Stability analysis of stochastic gradient descent for homogeneous neural networks and linear classifiers

Alexandre Lemire Paquin (corresponding author), Brahim Chaib-draa, Philippe Giguère

[a]*Department of Computer Science and software engineering, Laval University, Pavillon Adrien-Pouliot 1065, av. de la Médecine, Quebec, G1V0A6, Quebec, Canada*

Abstract

We prove new generalization bounds for stochastic gradient descent when training classifiers with invariances. Our analysis is based on the stability framework and covers both the convex case of linear classifiers and the non-convex case of homogeneous neural networks. We analyze stability with respect to the normalized version of the loss function used for training. This leads to investigating a form of angle-wise stability instead of euclidean stability in weights. For neural networks, the measure of distance we consider is invariant to rescaling the weights of each layer. Furthermore, we exploit the notion of on-average stability in order to obtain a data-dependent quantity in the bound. This data-dependent quantity is seen to be more favourable when training with larger learning rates in our numerical experiments. This might help to shed some light on why larger learning rates can lead to better generalization in some practical scenarios.

Keywords:
Generalization, Deep learning, Stochastic gradient descent, Stability

Email addresses: alexandre.lemire-paquin.1@ulaval.ca (Alexandre Lemire Paquin (corresponding author)), brahim.chaib-draa@ift.ulaval.ca (Brahim Chaib-draa), philippe.giguere@ift.ulaval.ca (Philippe Giguère)

Stability analysis of stochastic gradient descent for homogeneous neural networks and linear classifiers

Anonymous authors (under review)

Abstract

We prove new generalization bounds for stochastic gradient descent when training classifiers with invariances. Our analysis is based on the stability framework and covers both the convex case of linear classifiers and the non-convex case of homogeneous neural networks. We analyze stability with respect to the normalized version of the loss function used for training. This leads to investigating a form of angle-wise stability instead of euclidean stability in weights. For neural networks, the measure of distance we consider is invariant to rescaling the weights of each layer. Furthermore, we exploit the notion of on-average stability in order to obtain a data-dependent quantity in the bound. This data-dependent quantity is seen to be more favourable when training with larger learning rates in our numerical experiments. This might help to shed some light on why larger learning rates can lead to better generalization in some practical scenarios.

Keywords:
Generalization, Deep learning, Stochastic gradient descent, Stability

1. Introduction

In the last few years, deep learning has succeeded in establishing state-of-the-art performances in a wide variety of tasks in fields like computer vision, natural language processing and bioinformatics (LeCun et al., 2015). Understanding when and how these networks generalize better is important to keep improving their performance. Many works starting mainly from Neyshabur et al. (2015), Zhang et al. (2017) and Keskar et al. (2017) hint to a rich interplay between regularization and the optimization process of learning the weights of the network. The idea is that a form of inductive bias can be realized implicitly by the optimization algorithm. The most popular algorithm

to train neural networks is stochastic gradient descent (SGD). It is therefore of great interest to study the generalization properties of this algorithm. An approach that is particularly well suited to investigate learning algorithms directly is the framework of stability (Bousquet and Elisseeff, 2002), (Elisseeff et al., 2005). It is argued in Nagarajan and Kolter (2019) that generalization bounds based on uniform convergence might be condemned to be essentially vacuous for deep networks. Stability bounds offer a possible alternative by trying to bound directly the generalization error of the output of the algorithm. The seminal work of Hardt et al. (2016) exploits this framework to study SGD for both the convex and non-convex cases. The main intuitive idea is to look at how much changing one example in the training set can generate a different trajectory when running SGD. If the two trajectories must remain close to each other then the algorithm has better stability.

This raises the question of how to best measure the distance between two classifiers. Our work investigates a measure of distance respecting invariances in homogeneous neural networks (and linear classifiers) instead of the usual euclidean distance. The measure of distance we consider is directly related to analyzing stability with respect to the normalized loss function (see equations 4 and 15) instead of the standard loss function used for training. In the convex case, we prove an upper bound on uniform stability with respect to the normalized loss function, which can then be used to prove a high probability bound on the test error of the output of SGD. In the non-convex case, we propose an analysis directly targeted toward homogeneous neural networks. How analysis exploits both the layerwise structure of neural networks and the invariances resulting from using homogeneous activation functions. We prove upper bounds on the on-average stability with respect to the normalized loss function, which can then be used to give generalization bounds on the test error. One nice advantage coming with our approach is that we do not need to assume that the loss function is bounded. Indeed, even if the loss function used for training is unbounded, the normalized loss is necessarily bounded.

Our main results for neural networks involve a data-dependent quantity that we estimate during training in our numerical experiments. The quantity is the sum over each layer of the ratio between the norm of the gradient for this layer and the norm of the parameters for the layer. We observe that larger learning rates lead to trajectories in parameter space keeping this quantity smaller during training. There are two ways to get our data-dependent quantity smaller during training. The first is by facilitating convergence (having smaller norms for the gradients). The second is by increasing the

weights of the network. If the weights are larger, the same magnitude for an update in weight space results in a smaller change in angle (see Figure 1). In our experiments, larger learning rates are seen to be more favorable in both regards.

Our main contributions are summarized as follows:

1) The first analysis of stability for SGD directly exploiting invariances in homogeneous neural networks (smooth and non-smooth cases including under weak convexity).
2) Empirical observations suggesting that our data-dependent quantity is interesting in understanding how larger learning rates can improve generalization.
3) An analysis of stability for the convex case naturally incorporating the norm of the initial point.

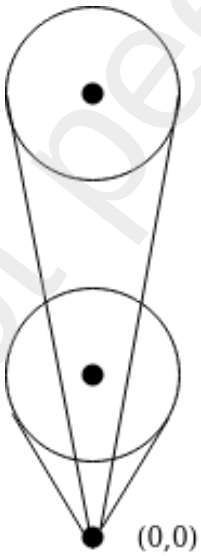


Figure 1: For the same magnitude of step taken (same ball radius), a larger norm of parameters leads to a smaller change in angle.

2. Related work

Normalized loss functions have been considered before (Poggio et al., 2019), (Liao et al., 2018). In Liao et al. (2018), test error is seen to be well correlated

with the normalized loss. This observation is one motivation for our study. (Poggio et al., 2019) writes down a generalization bound based on Rademacher complexity but motivated by the possible limitations of uniform convergence for deep learning (Nagarajan and Kolter, 2019) we take the stability approach instead.

