-SGD	99.12%	76.56%

[3]Izmailov et al. [25] have done a similar experiment.

To further support our claim, we list our result in Table 1, from which we can observe that SGD solutions always have higher training accuracy, but worse test accuracy, compared to SWA solutions. This supports our claim in Theorem 1, which states that a bias towards the flat sides of asymmetric valleys could help improve generalization, although it yields higher training error.

Verifying Theorem 2. We further verify that averaging SGD solutions creates a bias towards the flat side in expectation for many other asymmetric directions, not just for the specific direction we discussed above.

We take a ResNet-110 trained on CIFAR-100 as an example. Denote $\boldsymbol{u}_{inter}$ as the unit vector pointing from the SGD solution to the SWA solution, $\boldsymbol{u}_{rand}$ as another unit random direction, and the direction $\boldsymbol{u}_{inter} + \boldsymbol{u}_{rand}$ is used to explore the asymmetric landscape.

The results are shown in Figure 8, from which we can observe that SWA has a bias on the flat side compared with the SGD solution. We create 10 different random vectors for each network and each dataset, and similar observations can be made (see more examples in Appendix F).

Batch size effect In addition to SWA algorithm, we also observe similar trend when training with different batch sizes. The results are deferred to Appendix G.

6.2 Illusions created by asymmetric valleys

We further point out that visualizing the "width" of a given solution $\boldsymbol{w}$ in a low-dimensional space may lead to illusive results. For example, one visualization technique used in [25] is to show how the loss changes along many random directions $\boldsymbol{v}_i$'s drawn from the d-dimensional Gaussian distribution.

We take the large batch and small batch solutions from the previous subsection as an example. Figure 9 visualizes the "width" of the two solutions using the method described above. From the figure, one may draw the conclusion that small batch training leads to a wider minimum compared to large batch training. However, these two solutions are in fact from the *same* basin (see the discussion in Appendix G). In other words, the loss curvature near the two solutions looks different because they are located at *different locations* in a same asymmetric valley, instead of being located at *different local minima*. Similar observation holds for SWA and SGD solutions, see Figure 10[4] .

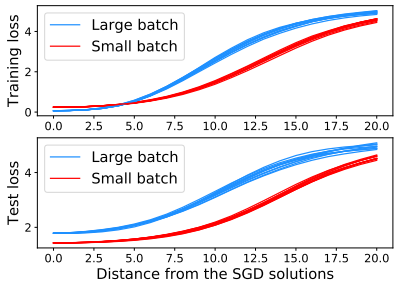


Figure 9: Random ray of large batch and small batch solution.

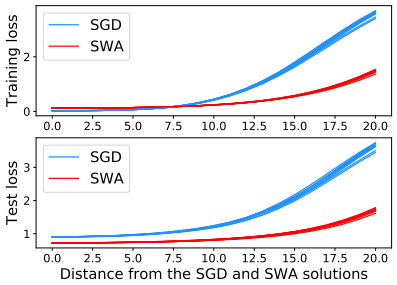


Figure 10: Random ray of SGD and SWA solution

7 Conclusion

In this paper, we introduced the notion of asymmetric valley to characterize the loss landscape of deep networks, expanding the current research that simply categorizes local minima by sharpness/flatness. This notion allowed us to analyze and understand the geometry of loss landscape from a new perspective. For example, based on a formal definition of asymmetric valley, we showed that a biased solution lying on the flat side of the valley generalizes better than the exact empirical minimizer. Further, it is proved that by averaging the weights obtained along the SGD trajectory naturally leads to such biased solution. We also conducted extensive experiments with state-of-the-art deep models to analyze the properties of asymmetric valleys. It is showed that due to the existence of asymmetric valleys, intriguing illustions can be created when visualizing high dimensional loss surface in the 1D space. We hope this work will deepen our understanding on the loss landscape of deep neural networks, and inspire new theories and algorithms that further improve generalization.

[4] Similar observations were made by Izmailov et al. [25] as well.

References

[1] Allen-Zhu, Z. How to make the gradients small stochastically: Even faster convex and nonconvex SGD. In *NeurIPS*, pp. 1165–1175, 2018.

[2] Allen-Zhu, Z. Natasha 2: Faster non-convex optimization than SGD. In *NeurIPS*, pp. 2680–2691, 2018.

[3] Allen-Zhu, Z. and Li, Y. NEON2: finding local minima via first-order oracles. In *NeurIPS*, pp. 3720–3730, 2018.

[4] Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 254–263. PMLR, 2018.

[5] Athiwaratkun, B., Finzi, M., Izmailov, P., and Wilson, A. G. There are many consistent explanations of unlabeled data: Why you should average. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=rkgKBhA5Y7`.

[6] Bartlett, P. L., Foster, D. J., and Telgarsky, M. J. Spectrally-normalized margin bounds for neural networks. In *NIPS*, pp. 6241–6250, 2017.

[7] Cesa-Bianchi, N., Conconi, A., and Gentile, C. On the generalization ability of on-line learning algorithms. In *NIPS*, pp. 359–366. MIT Press, 2001.

[8] Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J. T., Sagun, L., and Zecchina, R. Entropy-sgd: Biasing gradient descent into wide valleys. In *ICLR*. OpenReview.net, 2017.

[9] Choromanska, A., LeCun, Y., and Arous, G. B. Open problem: The landscape of the loss surfaces of multilayer networks. In *COLT*, volume 40 of *JMLR Workshop and Conference Proceedings*, pp. 1756–1760. JMLR.org, 2015.

[10] Cooper, Y. The loss landscape of overparameterized neural networks. *CoRR*, abs/1804.10200, 2018.

[11] Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1019–1028. PMLR, 2017.

[12] Dräxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. A. Essentially no barriers in neural network energy landscape. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1308–1317. PMLR, 2018.

[13] Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *NeurIPS*, pp. 8803–8812, 2018.

[14] Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points - online stochastic gradient for tensor decomposition. In *COLT*, volume 40 of *JMLR Workshop and Conference Proceedings*, pp. 797–842. JMLR.org, 2015.

[15] Ge, R., Lee, J. D., and Ma, T. Learning one-hidden-layer neural networks with landscape design. In *ICLR*. OpenReview.net, 2018.

[16] Goodfellow, I. J. and Vinyals, O. Qualitatively characterizing neural network optimization problems. In *ICLR*, 2015.

[17] Goyal, P., Dollár, P., Girshick, R. B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch SGD: training imagenet in 1 hour. *CoRR*, abs/1706.02677, 2017.

[18] Gunasekar, S., Lee, J., Soudry, D., and Srebro, N. Characterizing implicit bias in terms of optimization geometry. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1827–1836. PMLR, 2018.

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

[20] Hochreiter, S. and Schmidhuber, J. Simplifying neural nets by discovering flat minima. In *NIPS*, pp. 529–536. MIT Press, 1994.

[21] Hoffer, E., Hubara, I., and Soudry, D. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *NIPS*, pp. 1729–1739, 2017.

[22] 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*. OpenReview.net, 2017.

[23] Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. Densely connected convolutional networks. In *CVPR*, pp. 2261–2269. IEEE Computer Society, 2017.

[24] Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015.

[25] Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D. P., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. In *UAI*, pp. 876–885. AUAI Press, 2018.

[26] Jastrzebski, S., Kenton, Z., Arpit, D., Ballas, N., Fischer, A., Bengio, Y., and Storkey, A. J. Three factors influencing minima in SGD. *CoRR*, abs/1711.04623, 2017.

[27] Ji, Z. and Telgarsky, M. Risk and parameter convergence of logistic regression. *CoRR*, abs/1803.07300, 2018.

[28] Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1724–1732. PMLR, 2017.

[29] Jin, C., Liu, L. T., Ge, R., and Jordan, M. I. On the local minima of the empirical risk. In *NeurIPS*, pp. 4901–4910, 2018.