Generalization of SGD has been investigated before in a large body of literature. Soudry et al. (2018) showed that gradient descent converges to the max-margin solution for logistic regression and Lyu and Li (2019) provides an extension to deep non-linear homogeneous networks. Nacson et al. (2019) gives similar results for stochastic gradient descent. The framework of algorithmic stability allows to include directly properties of optimization algorithms inside generalization bounds and is therefore well suited to study SGD. Starting from Hardt et al. (2016) without being exhaustive, a few representative examples of this line of research are Richards and Rabbat (2021), Bassily et al. (2020), Kuzborskij and Lampert (2018), Mou et al. (2018), Liu et al. (2017) and London (2017). London (2017) considers a combination of PAC-Bayes and algorithmic stability in order to study randomized learning algorithms. Liu et al. (2017) introduce the notion of argument stability and prove bounds on the generalization error of learning algorithm in terms of their argument stability. Mou et al. (2018) prove stability bounds for stochastic gradient Langevin Dynamics. Bassily et al. (2020) study the uniform stability of SGD on non-smooth convex loss functions (e.g. Hinge loss). Our main focus of interest is instead on the non-convex case and we restrict ourselves on the smooth case for convex loss functions. Richards and Rabbat (2021) investigate the stability of gradient descent under the hypothesis of weak convexity. They show that the magnitude of the most negative eigenvalue of the Hessian influence the stability of gradient descent. They do not consider the case of stochastic gradient descent however. Our work consider stochastic gradient descent for different batch sizes.

Since the work of Zhang et al. (2017) showing that currently used deep neural networks are so overparameterized that they can easily fit random labels, taking properties of the data distribution into account seems necessary to understand generalization of deep networks. In the context of stability, this means moving from uniform stability to on-average stability. This is the main concern of the work of Kuzborskij and Lampert (2018). They develop data-dependent stability bounds for SGD by extending over the work of Hardt et al. (2016). Their results have a dependence on the risk of the initialization point and the curvature of the initialization. They have to

assume a bound on the noise of the stochastic gradient. We do not make this assumption in our work. Furthermore, we maintain in our bounds for neural networks the properties after a "burn-in" period. This is motivated by the empirical work of Jastrzebski et al. (2020) arguing that in the early phase of training, the learning rate and batch size determine the properties of the trajectory after a "break-even point". Another work interested in on-average stability is Zhou et al. (2022). Differently from our work, their approach makes the extra assumptions that the variance of the stochastic gradients is bounded and also that the loss is bounded. Furthermore, our analysis directly exploits the structure of neural networks and the properties following from using homogeneous non-linearities. The work of Lei and Ying (2020) uses on-average model stability in order to remove the need for a uniform Lipschitz constant. The key quantities in their bounds are the empirical risks during training. Our key empirical quantities ($\hat{\zeta}_t(S)$) are instead given in terms of the norm of parameters and the norm of gradients during training. We interpret these quantities as a measure of exploration of the hypothesis space.

It has been observed in the early work of Keskar et al. (2017) that training with larger batch sizes can lead to a deterioration in test accuracy. The simplest strategy to reduce (at least partially) the gap with small batch training is to increase the learning rate (He et al., 2019), (Smith and Le, 2018), (Hoffer et al., 2017), (Goyal et al., 2017). We choose this scenario to investigate empirically the relevance of our stability bound for SGD on neural networks. Note that the results in Hardt et al. (2016) are more favorable to smaller learning rates. It seems therefore important in order to get theory closer to practice to understand better in what sense larger learning rates can improve stability.

3. Preliminaries

Let $l(w, z)$ be a non-negative loss function. Furthermore, let A be a randomized algorithm and denote by $A(S)$ the output of A when trained on training set $S = \{z_1, \cdots, z_n\} \sim \mathcal{D}^n$. The true risk for a classifier w is given as

$$L_{\mathcal{D}}(w) := \mathbb{E}_{z\sim\mathcal{D}} l(w, z)$$

and the empirical risk is given by

$$L_S(w) := \frac{1}{n}\sum_{i=1}^{n} l(w, z_i).$$

When considering the $0-1$ loss of a classifier w, we will write $L_{\mathcal{D}}^{0-1}(w)$. Furthermore, we will add a superscript α when the normalized losses l^{α} are under consideration (see equations 4 and 15). Our main interest is to ensure small test error and so we want to bound $L_{\mathcal{D}}^{0-1}(w)$. The usual approach is to minimize a surrogate loss upper bounding the $0-1$ loss. In this paper, we consider stochastic gradient descent with different batch sizes to minimize the empirical surrogate loss. The update rule of this algorithm for learning rates λ_t and a subset $B_t \subset S$ of size B is given by

$$w_{t+1} = w_t - \lambda_t \frac{1}{B} \sum_{z_j \in B_t} \nabla l(w_t, z_j). \tag{1}$$

We assume sampling uniformly with replacement in order to form each batch of training examples. In order to investigate generalization of this algorithm, we consider the framework of stability (Bousquet and Elisseeff, 2002).

We now give the definitions for uniform stability and on-average stability (random pointwise hypothesis stability in Elisseeff et al. (2005)) for randomized algorithms (see also Hardt et al. (2016) and Kuzborskij and Lampert (2018)). The definitions can be formulated with respect to any loss function but since we will study stability with respect to the l^{α} losses, we write the definitions in the context of this special case.

Definition 3.1. The algorithm A is said to be ϵ_{uni}^{α}-uniformly stable if for all $i \in \{1, \ldots, n\}$

$$\sup_{S, z_i', z} \mathbb{E}\left[|l^{\alpha}(A(S), z) - l^{\alpha}(A(S^{(i)}), z)| \right] \leq \epsilon_{uni}^{\alpha}. \tag{2}$$

Here, the expectation is taken over the randomness of A. The notation $S^{(i)}$ means that we replace the i^{th} example of S with z_i'.

Definition 3.2. The algorithm A is said to be ϵ_{av}^{α}-on-average stable if for all $i \in \{1, \ldots, n\}$

$$\mathbb{E}\left[|l^{\alpha}(A(S), z) - l^{\alpha}(A(S^{(i)}), z)| \right] \leq \epsilon_{av}^{\alpha}. \tag{3}$$

Here, the expectation is taken over $S \sim \mathcal{D}^n$, $z \sim \mathcal{D}$ and the randomness of A. The notation $S^{(i)}$ means that we replace the i^{th} example of S with z.