[30] Jin, C., Netrapalli, P., and Jordan, M. I. Accelerated gradient descent escapes saddle points faster than gradient descent. In *COLT*, volume 75 of *Proceedings of Machine Learning Research*, pp. 1042–1085. PMLR, 2018.

[31] Kawaguchi, K., Kaelbling, L. P., and Bengio, Y. Generalization in deep learning. *CoRR*, abs/1710.05468, 2017.

[32] 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*. OpenReview.net, 2017.

[33] Kleinberg, R., Li, Y., and Yuan, Y. An alternative view: When does SGD escape local minima? In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2703–2712. PMLR, 2018.

[34] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. In *NeurIPS*, pp. 6391–6401, 2018.

[35] Masters, D. and Luschi, C. Revisiting small batch training for deep neural networks. *CoRR*, abs/1804.07612, 2018.

[36] Mehta, D., Chen, T., Tang, T., and Hauenstein, J. D. The loss surface of deep linear networks viewed through the algebraic geometry lens. *CoRR*, abs/1810.07716, 2018.

[37] Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. In *NIPS*, pp. 5949–5958, 2017.

[38] Neyshabur, B., Bhojanapalli, S., and Srebro, N. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. In *ICLR*. OpenReview.net, 2018.

[39] Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. Towards understanding the role of over-parametrization in generalization of neural networks. *CoRR*, abs/1805.12076, 2018.

[40] Pennington, J. and Bahri, Y. Geometry of neural network loss surfaces via random matrix theory. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2798–2806. PMLR, 2017.

[41] Polyak, B. T. and Juditsky, A. B. Acceleration of stochastic approximation by averaging. *SIAM J. Control Optim.*, 30(4):838–855, July 1992. ISSN 0363-0129.

[42] Rakhlin, A., Shamir, O., and Sridharan, K. Making gradient descent optimal for strongly convex stochastic optimization. In *ICML*. icml.cc / Omnipress, 2012.

[43] Sagun, L., Evci, U., Güney, V. U., Dauphin, Y., and Bottou, L. Empirical analysis of the hessian of over-parametrized neural networks. In *ICLR (Workshop)*. OpenReview.net, 2018.

[44] Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. Stochastic convex optimization. In *COLT*, 2009.

[45] Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[46] Smith, S. L. and Le, Q. V. A bayesian perspective on generalization and stochastic gradient descent. In *ICLR*. OpenReview.net, 2018.

[47] Smith, S. L., Kindermans, P., Ying, C., and Le, Q. V. Don't decay the learning rate, increase the batch size. In *ICLR*. OpenReview.net, 2018.

[48] Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19:70:1–70:57, 2018.

[49] Wu, L., Zhu, Z., and E, W. Towards understanding generalization of deep learning: Perspective of loss landscapes. *CoRR*, abs/1706.10239, 2017.

[50] Xu, Y., Rong, J., and Yang, T. First-order stochastic algorithms for escaping from saddle points in almost linear time. In *NeurIPS*, pp. 5535–5545, 2018.

[51] Yang, G., Zhang, T., Kirichenko, P., Bai, J., Wilson, A. G., and Sa, C. D. Swalp: Stochastic weight averaging in low precision training. In *ICML*, 2019.

[52] Zhou, W., Veitch, V., Austern, M., Adams, R. P., and Orbanz, P. Non-vacuous generalization bounds at the imagenet scale: a PAC-bayesian compression approach. In *International Conference on Learning Representations*, 2019.

A Additional Figures for Section 3.2: Asymmetric Directions

To show that asymmetric valley can be commonly observed, we conduct experiments ranging from the simplest network to modern deep neural networks.

A simple case First, we will show that asymmetric valley can be observed on a simple MLP (one hidden layer with 10 hidden neurons) on a logistic regression task in Figure 11

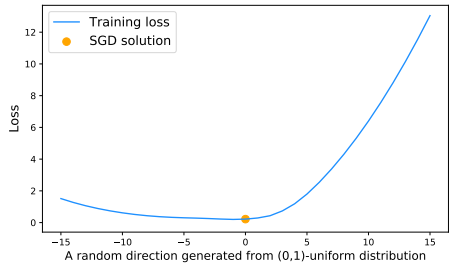


Figure 11: Asymmetric direction for a solution of MLP on logistic regression. $(\overline{r}, \underline{r}, p, c) = (10.0, 5.0, 0.11, 6.0)$.

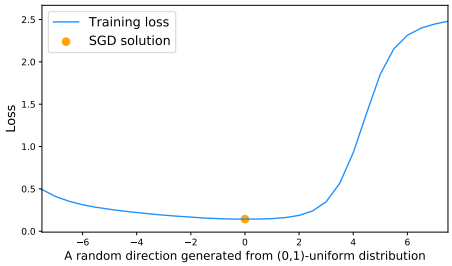


Figure 12: Asymmetric direction for a solution of ResNet-164 on CIFAR-10. $(\overline{r}, \underline{r}, p, c) = (4.0, 2.5, 0.033, 4.8)$.

Other datasets and networks See Figure 12, Figure 13, Figure 14, Figure 15, and Figure 16.

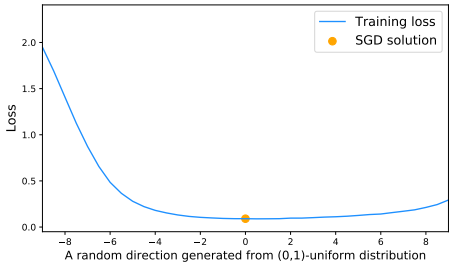


Figure 13: Asymmetric direction for a solution of DenseNet-100 on CIFAR-10. $(\overline{r}, \underline{r}, p, c) = (7.0, 5.0, 0.030, 4.8)$.

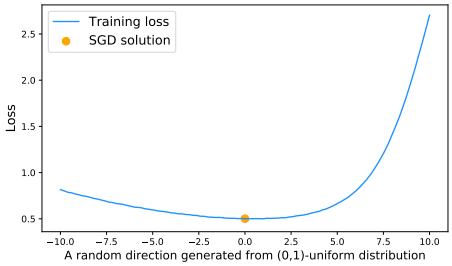


Figure 14: Asymmetric direction for a solution of ResNet-110 on CIFAR-100. $(\overline{r}, \underline{r}, p, c) = (7.0, 5.0, 0.039, 2.7)$.

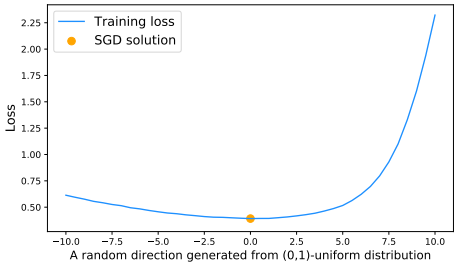


Figure 15: Asymmetric direction for a solution of ResNet-164 on CIFAR-100. $(\overline{r}, \underline{r}, p, c) = (7.0, 5.0, 0.031, 2.5)$.

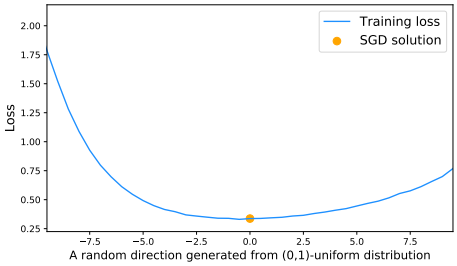


Figure 16: Asymmetric direction for a solution of DenseNet-100 on CIFAR-100. $(\overline{r}, \underline{r}, p, c) = (8.5, 6.5, 0.087, 2.1)$.