In Hardt et al. (2016), uniform stability with respect to the same loss the algorithm is executed on is considered. This is a natural choice, however if we are interested in the $0-1$ loss, different set of parameters w, w' can represent equivalent classifiers (that is, predict the same label for any input). This is the case for logistic regression since any rescaling of the parameters yields the same classifier (but they can have different training losses). This is also the case for homogeneous neural networks where we can rescale each layer without affecting the classifier. This is why we consider stability with respect to normalized losses instead. Note that we are still considering SGD executed on the original loss l (we do not change the algorithm A). The intuitive idea is to measure stability in terms of angles (more precisely, we consider distances between normalized vectors) instead of standard euclidean distances (see Figure 1). The proofs in Hardt et al. (2016) consist in bounding $\mathbb{E}||w_t - w'_t||$, where w_t represents the weights at iteration t when training on S and w'_t represents the weights at iteration t when training on the modified training set $S^{(i)}$. We will instead bound $\mathbb{E}||\frac{w_t}{||w_t||} - \frac{w'_t}{||w'_t||}||$ (or $\mathbb{E}[d(f,g)]$ for an appropriate measure of "distance" d between neural networks f and g).

In order to clarify further the motivation, consider the following algorithm: phase 1; run SGD as usual, phase 2; multiply all the weights by a large constant at the end of training. This algorithm effectively outputs the same classifier as SGD. Assume that the training loss is very close to 0 at the end of training for both SGD and this modified version of SGD (this would be true under overparametrization for example). In a generalization bound of the form $L_{\mathcal{D}}^{0-1}(A(S)) \leq L_S(A(S)) + Reg$, the determining factor to differentiate the two aforementioned algorithms will then be Reg (since in both cases $L_S(A(S)) \approx 0$). Under a standard stability analysis for the surrogate loss (used for training), the term Reg will be much worse for the modified algorithm. Indeed, the proof will be bounding $||w - w'||$ and if C is the constant used to multiply all the weights then the term Reg will be multiplied also by C. Choosing C large enough, the bound will become arbitrarily bad. However, the $0-1$ test loss is the same for both algorithms. Our approach does not suffer from this problem. Indeed, our analysis leads to the same generalization bound for both SGD and this modified version of SGD. Even though this modified version of SGD would not be used in practice, it still illustrates a problem that would occur if the directions taken by SGD happen to have a high component in the direction of increasing weights. The term Reg would be very large, thus leading to a poor bound.

Throughout the paper, $||\cdot||$ will denote the euclidean norm for vectors and the Frobenius norm for matrices. The proofs are given in Appendix A for the convex case and in Appendix B for the non-convex case.

4. Convex case: A first step toward the non-convex case

Since the convex case is easier to handle, it can be seen as a good preparation for the non-convex case. Consider a linear classifier parameterized by either a vector of weights (binary case) or a matrix of weights (multi-class case) that we denote by w in both cases. The normalized losses are defined by

$$l^{\alpha}(w,z) := l(\alpha\frac{w}{||w||},z), \tag{4}$$

for $\alpha > 0$.

In order to state the main result of this section, we need two common assumptions: L-Lipschitzness of l as a function of w and β-smoothness.

Definition 4.1. The function $l(w,z)$ is $L-$Lipschitz for all z in the domain (with respect to w) if for all w, w', z,

$$|l(w,z) - l(w',z)| \leq L||w - w'||. \tag{5}$$

Definition 4.2. The function $l(w,z)$ is $\beta-$smooth if for all w, w', z,

$$||\nabla l(w,z) - \nabla l(w',z)|| \leq \beta||w - w'||. \tag{6}$$

We are now ready to state the main result of this section.

Theorem 4.3. *Assume that $l(w,z)$ is convex, $\beta-$smooth and $L-$Lipschitz for all z. Furthermore, assume that the initial point w_0 satisfies $||w_0|| \geq K$ for some K such that $\hat{K} = K - L\sum_{i=0}^{T-1}\lambda_i > 0$ for a sequence of learning rates $\lambda_i \leq 2/\beta$. SGD is then run with batch size B on loss function $l(w,z)$ for T steps with the learning rates λ_t starting from w_0. Denote by ϵ^{α}_{uni} the uniform stability of this algorithm with respect to l^{α}. Then,*

$$\epsilon^{\alpha}_{uni} \leq \alpha\frac{2L^2B}{n\hat{K}}\sum_{i=0}^{T-1}\lambda_i. \tag{7}$$

What is the main difference between our bound and the bound in Hardt et al. (2016) (see Theorem Appendix A.3 in Appendix A) ? Our bound takes into account the norm of the initialization. The meaning of the bound is that it is not enough to use small learning rates and a small number of epochs to guarantee good stability (with respect to the normalized loss). We also need to take into account the norm of the parameters (here the norm of the initialization) to make sure that the "effective" learning rates are small. Note that all classifiers are contained in any ball around the origin even if the radius of the ball is arbitrarily small. Therefore, all control over stability is lost very close to the origin where even a small step (in Euclidean distance) can lead to a drastic change in the classifier. The norm of the initialization must therefore be large enough to ensure that the trajectory cannot get too close to the origin (in worst case, since uniform stability is considered). As a side note, we also incorporated the batch size into the bound which is not present in Hardt et al. (2016) (only $B = 1$ is considered).

When the learning rates are constant, let's say $\lambda_t = \frac{1}{\sqrt{n}}$ (this is a classical choice), single pass SGD ($T = n$) leads to $\epsilon^{\alpha}_{uni} = O(\frac{1}{\sqrt{n}(K-L\sqrt{n})})$. Without the term $\hat{K}$ (as in the standard euclidean analysis), the complexity would be $O(\frac{1}{\sqrt{n}})$. By choosing a large enough K (for example $K = (L+1)\sqrt{n}$) it is possible to improve over this previous complexity and get $O(\frac{1}{n})$. Another classical choice for the learning rates is $\lambda_t = O(\frac{1}{t})$. In this case, the sum of the learning rates is of order $O(\log(n))$. This means that we can take K of order $\log(n)$. We then obtain also $\epsilon^{\alpha}_{uni} = O(\frac{1}{n})$.

From this result, it is now possible to obtain a high probability bound for the test error. The bound is over draws of training sets S but not over the randomness of A. [1] So, we actually have the expected test error over the randomness of A in the bound. This is reminiscent of PAC-Bayes bounds where here the posterior distribution would be induced from the randomness of the algorithm A.

Theorem 4.4. *Fix $\alpha > 0$. Let $M_\alpha := \sup\{l(w,z)\ s.t.\ ||w|| \leq \alpha, ||x|| \leq R\}$. Then, for any $n > 1$ and $\delta \in (0,1)$, the following hold with probability greater*

[1] It is possible to obtain a bound holding over the randomness of A by exploiting the framework of Elisseeff et al. (2005). However, the term involving ρ in their theorem 15 does not converge to 0 when the size of the training set grows to infinity.

or equal to $1-\delta$ *over draws of training sets* S*:*

$$\mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S)) \leq \mathbb{E}_A L_S^{\alpha}(A(S)) + \epsilon_{uni}^{\alpha} + (2n\epsilon_{uni}^{\alpha} + M_{\alpha})\sqrt{\frac{\ln(1/\delta)}{2n}}. \quad (8)$$

Proof: The proof is an application of McDiarmid's concentration bound. Note that we do not need the training loss to be bounded since we consider the normalized loss which is bounded. The proof follows the same line as Theorem 12 in Bousquet and Elisseeff (2002) and we do not replicate it here. Note that we need to use that uniform stability implies generalization in expectation which is proven for example in Theorem 2.2 from Hardt et al. (2016).

Furthermore, a bound holding uniformly over all α's can be obtained using standard techniques.

Theorem 4.5. *Let* $C > 0$*. Assume that* $l^{\alpha}(w, z)$ *is a convex function of* α *for all* w, z *and that* ϵ_{uni}^{α} *is a non-decreasing function of* α*. Then, for any* $n > 1$ *and* $\delta \in (0, 1)$*, the following hold with probability greater or equal to* $1-\delta$ *over draws of training sets* S*:*

$$\mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S)) \leq \inf_{\alpha \in (0,C]} \Bigg\{ \mathbb{E}_A \max\left(L_S^{\alpha/2}(A(S)), L_S^{\alpha}(A(S))\right) + \epsilon_{uni}^{\alpha} + (2n\epsilon_{uni}^{\alpha} + M_{\alpha})\sqrt{\frac{2\ln(\sqrt{2}(2 + \log_2 C - \log_2 \alpha)) + \ln(1/\delta)}{2n}} \Bigg\}.$$

We discuss briefly the connection between our results and margin-based bounds obtained from Rademacher complexity. Write Φ^{γ} for the normalized ($\alpha = 1$) γ-ramp loss. The parameter γ then represents the geometric margin. In our work, α plays a similar role to $\frac{1}{\gamma}$. From Theorem 26.12 in Shalev-Shwartz and Ben-David (2014), if $||x|| \leq R$, it holds uniformly over all w on the unit sphere with probability at least $1-\delta$ that

$$L_{\mathcal{D}}^{0-1}(w) \leq \Phi_S^{\gamma}(w) + \frac{2R}{\gamma\sqrt{n}} + \sqrt{\frac{2\ln(2/\delta)}{n}}. \quad (9)$$

The Rademacher complexity term is given here by $\frac{R}{\gamma\sqrt{n}}$. We first remark that ϵ_{uni}^{α} is instead of order $\frac{1}{n}$. However, this term appears two times in 8, and the second time, it is multiplied by n before multiplying $\sqrt{\frac{\ln(1/\delta)}{2n}}$. There is

therefore also an added term of order $\frac{1}{\sqrt{n}}$ in our bound. The quantity involves the hyperparameters of SGD as given in Theorem 4.3. Since it is possible to control the hyperparameters of SGD, our bound can improve on the constants in front of $\frac{1}{\sqrt{n}}$ when compared to the Rademacher complexity bound. The intuition is that the Rademacher complexity is measuring the complexity of the full unit sphere while our approach based on stability only needs to consider an effective space of exploration on this sphere (implicitly defined by the hyperparameters of SGD). The following interpretation can therefore be given: If SGD is fast at finding a solution with large geometric margins on the training data then the solution will generalize better. Furthermore, even if we fail at implicitly minimizing L_S^α for very small values of α, the bound could still be reasonable if the exploration of the search space was limited.

In the next section, we investigate the non-convex case. We exploit on-average stability to obtain a data-dependent quantity in the bound. Note that it is also argued in Kuzborskij and Lampert (2018) that the worst case analysis of uniform stability might not be appropriate for deep learning.

5. Non-convex case

We consider homogeneous neural networks in the setup of multiclass classification. Write $f(x) = W_l(\sigma(\cdots W_2(\sigma(W_1 x))))$, where x is an input to the neural network, W_i denotes the weight matrix at layer i and σ denotes a homogeneous non-linearity ($\sigma(cx) = c^k\sigma(x)$ for any constant $c > 0$). Examples for the non-linearity are the ReLU function ($k = 1$), the quadratic function ($k = 2$) and the identity function ($k = 1$, leading to deep linear networks). Consider a non-negative loss function $l(s, y)$ that receives a score vector $s = f(x)$ and a label y as inputs. We require the loss function to be L-Lipschitz for all y as a function of s. That is, for all s, s', y,

$$|l(s, y) - l(s', y)| \leq L||s - s'||. \tag{10}$$

For example, we can use the cross-entropy loss (softmax function with negative log likelihood). In this case, it is simple to show by bounding the norm of the gradient of $l(s, y)$ with respect to s that we can use $L = \sqrt{2}$. Note that this is slightly different from the Lipschitz assumption of the previous section (given with respect to the weights w).

In order to control the behaviour of the non-linearity, we assume that for any $c > 0$, there exist constants B_c and L_c such that for any $x, y \in \mathbb{R}^d$ with

$||x|| \leq c$ and $||y|| \leq c$ we have

$$||\sigma(x)|| \leq B_c||x||, \tag{11}$$
$$||\sigma(x) - \sigma(y)|| \leq L_c||x - y||. \tag{12}$$

Note that the non-linearity σ is being applied component-wise when the input is a vector as above. It is easy to verify that, for the ReLU function, we have $B_c = 1$ and $L_c = 1$ for all c. Furthermore, for the quadratic function x^2, we have $B_c = c$ and $L_c = 2c$. The following lemma will be the starting point for our analysis.

Lemma 5.1. *Assume that $||x|| \leq R$. Let $\alpha_1, \cdots \alpha_l$ be positive real numbers and, for $1 \leq j \leq l$, denote $\tilde{W}_j := \frac{W_j}{||W_j||}$ and $\tilde{W}'_j := \frac{W'_j}{||W'_j||}$.*
Write $s_j = \alpha_j \tilde{W}_j(\sigma(\cdots \alpha_2 \tilde{W}_2(\sigma(\alpha_1 \tilde{W}_1 x))))$ and
$s'_j = \alpha_j \tilde{W}'_j(\sigma(\cdots \alpha_2 \tilde{W}'_2(\sigma(\alpha_1 \tilde{W}'_1 x))))$. Also, let c_j be an upper bound on the norm of layer j (this will be a constant depending on $\alpha_1, \cdots \alpha_j$ and R). Then, we have

$$||s_l - s'_l|| \leq R\Big(\prod_{j=1}^{l} \alpha_j\Big) \sum_{i=1}^{l} \tau_i ||\tilde{W}_i - \tilde{W}'_i||, \tag{13}$$

where $\tau_i = \prod_{j=1, j\neq i}^{l} \begin{pmatrix} B_{c_j} & if \quad j < i \\ L_{c_{j-1}} & if \quad j > i \end{pmatrix}$.