B Proof for Theorem 1

Proof. Since $\boldsymbol{\delta}$ has 2^d possible value for a given $\bar{\boldsymbol{\delta}}$, we can use an integer $j \in \{0, \cdots, 2^d - 1\}$ to represent each value. When writing j in binary, its i-th digit represents whether $\boldsymbol{\delta}_i = \bar{\boldsymbol{\delta}}_i$ (equal to 1) or $\boldsymbol{\delta}_i = -\bar{\boldsymbol{\delta}}_i$ (equal to 0). We use $j \wedge 2^i$ to represent the bitwise AND operator between j and 2^i, which equals 0 if the i-th digit of j is 0.

To prove our theorem, it suffices to show that for any $i \in [k]$,

$$\mathbb{E}_{\boldsymbol{\delta}} \mathsf{L}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0}\right) - \mathbb{E}_{\boldsymbol{\delta}} \mathsf{L}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0}\right) \geq (c_i - 1)\boldsymbol{l}_i p_i/2 - 2\xi > 0 \tag{1}$$

If (1) is true, it suffices to take summation over i on both sides, and we will get our conclusion. Therefore, below we will prove (1).

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\delta}} \mathsf{L}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0}\right) - \min_{\boldsymbol{w}} \mathsf{L}(\boldsymbol{w}) + \min_{\boldsymbol{w}} \hat{\mathsf{L}}(\boldsymbol{w}) \\
=&\mathbb{E}_{\boldsymbol{\delta}} \mathsf{L}'\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0}\right) \overset{①}{\geq} \frac{1}{2^d} \sum_{j=0}^{2^d-1} \hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j\right) - \xi \\
=&\frac{1}{2^d} \sum_{\substack{j=0 \\ j \wedge 2^i = 0}}^{2^d-1} \left[\hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j\right) + \hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i}\right)\right] - \xi
\end{aligned} \tag{2}$$

Where ① holds by Assumption 1, and the fact that $\| \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} \|_2 \leq \|\boldsymbol{l}\|_2 = R$. For every j s.t. $j \wedge 2^i = 0$,

$$\begin{aligned}
&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j \\
=&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j + \langle \boldsymbol{\delta}^j, \boldsymbol{u}^i \rangle \boldsymbol{u}^i - \langle \boldsymbol{\delta}^j, \boldsymbol{u}^i \rangle \boldsymbol{u}^i \\
=&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j - \bar{\boldsymbol{\delta}}_i \boldsymbol{u}^i - \langle \boldsymbol{\delta}^j, \boldsymbol{u}^i \rangle \boldsymbol{u}^i + \boldsymbol{l}_i \boldsymbol{u}^i \\
=&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j - \langle \boldsymbol{\delta}^j, \boldsymbol{u}^i \rangle \boldsymbol{u}^i + (\boldsymbol{l}_i - \bar{\boldsymbol{\delta}}_i) \boldsymbol{u}^i
\end{aligned}$$

Since $\| \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} \|_2 \leq \|\boldsymbol{l}\|_2$, $\|\delta^j\|_2 = \|\bar{\boldsymbol{\delta}}\|_2$, we know that $\forall j, \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j \in \mathbb{B}(R')$. By Assumption 2, for every $i \in [k]$, $\boldsymbol{u}^i$ is asymmetric with respect to $\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j - \langle \boldsymbol{\delta}^j, \boldsymbol{u}^i \rangle \boldsymbol{u}^i$. Since $\boldsymbol{l}_i \leq \bar{\boldsymbol{\delta}}_i - \underline{r}$, we have $\boldsymbol{l}_i - \bar{\boldsymbol{\delta}}_i < -\underline{r}$. By the definition of asymmetric direction, we know

$$\hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j\right) \geq \hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j\right) + c_i \boldsymbol{l}_i p_i \tag{3}$$

Similarly,

$$\begin{aligned}
&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i} \\
=&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i} + \langle \boldsymbol{\delta}^{j+2^i}, \boldsymbol{u}^i \rangle \boldsymbol{u}^i - \langle \boldsymbol{\delta}^{j+2^i}, \boldsymbol{u}^i \rangle \boldsymbol{u}^i + \boldsymbol{l}_i \boldsymbol{u}^i \\
=&\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i} - \langle \boldsymbol{\delta}^{j+2^i}, \boldsymbol{u}^i \rangle \boldsymbol{u}^i + (\bar{\boldsymbol{\delta}}_i + \boldsymbol{l}_i) \boldsymbol{u}^i
\end{aligned}$$

Since $\boldsymbol{l}_i \leq r - \bar{\boldsymbol{\delta}}_i$, we have $\bar{\boldsymbol{\delta}}_i + \boldsymbol{l}_i \leq r$. Therefore,

$$\hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i}\right) \geq \hat{\mathsf{L}}\left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0}\boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i}\right) - \boldsymbol{l}_i p_i \tag{4}$$

Combining (3) and (4), we have,

$$
\begin{aligned}
(2) \geq & \frac{1}{2^d} \sum_{\substack{j=0 \\ j \wedge 2^i = 0}}^{2^d-1} \left[\hat{\mathsf{L}} \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j \right) + c_i \boldsymbol{l}_i p_i + \hat{\mathsf{L}} \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} + \boldsymbol{\delta}^{j+2^i} \right) - \boldsymbol{l}_i p_i \right] - \xi \\
= & \frac{1}{2^d} \sum_{j=0}^{2^d-1} \left[\hat{\mathsf{L}} \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_i \boldsymbol{u}_{i_0} + \boldsymbol{\delta}^j \right) \right] + (c_i - 1) \boldsymbol{l}_i p_i / 2 - \xi \\
\overset{②}{\geq} & \mathbb{E}_{\boldsymbol{\delta}} \mathsf{L}' \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} \right) + (c_i - 1) \boldsymbol{l}_i p_i / 2 - 2\xi \\
= & \mathbb{E}_{\boldsymbol{\delta}} \mathsf{L} \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} \right) - \min_{\boldsymbol{w}} \mathsf{L}(\boldsymbol{w}) + \min_{\boldsymbol{w}} \hat{\mathsf{L}}(\boldsymbol{w}) + (c_i - 1) \boldsymbol{l}_i p_i / 2 - 2\xi
\end{aligned}
$$

Where ② holds by Assumption 1 and the fact that $\| \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} \|_2 \leq \| \boldsymbol{l} \|_2 = R$. That means,

$$
\mathbb{E}_{\boldsymbol{\delta}} \mathsf{L} \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i-1} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} \right) \geq \mathbb{E}_{\boldsymbol{\delta}} \mathsf{L} \left(\hat{\boldsymbol{w}}^* + \sum_{i_0=1}^{i} \boldsymbol{l}_{i_0} \boldsymbol{u}_{i_0} \right) + (c_i - 1) \boldsymbol{l}_i p_i / 2 - 2\xi > 0
$$

Where the last inequality holds as $\boldsymbol{l}_i > \frac{4\xi}{(c_i - 1) p_i}$.

□

C Additional Figures for Section 4.2: Shift Exists Empirically

See Figure 17, Figure 18, and Figure 19.

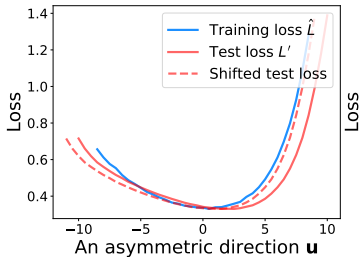


Figure 17: Shift on asymmetric direction (DenseNet-100 on CIFAR-100), $\xi_{\boldsymbol{\delta}=1} = 0.119$, $\xi_{\boldsymbol{\delta}=0} = 0.439$

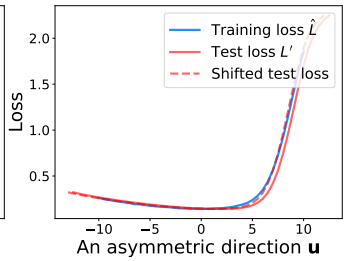


Figure 18: Shift on asymmetric direction (ResNet-164 on CIFAR-10), $\xi_{\boldsymbol{\delta}=0.5} = 0.0699$, $\xi_{\boldsymbol{\delta}=0} = 0.189$

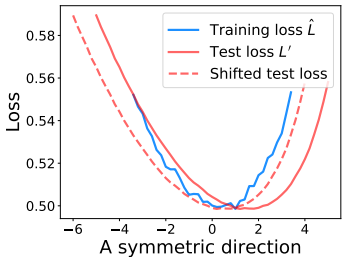


Figure 19: Shift on symmetric direction (ResNet-110 on CIFAR-100), $\xi_{\boldsymbol{\delta}=1} = 0.0197$, $\xi_{\boldsymbol{\delta}=0} = 0.0431$

D Proof for Theorem 2

To prove Theorem 2, we will need the following concentration bound.