The previous lemma motivates a measure of "distance" between neural networks.

Definition 5.2. For neural networks f and g, where the weight matrices of f are given by $W_1 \cdots W_l$ and the weight matrices of g are given by $W'_1 \cdots W'_l$, define

$$d(f, g) := \sum_{i=1}^{l} \tau_i ||\frac{W_i}{||W_i||} - \frac{W'_i}{||W'_i||}||. \tag{14}$$

Note that this distance function is invariant to rescaling the weights of any layer. This is a desirable property since in a homogeneous network such a reparametrization leaves the class predicted by the classifier unchanged for any input to the network.

Let $\alpha_1, \cdots \alpha_l$ be positive real numbers. We define the $l^{\alpha_1,\cdots\alpha_l}(f,z)$ losses to be equal to

$$l(\alpha_l \frac{W_l}{||W_l||}(\sigma(\cdots \alpha_2 \frac{W_2}{||W_2||}(\sigma(\alpha_1 \frac{W_1}{||W_1||}x)))), y), \tag{15}$$

where $z = (x, y)$ and f is the neural network with weight matrix at layer i given by W_i. That is, we project the weight matrices to give the norm α_i to layer i and then we evaluate the loss l on this "normalized" network. For simplicity, we will only consider the case where all the α_i's are equal to say α and we will write $l^\alpha(f,z)$. From our definitions and lemma 5.1, we have that for all z and neural networks f and g,

$$|l^\alpha(f,z) - l^\alpha(g,z)| \leq LR\alpha^l d(f,g). \tag{16}$$

In order to bound stability with respect to l^α, we will have to ensure that the two trajectories cannot diverge too much in terms of $d(f,g)$.

We will consider two separate cases: the smooth case and the non-smooth case. When the activation function is smooth (for example x^k for $k \geq 1$), we will exploit the concept of layer-wise smoothness defined below.

Definition 5.3. Consider the gradient of the loss function with respect to the parameters W for some training example z. The vector containing only the partial derivatives for the weights of layer j will be denoted by $\nabla^{(j)} l(W,z)$. We define $\{\beta_j\}_{j=1}^l$-layerwise smoothness as the following property: For all j, z, $W = (W_1, \cdots, W_l)$ and $W' = (W'_1, \cdots, W'_l)$,

$$||\nabla^{(j)} l(W,z) - \nabla^{(j)} l(W',z)|| \leq \beta_j ||W_j - W'_j||. \tag{17}$$

We also let $\beta := \max\{\beta_j\}$. Note that β is upper bounding the spectral norm of the bloc diagonal approximation of the Hessian.

We are now ready to state the main theorem of this section for the smooth case.

Theorem 5.4. *Suppose that the loss function $l(s,y)$ is L-Lipschitz for all y, non-negative and that $l^\alpha(f,z)$ is bounded above by M_α. Furthermore, assume $\{\beta_j\}_{j=1}^l$-layerwise smoothness and that $||x|| \leq R$. Finally, let B denote the*

batch size, $\lambda_t \leq \frac{c}{t}$ *the learning rates and* T *the number of iterations SGD is being run. Then,*

$$\epsilon_{av}^{\alpha} \leq \inf_{t_0 \in \{1,2,\ldots,\frac{n}{B}\}} \left[\frac{2BLR\alpha^l}{(n-B)\beta} \left(\frac{T-1}{t_0-1} \right)^{c\beta} \sum_{t=t_0}^{T-1} \zeta_t + M_\alpha \frac{Bt_0}{n} \right], \quad (18)$$

where $\zeta_t := \sum_{j=1}^{l} \tau_j \mathbb{E}_{A,S,z} \left[C_j(S,z)^{T-t} \frac{||\nabla^{(j)} L_{B_t}(W_t)||}{K_t^{(j)}(S,z)} \right]$,

with $K_t^{(j)}(S,z) := \min\{||W_{j,t}||, ||W'_{j,t}||\}$ *and* $C_j(S,z) := \max_{t_0 \leq t \leq T-1} \frac{K_t^{(j)}(S,z)}{K_{t+1}^{(j)}(S,z)}$.

To evaluate the bound, we need to find the best t_0. There is a tradeoff here between two quantities. A small t_0 is better for the term $M_\alpha \frac{Bt_0}{n}$ but is worse for the remaining term. This establishes the best "burn in" period. The amount of "exploration" before t_0 does not effect the generalization bound. The amount of exploration measured by ζ_t (and through the learning rate via the value of c) becomes important only after iteration t_0. The bound will be better if we can reach a region in parameter space such that the classifier is then effectively not changing too much. This is measured through the norm of the gradient but also takes into account the norm of the parameters (via $K_t^{(j)}(S,z)$). When we reach larger norms of parameters, stability (with respect to the normalized loss) is less negatively affected. The intuitive reason is the following: the same magnitude of step results in a smaller change in the classifier if the parameters are larger (see Figure 1). In Hoffer et al. (2017), it is observed that small batch training and larger learning rates (finding solutions generalizing better) are reaching larger norms of parameters (see also our Figure 3). Using standard Euclidean distance in the analysis of stability would lead us to believe that this behaviour is highly undesirable. Our analysis shows that this behaviour can actually be favorable to the on-average stability with respect to the normalized loss. The quantity ζ_t also involves the terms $C_j(S,z)$ measuring how fast the norm of the parameters is growing from one iteration to the next. The value of $C_j(S,z)$ is better (smaller) if the norm of the parameters grows faster.

In the smooth case, an additional assumption called *weak convexity* is sometimes introduced. The idea is to exploit the situation where the magnitude of the negative eigenvalues can not be too large. In order to exploit the structure of neural networks (similarly to our notion of layerwise smoothness), we define a notion of layerwise weak convexity below.

Definition 5.5. Consider the Hessian of the loss function with respect to the parameters W for some training example z. The block containing only the second order derivatives with respect to the weights of layer j will be denoted by $H^{(j)}(W, z)$. Let $\epsilon_j \geq 0$ for all j. we define $\{\epsilon_j\}_{j=1}^{l}$-layerwise weak convexity as the following property: for all j, z and W,

$$H^{(j)}(W, z) \geq -\epsilon_j I. \tag{19}$$

Furthermore, we define $\epsilon := \max\{\epsilon_j\}$.