Lemma 3 (Azuma's inequality)**.** *Let $X_1, X_2, X_3, ...X_n$ be independent random variables satisfying $|X_i - \mathbb{E}[X_i]| \leq c_i$, for $1 \leq i \leq n$. We have the following bound for $X = \sum_{i=1}^{n} X_i$:*

$$
\Pr(|X - \mathbb{E}(X)| \geq \lambda) \leq 2e^{-\frac{\lambda^2}{2 \sum_{i=1}^{n} c_i^2}}
$$

Let $p_{\min} \triangleq -\eta(a_- + a_+ + 2\nu)$, $p_{\max} \triangleq -\eta(b_- - \nu)$. Since $-a_- = c a_+$, we know $p_{\min} > (c-1)\eta a_+ - 2\eta\nu$. First, we have the following bounds on the first step w_0.

Lemma 4. *For every $i \in [h]$, $w_0 \in [p_{\min}, p_{\max}]$.*

Proof. Since w_0 is the first step that SGD jumps from the flat side to the sharp side, denote the previous location as $w_{-1} < 0$. Since w_{-1} is at the sharp side, we know that the gradient is $\nabla \mathsf{L}(w_{-1}) \leq a_-$. Therefore, we have

$$w_0 = w_{-1} - \eta(\nabla \mathsf{L}(w_{-1}) + \omega_{-1})$$

Where ω_{-1} is the noise bounded by ν.

At the time when SGD jump from the flat side to sharp side, denote the target position as w'_{-1}. We know that $w'_{-1} \in [-\eta(a_+ + \nu), 0]$. Since the gradient on the sharp side is at most a_-, we know the next step is lower bounded by $-\eta(a_+ + 2\nu + a_-) = p_{\min} > 0$. In other words, SGD stays at the sharp side for only 1 iterations (this matches with our empirical observation, see e.g. Figure 5).

That means, the bound on w'_{-1} can be applied to w_{-1} as well, because they are the same iterate. By applying the upper and lower bound on $\nabla \mathsf{L}(w_{-1})$, we get:

$$w_0 \geq -\eta(a_+ + \nu) - \eta(a_- + \nu) = p_{\min}$$

and also

$$w_0 \leq 0 - \eta(b_- - \nu) = p_{\max}$$

□

Below we first define $T_{\min} \triangleq \left(\frac{-\sqrt{2}\nu \log^{1/2}(2\tau) + \sqrt{2\nu^2 \log(2\tau) - 4a_+(a_- + a_+ + 2\nu)}}{2a_+} \right)^2$, where τ is a constant with value to be set later. $T_{\min}$ satisfies the following inequality.

Lemma 5. $\forall t \leq T_{\min}, p_{\min} - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau) \geq 0.$

Proof. By the definition of $p_{\min}$, we have

$$\begin{aligned}
&- \eta(a_- + a_+ + 2\nu) - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau) \geq 0 \\
\Leftarrow &(a_- + a_+ + 2\nu) + ta_+ + \sqrt{2t}\nu \log^{1/2}(2\tau) \leq 0 \\
\Leftarrow &(a_- + a_+ + 2\nu) + \Delta^2 a_+ + \sqrt{2}\Delta r \log^{1/2}(2\tau) \leq 0 \quad (\Delta \triangleq \sqrt{\mathrm{t}}) \\
\Leftarrow &\Delta \in \left[0, \frac{-\sqrt{2}\nu \log^{1/2}(2\tau) + \sqrt{2\nu^2 \log(2\tau) - 4a_+(a_- + a_+ + 2\nu)}}{2a_+} \right] \\
\Leftarrow &t \leq \left(\frac{-\sqrt{2}\nu \log^{1/2}(2\tau) + \sqrt{2\nu^2 \log(2\tau) - 4a_+(a_- + a_+ + 2\nu)}}{2a_+} \right)^2
\end{aligned}$$

□

Now, we have the following theorem that says with decent probability, the minimum number of iterates on the flat side in i-th round is at least $T_{\min}$.

Theorem 6. *If we start at $w_0 \geq p_{\min}$, for every fixed $\tau > T_{\min}$, with probability at least $1 - \frac{T_{\min}}{\tau}$, we have $\forall t \leq T_{\min}, w_t > w_0 - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau) \geq 0$.*

Proof. Define filtration $\mathcal{F}_t = \sigma\{\omega_0, \cdots, \omega_{t-1}\}$, where $\sigma\{\cdot\}$ denotes the sigma field. Define the event $\mathfrak{E}_T = \{\forall t \leq T, w_t > w_0 - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau)\}$ and define $G_t = w_0 - w_t - t\eta a_+ + M$, where $M \triangleq (T_{\min} + 1)(w_0 + \nu + 2\eta a_+)$. Since we only consider the case $t \leq T_{\min}$, we have

$$G_t = w_0 - w_t - t\eta a_+ + (T_{\min} + 1)(w_0 + \nu + 2\eta a_+) > w_0 - w_t - t\eta a_+ + w_t + t\eta a_+ > 0$$

Therefore, G_t is always positive. By SGD updating rule, we have

$$\begin{aligned}
&\mathbb{E}[G_{t+1}\mathbb{1}_{\mathfrak{E}_t}|\mathcal{F}_t] = \mathbb{E}[(w_0 - w_{t+1} - (t+1)\eta a_+ + M)\mathbb{1}_{\mathfrak{E}_t}|\mathcal{F}_t] \\
\leq &\mathbb{E}[(w_0 - w_t + \eta\omega_t - t\eta a_+ + M)\mathbb{1}_{\mathfrak{E}_t}|\mathcal{F}_t] = w_0 - w_t - t\eta a_+ + M = G_t\mathbb{1}_{\mathfrak{E}_t}
\end{aligned} \tag{5}$$

Since $\mathbb{1}_{\mathfrak{E}_t} \leq \mathbb{1}_{\mathfrak{E}_{t-1}}$, and G_t is always positive, we have

$$G_t\mathbb{1}_{\mathfrak{E}_t} \leq G_t\mathbb{1}_{\mathfrak{E}_{t-1}} \tag{6}$$

Combining (5) and (6) together, we know $G_t \mathbb{1}_{\mathfrak{E}_{t-1}}$ is a supermartingale.

We can also bound the absolute value of the difference in every iteration:

$$\begin{aligned}
&|G_{t+1}\mathbb{1}_{\mathfrak{E}_t} - \mathbb{E}[G_{t+1}\mathbb{1}_{\mathfrak{E}_t}|\mathcal{F}_t]| \\
=&|(w_0 - w_{t+1} - (t+1)\eta a_+ + M) - (w_0 - w_t - \nabla \mathsf{L}(w_t) - (t+1)\eta a_+ + M)|\mathcal{F}_t] \\
\leq&\eta\nu
\end{aligned}$$

By Azuma's inequality, we get:

$$\Pr\left(G_t \mathbb{1}_{\mathfrak{E}_{t-1}} - G_0 \geq \lambda\right) \leq 2e^{-\frac{\lambda^2}{2t\eta^2\nu^2}}$$

That gives,

$$\Pr\left(G_t \mathbb{1}_{\mathfrak{E}_{t-1}} - G_0 \geq \sqrt{2t}\eta\nu \log^{1/2}(2\tau)\right) \leq 1/\tau$$

That means, if $\mathbb{1}_{\mathfrak{E}_t - 1}$ holds, with probability at least $1 - 1/\tau$,

$$w_0 - w_t - t\eta a_+ + M < \sqrt{2t}\eta\nu \log^{1/2}(2\tau) + G_0 = \sqrt{2t}\eta\nu \log^{1/2}(2\tau) + M$$

Which gives

$$w_t > w_0 - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau)$$

In other words, that means if $\mathbb{1}_{\mathfrak{E}_{t-1}}$ holds, then $\mathbb{1}_{\mathfrak{E}_t}$ also holds with probability at least $1 - 1/\tau$.

Therefore, if we are running $T_{\min}$ steps, we know that with probability at least $1 - \frac{T_{\min}}{\tau}$, $\mathbb{1}_{\mathfrak{E}_{T_{\min}}}$ holds. Therefore, by Lemma 5,

$$\forall t \leq T_{\min}, w_t > w_0 - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau) \geq p_{\min} - t\eta a_+ - \sqrt{2t}\eta\nu \log^{1/2}(2\tau) \geq 0 \quad \square$$

Similarly, we define $T_{\max} \triangleq \left(\frac{-\sqrt{2}\nu \log^{1/2}(2\tau) + \sqrt{2\nu^2 \log(2\tau) - 4(b_- - \nu)b_+}}{2b_+}\right)^2$, which satisfies the following inequality.

Lemma 7. $p_{\max} - T_{\max}\eta b_+ - \sqrt{2T_{\max}}\eta\nu \log^{1/2}(2\tau) < 0.$

Proof. By the definition of $p_{\max}$, we want to show that

$$(b_- - \nu) + T_{\max} b_+ + \sqrt{2T_{\max}} r \log^{1/2}(2\tau) \geq 0$$

Which holds by the definition of $T_{\max}$. $\square$

The Theorem below shows with decent probability, $T_{\max} - 1$ is an upper bound on the total number of iterates on the flat side in the i-th round.

Theorem 8. *If $w_0 \leq p_{\max}$, with probability at least $1 - \frac{T_{\max}}{\tau}$, $w_{T_{\max}} < 0$.*

Proof. Define event $\mathfrak{E}'_T = \{\forall t \leq T, w_t < w_0 - t\eta b_+ + \sqrt{2t}\eta\nu \log^{1/2}(2\tau)\}$, and $G'_t = w_t + t\eta b_+ > 0$.

We have

$$\begin{aligned}
&\mathbb{E}[G'_{t+1}\mathbb{1}_{\mathfrak{E}'_t}|\mathcal{F}_t] \\
=&\mathbb{E}[(w_{t+1} + (t+1)\eta b_+)\mathbb{1}_{\mathfrak{E}'_t}|\mathcal{F}_t] \\
\leq&\mathbb{E}[(w_t - \eta\omega_t + t\eta b_+)\mathbb{1}_{\mathfrak{E}'_t}|\mathcal{F}_t] \\
=&(w_t + t\eta b_+)\mathbb{1}_{\mathfrak{E}'_t} \\
=&G'_t \mathbb{1}_{\mathfrak{E}'_t}
\end{aligned}$$

Moreover, we know $\mathbb{1}_{\mathfrak{E}'_t} \leq \mathbb{1}_{\mathfrak{E}'_{t-1}}$, which means $G'_t \mathbb{1}_{\mathfrak{E}'_t} \leq G'_t \mathbb{1}_{\mathfrak{E}'_{t-1}}$. So $G'_t \mathbb{1}_{\mathfrak{E}'_{t-1}}$ is a supermartingale.

We can also bound the absolute value of the difference in every iteration:

$$\begin{aligned}&|G'_{t+1}\mathbb{1}_{\mathfrak{E}'_t} - \mathbb{E}[G'_{t+1}\mathbb{1}_{\mathfrak{E}'_t}|\mathcal{F}_t]|\\=&|(w_{t+1}+(t+1)\eta b_+) - (w_t - \eta\nabla \mathsf{L}(w_t) + (t+1)\eta b_+)|\mathcal{F}_t]|\\\leq&\eta\nu\end{aligned}$$

Using Azuma inequality, we get

$$\Pr\left(G'_t\mathbb{1}_{\mathfrak{E}'_{t-1}} - G'_0 \geq \sqrt{2t}\eta\nu\log^{1/2}(2\tau)\right) \leq 2e^{-\frac{t\eta^2\nu^2\log(2\tau)}{t\eta^2\nu^2}} = \frac{1}{\tau}$$

That means, if $\mathbb{1}_{\mathfrak{E}'_{t-1}}$ holds, with probability at least $1-1/\tau$,

$$w_t < w_0 - t\eta b_+ + \sqrt{2t}\eta\nu\log^{1/2}(2\tau)$$

In other words, $\mathbb{1}_{\mathfrak{E}'_t}$ also holds. Therefore, if we are running $T_{\max}$ steps, we know that with probability at least $1-\frac{T_{\max}}{\tau}$, $\mathbb{1}_{\mathfrak{E}'_{T_{\max}}}$ holds. Therefore, by Lemma 7, we know

$$w_{T_{\max}} < w_0 - T_{\max}\eta b_+ - \sqrt{2T_{\max}}\eta\nu\log^{1/2}(2\tau) < 0 \qquad \square$$

Remark. To make sure Theorem 6 is not vacuous, we need to make sure that $T_{\min}\geq 1$. If we want to make $T_{\min}$, say, at least 2, by Lemma 5, we have:

$$p_{\min} - 2\eta a_+ - 2\eta\nu\log^{1/2}(2\tau) \geq 0$$

Notice that $p_{\min} > (c-1)\eta a_+ - 2\eta\nu$, so we could solve the above inequality and get

$$\begin{aligned}&(c-1)\eta a_+ - 2\eta\nu - 2\eta a_+ - 2\eta\nu\log^{1/2}(2\tau)\geq 0\\\Rightarrow&\frac{(c-3)a_+ - 2\nu}{2\nu} \geq \log^{1/2}(2\tau)\\\Rightarrow&\tau \leq \frac{e^{\left(\frac{(c-3)a_+}{2\nu}-1\right)^2}}{2}\end{aligned}$$

Since we assume that c is a large constant and $a_+\geq\nu$, so τ can be fairly large in order to make sure $T_{\min}\geq 2$. We also know that $T_{\min}\leq\frac{-(a_- + a_+ + 2\nu)}{a_+} < c$.

On the other hand, by simple calculation, we know $T_{\max}\leq\frac{-(b_- - \nu)}{b_+} < c' < \frac{e^{c/3}}{6}$. Therefore, we can always pick a τ such that $\frac{T_{\min}+T_{\max}}{\tau}\leq\frac{1}{2}$. So finally, we are ready to prove Theorem 2.

Proof of Theorem 2. By Lemma 4 and Theorem 8, $T_{\max}$ is an upper bound on the length of the i-th round. By Theorem 6, we know that SGD will stay at flat side for at least $T_{\min}$ steps, and each step is lower bounded by $w_t > w_0 - t\eta a_+ - \sqrt{2t}\eta\nu\log^{1/2}(2\tau)$, therefore we know that with probability $1-\frac{T_{\min}+T_{\max}}{\tau}$:

$$\begin{aligned}\frac{1}{T_i}\sum_{j=0}^{T_i} w_j^i &\geq \frac{1}{T_{\max}}\left(\sum_{t=0}^{T_{\min}}[w_0 - t\eta a_+ - \sqrt{2t}\eta\nu\log^{1/2}(2\tau)] - \eta(a_+ + \nu)\right)\\&\geq\frac{1}{T_{\max}}\left(\eta a_+\frac{(T_{\min}+1)T_{\min}}{2} + \sqrt{2T_{\min}}\eta\nu\log^{1/2}(2\tau) - \eta(a_+ + \nu)\right)\\&\geq\frac{T_{\min}^2}{T_{\max}}\eta a_+\end{aligned}$$

The above inequality discussed the scenario when Theorem 6 and Theorem 8 hold. If they do not hold, which happens with probability at most $\frac{T_{\min}+T_{\max}}{\tau}$, we need to get lower bound for $\frac{1}{T_i}\sum_{j=0}^{T_i} w_j^i$.