Theorem 5.6. *Suppose that the loss function $l(s, y)$ is L-Lipschitz for all y, non-negative and that $l^{\alpha}(f, z)$ is bounded above by M_{α}. Furthermore, assume $\{\beta_j\}_{j=1}^{l}$-layerwise smoothness, $\{\epsilon_j\}_{j=1}^{l}$-layerwise weak convexity and that $||x|| \leq R$. Finally, let B denote the batch size, $\lambda = \lambda_t \leq \frac{3}{2\beta}$ constant learning rates and T the number of iterations SGD is being run. Then,*

$$\epsilon_{av}^{\alpha} \leq \inf_{t_0 \in \{1,2,\ldots,\frac{n}{B}\}} \left[\frac{2BLR\alpha^{l}\lambda(T - t_0)}{n} \exp\left(\frac{(T - 1 - t_0)\lambda\epsilon}{1 - 2\lambda\epsilon} \right) \sum_{t=t_0}^{T-1} \zeta_t + M_{\alpha}(\frac{Bt_0}{n}) \right], \tag{20}$$

where ζ_t is defined as in theorem 5.4.

We note that a result holding in the convex case can be retrieved by letting $\epsilon = 0$. The main difference with the result from the previous section is that the worst case constant term $L/\hat{K}$ is now replaced with the data-dependent term $\sum_{t=t_0}^{T-1} \zeta_t$.

In practice, the ζ_t's are converging to 0 (see Figure 2). Assume here for the sake of illustration that $\zeta_t = O(1)$. Choosing $\lambda = O(\frac{1}{\sqrt{n}})$ and $T - t_0 = O(\sqrt{n})$, we obtain and upper bound that decays to 0 as n grows to infinity (for example we can take $T = t_0 + \sqrt{n}$ and $t_0 = n^c$ with $c < 1$). We have chosen λ and $T - t_0$ above in order to control the exponential term.

We now finally provide a result for the non-smooth case since the ReLU activation function is very common in practice.

Theorem 5.7. *Suppose that the loss function $l(s, y)$ is L-Lipschitz for all y, non-negative and that $l^{\alpha}(f, z)$ is bounded above by M_{α}. Furthermore, assume*

$||x|| \leq R$. *Finally, let* B *denote the batch size,* λ_t *the learning rates and* T *the number of iterations SGD is being run. Then,*

$$\epsilon_{av}^{\alpha} \leq \inf_{t_0 \in \{1,2,\ldots,\frac{n}{B}\}} \left[2LR\alpha^l \sum_{t=t_0}^{T-1} \lambda_t \zeta_t + M_\alpha \frac{Bt_0}{n} \right], \tag{21}$$

where ζ_t *is defined as in theorem 5.4.*

If we assume that $\zeta_t = O(1)$ as in the weakly convex case, using $\lambda_t = 1/t^2$ would implies that ϵ_{av}^{α} converges to 0 as n grows to infinity (taking $t_0 = \sqrt{n}$ for example). However, those learning rates would be very small. This is alleviated by the fact that the ζ_t's are converging to 0 (faster with larger learning rates in our experiments). Larger learning rates will therefore not hinder generalization if convergence is accelerated (and can even improve it). We also want to point out that choosing $\lambda = O(\frac{1}{\sqrt{n}})$ and $T - t_0 = O(\sqrt{n})$ as in the weakly convex case would not lead to a bound converging to 0 as n grows to infinity. As we should expect, the quality of the bound is thus better when more regularity assumptions can be made.

Once we can bound ϵ_{av}^{α}, it is then possible to get a probabilistic bound on the test error (holding over the randomness in the training sets and the randomness in the algorithm) by exploiting Theorem 12 in Elisseeff et al. (2005).

Theorem 5.8. *Fix* $\alpha > 0$. *Then, for any* $n > 1$ *and* $\delta \in (0, 1)$, *the following hold with probability greater or equal to* $1 - \delta$ *over draws of training sets* S *and the randomness of the algorithm* A*:*

$$L_{\mathcal{D}}^{0-1}(A(S)) \leq L_S^{\alpha}(A(S)) + \sqrt{\left(\frac{1}{\delta}\right) \frac{2M_\alpha^2 + 12nM_\alpha \epsilon_{av}^{\alpha}}{n}}. \tag{22}$$

6. Experiments

6.1. Learning rates and ζ_t

In this section we conduct some experiments on the datasets CIFAR10 Krizhevsky (2009) and MNIST LeCun and Cortes (2010). We consider the scenario where we try to reduce the performance gap between small batch and large batch training by increasing the learning rate. We will give some evidence suggesting that the quantity ζ_t can be of interest to assess generalization in this case.

We use a global learning rate being decayed one time by a factor of 10 in our experiments. No weight decay or momentum is used to stay closer to our theoretical analysis of SGD. Note that in principle, the learning rate could be as large as we want during the inital burn-in period (before t_0) without hurting stability. However, this burn-in period must be inside the first epoch in the theoretical results we presented. Since in practice we train for many epochs, it is not clear if such a small burn-in period is long enough to be significant in current practice. We still think that the quantity ζ_t is relevant to investigate empirically. We approximate its value on a training set S with the quantity $\hat{\zeta}_t(S) := \sum_{j=1}^{l} \frac{||\nabla^j L_{B_t}(W_t)||}{||W_{j,t}||}$. The quantities $C_j(S,z)$ can be evaluated empirically to be very close to 1 and so we neglect them in the expression for $\hat{\zeta}_t(S)$. Also note that $\tau_j = 1$ for all j in the case of ReLU networks. Instead of plotting the value for each iteration, we average $\hat{\zeta}_t(S)$ for each epoch. This leads to smoother curves.

We use a 5-layer convolutional Relu network consisting in 2 convolutional layers with maxpooling and then 3 fully connected layers with cross-entropy loss on CIFAR10. We use also the cross-entropy loss on MNIST but the neural network is a 6-layers fully connected network. In both cases, we use batch-normalization to facilitate training. All the results in the figures are obtained when using a batch size of 2048. We started by training with a smaller batch size of 256 and then tried to reduce the gap in performance between large batch and small batch training by increasing the learning rate. For example, on CIFAR10, we obtain a test accuracy of 86.23% when using a batch size of 256 and a learning rate of 0.5. When increasing the batch size to 2048 (and maintaining the learning rate to 0.5), the test accuracy dropped to 85.14%. This happened even if the training loss reach approximately the same value in both cases (0.0123 for batch size 256 and 0.0167 for batch size 2048). We then increased the learning rate to 1.0 and then to 1.5 reaching 85.63% in both cases (not completely solving the gap but reducing it). A similar phenomenon happens for MNIST. Here, with batch size 256 we get 98.57% ($lr = 0.05$) of test accuracy and for batch size 2048, we get 97.52% ($lr = 0.05$), 98.00% ($lr = 0.1$) and 98.39% ($lr = 0.5$). We plotted the values of $\hat{\zeta}_t(S)$ during training in Figure 2. We can see that it is better during all training when increasing the learning rate.