Notice that by Lemma 4, we know that SGD stays at the sharp side for at most 1 iterate in each round, and also the iterates on the flat sides are always positive with $w_0 \geq p_{\min} > \eta(a_+ + \nu)$. Therefore, we have the following trivial bound:

$$\frac{1}{T_i}\sum_{j=0}^{T_i} w_j^i \geq \frac{-\eta(a_+ + \nu) + w_0}{2} > 0$$

Combining two cases together we get

$$\mathbb{E}\left[\frac{1}{T_i}\sum_{j=0}^{T_i} w_j^i\right] \geq \left(1 - \frac{T_{\min} + T_{\max}}{\tau}\right)\frac{T_{\min}^2}{T_{\max}}\eta a_+ + 0$$

Since we can pick τ s.t. $\frac{T_{\min}+T_{\max}}{\tau} \leq \frac{1}{2}$, we have

$$\mathbb{E}\left[\frac{1}{T_i}\sum_{j=0}^{T_i} w_j^i\right] \geq \frac{T_{\min}^2}{2T_{\max}}\eta a_+ \triangleq c_0 > 0 \qquad \square$$

E Additional Figures in Section 6.1: No Bumps Between SGD and SWA Solutions

Asymmetric valley of ResNet-56 on CIFAR-10, $(\overline{r}, \underline{r}, p, c) = (3.7, 3.0, 0.016, 10)$. See Figure 20.

Asymmetric valley of ResNet-110 on CIFAR-10, $(\overline{r}, \underline{r}, p, c) = (5.3, 3.5, 0.0050, 11)$. See Figure 21.

Asymmetric valley of ResNet-164 on CIFAR-10, $(\overline{r}, \underline{r}, p, c) = (2.5, 2.0, 0.027, 4.3)$. See Figure 22.

Asymmetric valley of VGG-16 on CIFAR-10, $(\overline{r}, \underline{r}, p, c) = (5.6, 4.0, 0.0033, 30)$. See Figure 23.

Asymmetric valley of DenseNet-100 on CIFAR-10, $(\overline{r}, \underline{r}, p, c) = (13.0, 8.0, 0.0029, 7.4)$. See Figure 24

Asymmetric valley of ResNet-56 on CIFAR-100, $(\overline{r}, \underline{r}, p, c) = (11.0, 6.0, 0.034, 15)$. See Figure 25.

Asymmetric valley of ResNet-110 on CIFAR-100, $(\overline{r}, \underline{r}, p, c) = (7.5, 4.5, 0.053, 6.3)$. See Figure 26.

Asymmetric valley of ResNet-164 on CIFAR-100, $(\overline{r}, \underline{r}, p, c) = (11.0, 6.0, 0.012, 18)$. See Figure 27.

Asymmetric valley of VGG-16 on CIFAR-100, $(\overline{r}, \underline{r}, p, c) = (9.0, 6.0, 0.0084, 17)$. See Figure 28.

Asymmetric valley of ResNet-56 on SVHN, $(\overline{r}, \underline{r}, p, c) = (5.0, 4.0, 0.018, 15)$. See Figure 29.

Asymmetric valley of ResNet-110 on SVHN, $(\overline{r}, \underline{r}, p, c) = (4.5, 2.5, 0.010, 11)$. See Figure 30.

Asymmetric valley of ResNet-164 on SVHN, $(\overline{r}, \underline{r}, p, c) = (4.5, 2.5, 0.033, 7.0)$. See Figure 31.

Asymmetric valley of VGG-16 on SVHN, $(\overline{r}, \underline{r}, p, c) = (4.5, 2.5, 0.0043, 43)$. See Figure 32.

Asymmetric valley of ResNet-56 on STL-10, $(\overline{r}, \underline{r}, p, c) = (8.0, 5.0, 0.33, 2.4)$. See Figure 33.

Asymmetric valley of ResNet-110 on STL-10, $(\overline{r}, \underline{r}, p, c) = (11.0, 6.0, 0.51, 3.5)$. See Figure 34.

Asymmetric valley of ResNet-164 on STL-10, $(\overline{r}, \underline{r}, p, c) = (12.0, 7.0, 0.092, 16)$. See Figure 35.

Asymmetric valley of VGG-16 on STL-10, $(\overline{r}, \underline{r}, p, c) = (5.0, 3.0, 0.11, 12)$. See Figure 36.

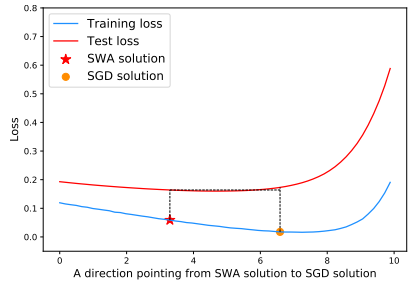


Figure 20: SWA and SGD interpolation (ResNet-56 on CIFAR-10)

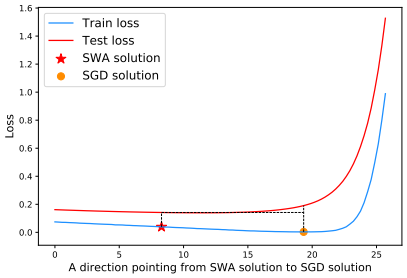


Figure 21: SWA and SGD interpolation (ResNet-110 on CIFAR-10)

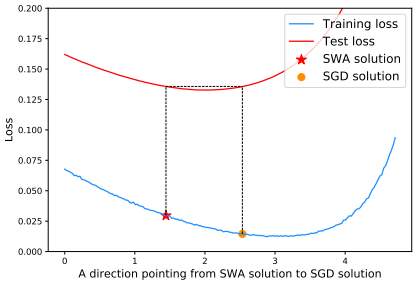


Figure 22: SWA and SGD interpolation (ResNet-164 on CIFAR-10)

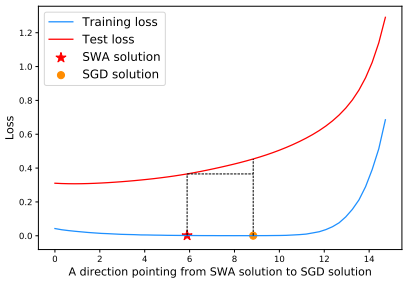


Figure 23: SWA and SGD interpolation (VGG-16 on CIFAR-10)

Figure 24: SWA and SGD interpolation (DenseNet-100 on CIFAR-10)

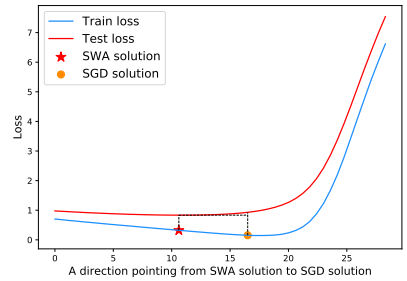


Figure 25: SWA and SGD interpolation (ResNet-56 on CIFAR-100)

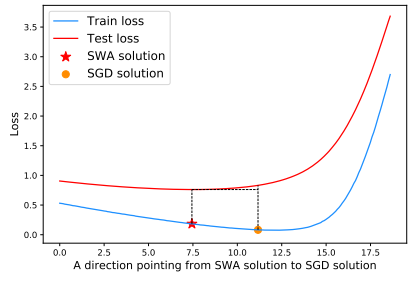


Figure 26: SWA and SGD interpolation (ResNet-110 on CIFAR-100)