To compare with the analysis from Hardt et al. (2016), the quantity ζ_t would be replaced with a global Lipschitz constant which would not be affected by the actual trajectory of the algorithm. Therefore, in comparison

to our bound, the bound in Hardt et al. (2016) would be much more favorable to smaller learning rates. In other words, the worst case analysis of uniform convergence would require much smaller learning rates to be used than our result to guarantee good stability. The quantity ζ_t can be improved by accelerating convergence because of the numerator (norm of the gradients) but also by increasing the denominator (norm of the parameters). A larger learning rate can help in both these regards (see Figure 3). Note also that considering only the norm of the gradients without the norm of the parameters would lead to a less favorable quantity compared to considering both the norm of the gradients and the norm of the parameters. A standard analysis of stability (without the normalized loss) similar to Kuzborskij and Lampert (2018) would not benefit from the norm of the parameters.

6.2. Generalization bound and test error

We show in this section the usefulness of considering the normalized loss for bounding the test error. We evaluate the bound in Theorem 5.8 and compare it to an analogous version for the unnormalized loss. For this analogous version, we replace the upper bound M on the loss function by the largest loss achieved during training. Furthermore, the quantity ϵ_{av} is upper bounded by the Lipschitz constant times the Euclidean distance between the weights of the networks. The Lipschitz constant is replaced by the largest norm of gradients obtained during training. For the normalized loss, we upper bound ϵ_{av}^{α} by $LR\alpha^{l}\mathbb{E}d(f,g)$ (see equation 16). We plot the test error, the upper bound for the normalized case with $\alpha = 1.0$ and the upper bound for the unnormalized case in Figure 4. Further experiments (with label noise and a comparison of ADAM and SGD) are given in Appendix Appendix C.

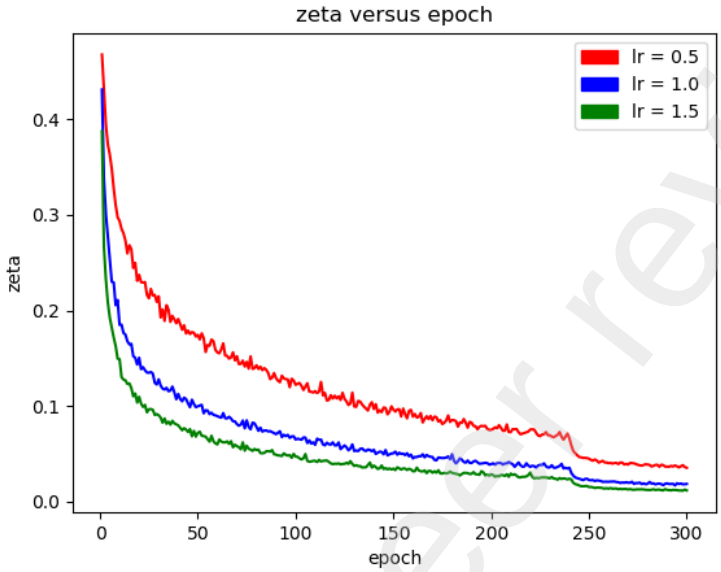
Figure 4: The bound obtained from the Euclidean distance is much worse than the bound obtained from our normalized distance. However, the generalization bound is still vacuous. The network is a 6-layer fully connected network trained on MNIST.

7. Conclusion

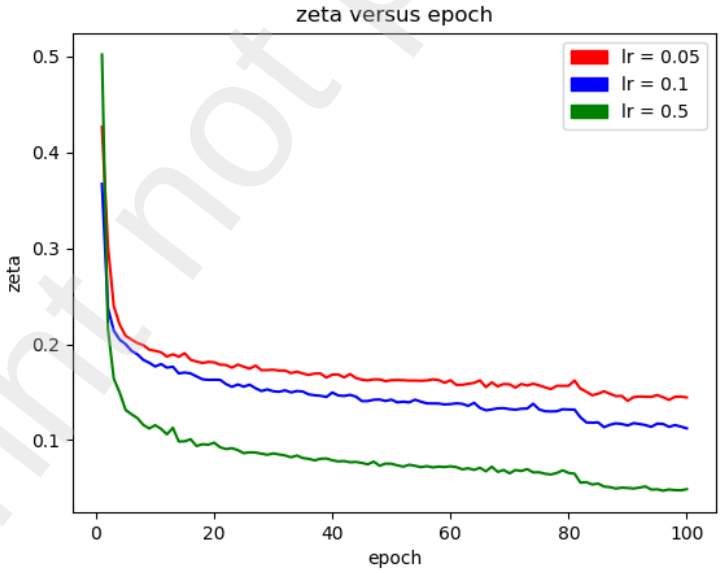
We investigated the algorithmic stability (uniform and on-average) of *SGD* with respect to the normalized loss functions. This leads naturally to consider a more meaningful measure of distance between classifiers respecting invariances in homogeneous neural networks and linear classifiers. Results are provided both for the smooth and non-smooth cases for neural networks. Furthermore, a bound on stability is derived under the assumption of weak convexity. Our experimental results show that our research direction is promising in order to improve our understanding of why using larger learning rates in training deep neural networks can lead to better generalization. Future work could investigate on-average stability with respect to the l^α losses for different optimization algorithms.

8. Acknowledgements

This work was supported by Fonds de recherche du Québec Nature et technologies (FRQNT).