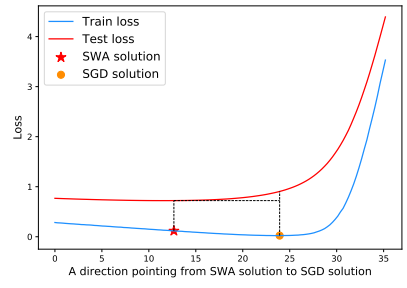


Figure 27: SWA and SGD interpolation (ResNet-164 on CIFAR-100)

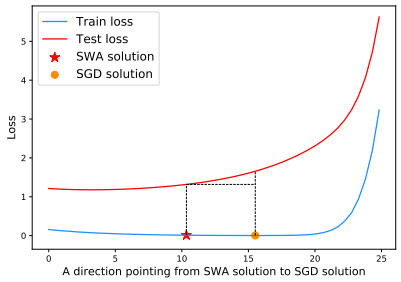


Figure 28: SWA and SGD interpolation (VGG-16 on CIFAR-100)

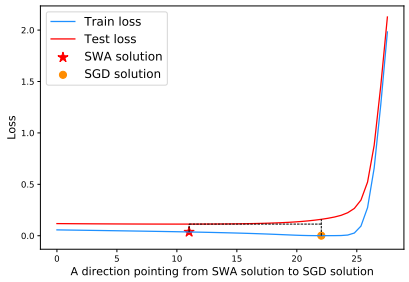


Figure 29: SWA and SGD interpolation (ResNet-56 on SVHN)

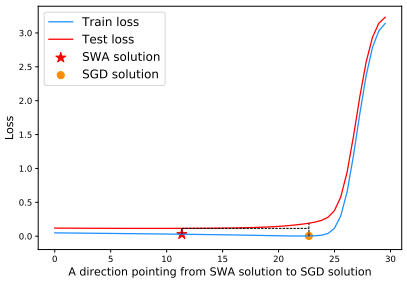


Figure 30: SWA and SGD interpolation (ResNet-110 on SVHN)

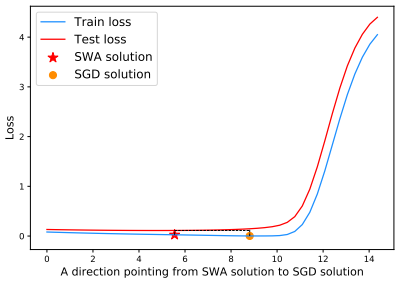


Figure 31: SWA and SGD interpolation (ResNet-164 on SVHN)

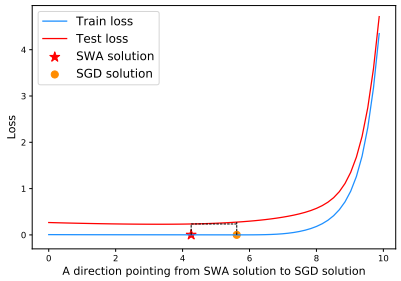


Figure 32: SWA and SGD interpolation (VGG-16 on SVHN)

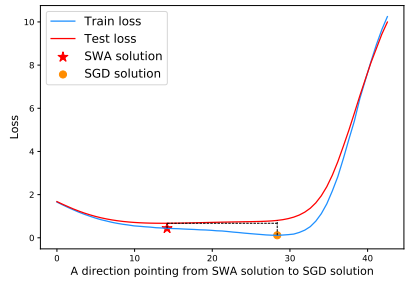


Figure 33: SWA and SGD interpolation (ResNet-56 on STL-10)

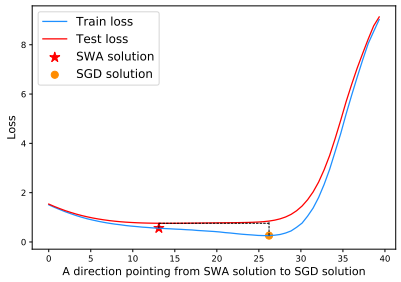


Figure 34: SWA and SGD interpolation (ResNet-110 on STL-10)

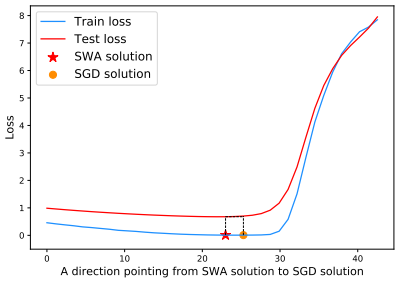


Figure 35: SWA and SGD interpolation (ResNet-164 on STL-10)

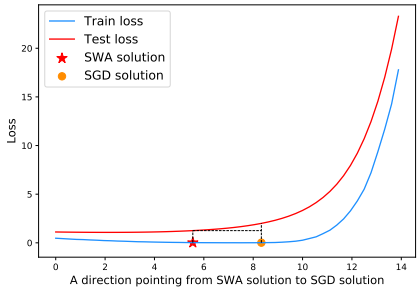


Figure 36: SWA and SGD interpolation (VGG-16 on STL-10)

F Additional Figures in Section 6.1: SGD Averaging Generates Good Bias

Examples for asymmetric directions of ResNet-110 on CIFAR-100 in Figure 37.

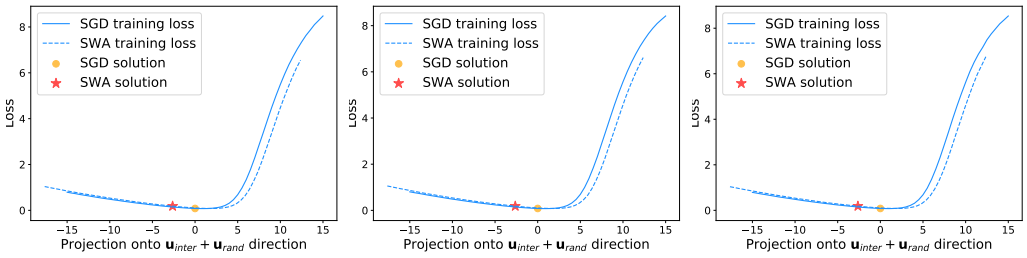


Figure 37: The average of SGD has a bias on flat side (ResNet-110 on CIFAR-100).

Examples for asymmetric directions of ResNet-164 on CIFAR-100 in Figure 38,

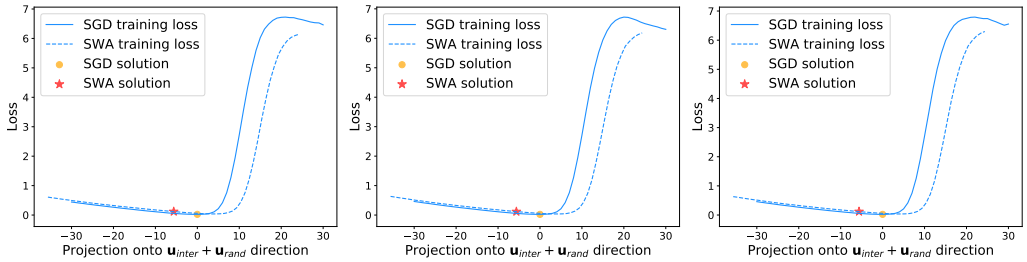


Figure 38: The average of SGD has a bias on flat side (ResNet-164 on CIFAR-100).

Examples for asymmetric directions of ResNet-110 on CIFAR-10 in Figure 39.

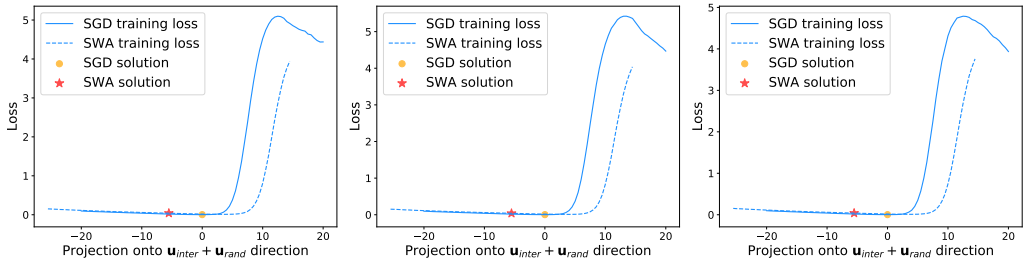


Figure 39: The average of SGD has a bias on flat side (ResNet-110 on CIFAR-10).

G Batch size effect

Keskar et al. [32] observed that training with small batch size using SGD algorithm generalizes better than training with large batch size. They argue that it is because large batch SGD tends to converge to sharp minima, while small batch SGD generally converges to flat minima. Here we present a slightly different view that batch size has an influence on choosing sides of an asymmetric valley.

We use a PreResNet-164 trained on CIFAR-100 as an example. We first running SGD with a batch size of 128 for 200 epochs to find a solution (denoted as *Large batch solution*), and then contintue the training with batch size 32 for another 80 epoch to find a nearby solution (denoted as *Small batch solution*). The reason for fine-tune is that we hope the two solutions are not far from each other, and we want to show that small batch size ensures a bias towards flat side.

From the results shown in Figure 40, it is clear that the small batch solution has worse training accuracy but better test accuracy. Meanwhile, there is no 'bump' between these solutions which suggests they are in the same basin. Therefore, small batch SGD generalizes better because it could find a better biased solution in the asymmetric valley under our training scheme, not because it finds a different wider or flatter minimum.

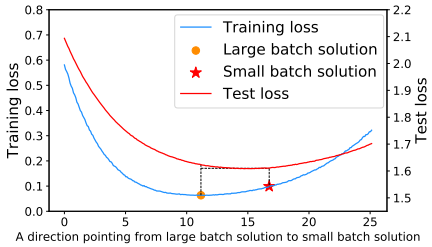


Figure 40: Large and small minibatch interpolation(batch size 128 to 32 of PreResNet-164 on CIFAR-100)

H Batch Norm and Asymmetric Valleys

In this section, we present empirical evidences that the Batch Normalization (BN) [24] adopted by modern neural networks seems to be a major cause for asymmetric valleys.

Directions on BN parameters are more asymmetric. For a given SGD solution, if we take a random direction where only the BN parameters have non-zero entries, and compare it with a random direction where only the non-BN parameters have non-zero entries, we observe that those BN-related directions are usually more asymmetric. The result with ResNet-110 on CIFAR-10 is shown in Figure 41, . As we can see, the Non-BN direction is sharp on both sides, but BN direction is flat on one side, and sharp on the other side. We also conducted trials with different networks and datasets, and obtained similar results (see Figure 42, 43 and 44).

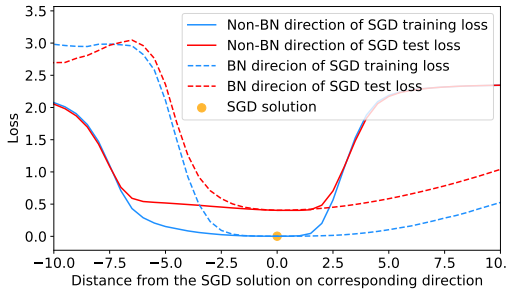


Figure 41: BN and Non-BN directions through a local minimum of ResNet-110 on CIFAR-10.

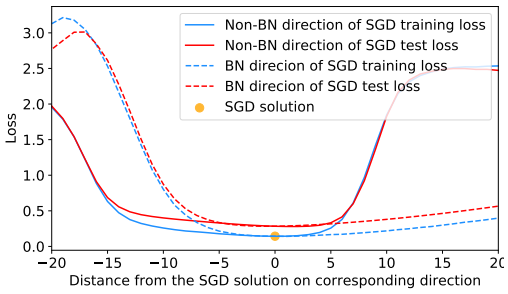


Figure 42: BN and Non-BN directions through a local minimum of of ResNet-164 on CIFAR-10.

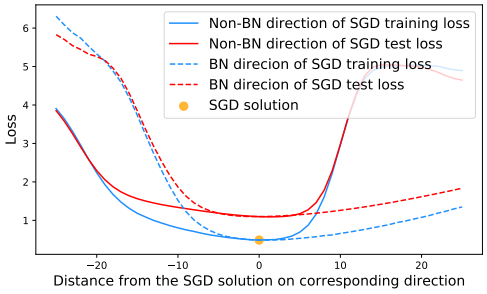


Figure 43: BN and Non-BN directions comparison of ResNet-110 on CIFAR-100

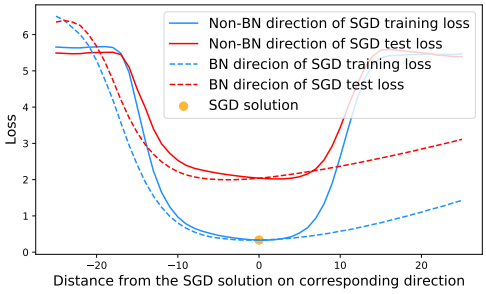


Figure 44: BN and Non-BN directions comparison of DenseNet-100 on CIFAR-100

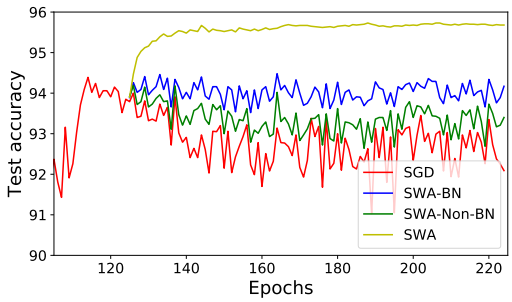


Figure 45: SGD averaging on BN parameters give better test accuracy compared with SGD averaging on non-BN parameters.

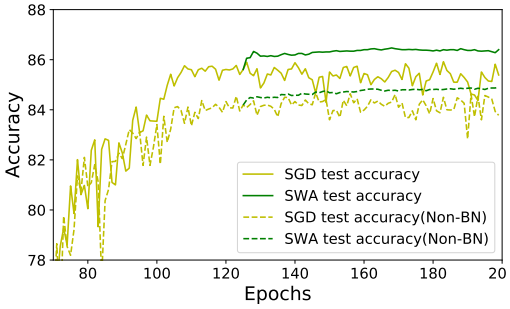


Figure 46: Test accuracy of ResNet-8 with and without BN layers, after running weight averaging (SWA).

SGD averaging is more effective on BN parameters. By Theorem 1 and 2, we know that SGD averaging could lead to biased solutions on asymmetric directions with better generalization. If BN indeed creates many asymmetric directions, can we improve the model performance by only averaging the weights of BN layers?

Note that BN parameters only constitute a small fraction of the total model parameters, e.g., 1.41% in a ResNet-110. In the follow experiment on ResNet-110 for CIFAR-10, we perform SGD averaging only on BN parameters, denoted as SWA-BN; and also averaging randomly selected non-BN parameters of the same amount (1.41% of the total parameters), denoted as SWA-Non-BN. The results are shown in Figure 45. It can be observed that averaging only BN parameters (blue curve) is more effective than averaging non-BN parameters (green curve), although there is still a gap comparing to averaging all the weights (yellow curve).

Moreover, we also conduct experiments with two 8-layer ResNets on CIFAR-10, one with BN layers and one without. We choose shallow networks here as deeper models without BN can not be effectively trained.

As shown in figure 46, we start weight averaging at the 126-th epoch. Although in both networks, we observe an improvement in test accuracy after averaging, it is clear that the network with BN layers have larger improvement compared with the network without BN layers. This again indicates that SGD averaging is more effective on BN parameters.