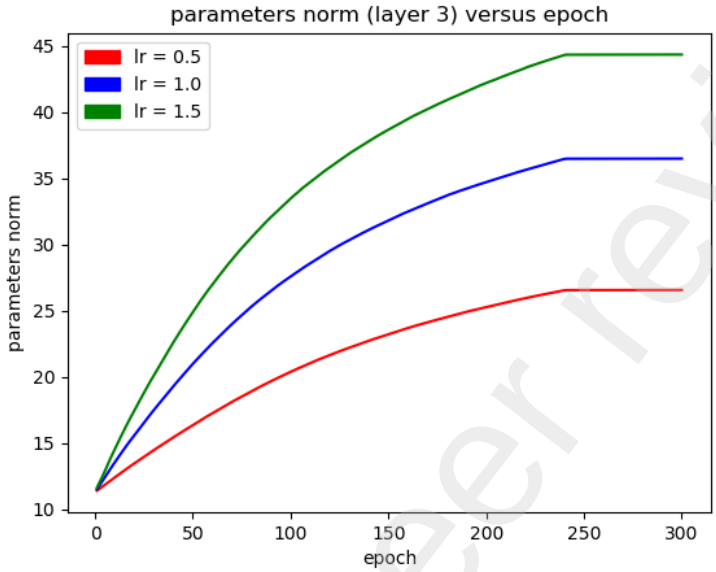


(a) CIFAR10

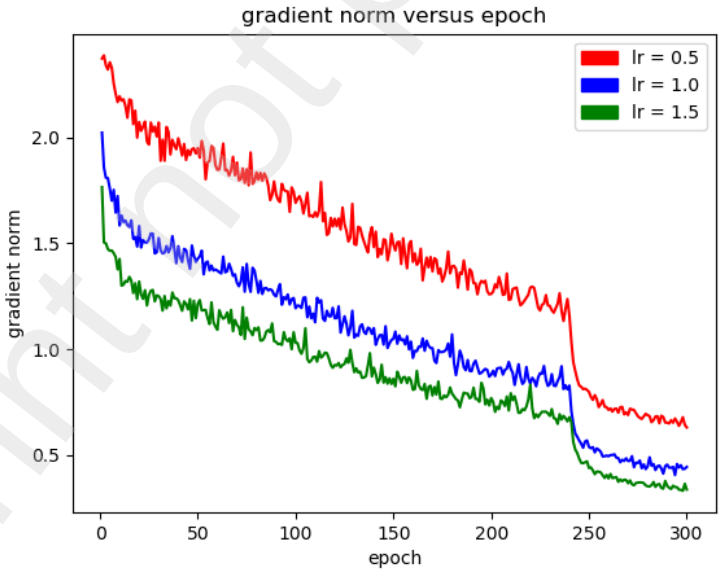


(b) MNIST

Figure 2: $\hat{\zeta}_t(S)$ when training a convolutional network on CIFAR10 and a fully connected network on MNIST.



(a) Norm of the parameters



(b) Norm of the gradient

Figure 3: Norm of the parameters (layer 3) and norm of the gradient when training a convolutional network on CIFAR10

Appendix A. Proofs for the convex case

Lemma Appendix A.1. *Let $v, w \in \mathbb{R}^n$ and $0 < c \leq \min\{||v||, ||w||\}$. Then,*

$$||\frac{v}{||v||} - \frac{w}{||w||}|| \leq \frac{||v-w||}{c}.$$

Proof: The proof follows from basic linear algebra manipulations. We give it here for completeness since it is important in what follows. We need to show that

$$\langle \frac{v}{||v||} - \frac{w}{||w||}, \frac{v}{||v||} - \frac{w}{||w||} \rangle \leq \frac{\langle v-w, v-w \rangle}{c^2}.$$

After some manipulations, one can see that this is equivalent to show that

$$||v||^2 + ||w||^2 - 2c^2 + 2(c^2 - ||v||||w||)\langle \frac{v}{||v||}, \frac{w}{||w||} \rangle \geq 0.$$

From Cauchy-Schwarz inequality, $\langle \frac{v}{||v||}, \frac{w}{||w||} \rangle \leq 1$. Since $c^2 - ||v||||w|| \leq 0$, the proof will be completed by showing that

$$||v||^2 + ||w||^2 - 2c^2 + 2(c^2 - ||v||||w||) \geq 0.$$

But this is true since

$$||v||^2 + ||w||^2 - 2||v||||w|| = (||v|| - ||w||)^2.$$

Lemma Appendix A.2. *Assume that the initial point w_0 satisfies $||w_0|| \geq K$ and that SGD is run with batch size B and a sequence of learning rates λ_t on an $L-$Lipschitz loss function $l(w, z)$ for all z. Then, for all $t \geq 1$,*

$$||w_t|| \geq K - L \sum_{i=0}^{t-1} \lambda_i.$$

Proof:

$$\begin{aligned}
||w_t|| &= ||w_{t-1} - \lambda_{t-1}\frac{1}{B}\sum_{j=1}^{B}\nabla l(w_{t-1}, z_j)|| \\
&\geq ||w_{t-1}|| - \lambda_{t-1}\frac{1}{B}||\sum_{j=1}^{B}\nabla l(w_{t-1}, z_j)|| \\
&\geq ||w_{t-1}|| - \lambda_{t-1}L \\
&\geq ||w_{t-2}|| - \lambda_{t-2}L - \lambda_{t-1}L \\
&\geq \cdots \\
&\geq ||w_0|| - L\sum_{i=0}^{t-1}\lambda_i \\
&\geq K - L\sum_{i=0}^{t-1}\lambda_i.
\end{aligned}$$

For ease of comparison, we give the statement of Theorem 3.8 in Hardt et al. (2016).

Theorem Appendix A.3. *(Theorem* 3.8 *in Hardt et al. (2016)) Assume that the loss function $f(\cdot; z)$ is β-smooth, convex and $L-$Lipschitz for every z. Suppose that we run SGD with step sizes $\alpha_t \leq 2/\beta$ for T steps. Then, SGD satisfies uniform stability with*

$$\epsilon_{uni} \leq \frac{2L^2}{n}\sum_{t=1}^{T}\alpha_t.$$

We are now ready to prove Theorem 4.3.

Proof of Theorem 4.3: The proof is similar to Hardt et al. (2016). Let w_t denotes the output of A after t steps on training set S and w'_t be the output of A after t steps on training set $S^{(i)}$ for some $i \in \{1, \cdots n\}$. From convexity, the update rule is $1-$expansive (see lemma 3.7 from Hardt et al. (2016)). This property can be used when the example i is not being picked at some iteration. Otherwise, the triangular inequality is used. Since the probability of picking the example i in a mini-batch of size B is smaller than $\frac{B}{n}$ (sampling with replacement) and exploiting Lemma Appendix A.1 and

Lemma Appendix A.2, we get

$$\begin{aligned}\mathbb{E}\left[||\frac{w_{t+1}}{||w_{t+1}||}-\frac{w'_{t+1}}{||w'_{t+1}||}||\right] &\leq \frac{1}{\hat{K}}\mathbb{E}\left[||w_{t+1}-w'_{t+1}||\right]\\ &\leq \frac{1}{\hat{K}}\left[\frac{B}{n}\left(\mathbb{E}||w_t-w'_t||+2L\lambda_t\right)+\left(1-\frac{B}{n}\right)\mathbb{E}||w_t-w'_t||\right]\\ &= \frac{1}{\hat{K}}\left(\mathbb{E}||w_t-w'_t||+\frac{2BL\lambda_t}{n}\right).\end{aligned}$$

Note that this is true since $\mathbb{E}||w_t-w'_t||\leq \mathbb{E}||w_t-w'_t||+2L\lambda_t$. Solving the recursion for $\mathbb{E}||w_t-w'_t||$, we have

$$\mathbb{E}||w_t-w'_t||\leq \tfrac{2BL}{n}\textstyle\sum_{i=0}^{t-1}\lambda_i.$$

Therefore,

$$\mathbb{E}\left[||\tfrac{w_{t+1}}{||w_{t+1}||}-\tfrac{w'_{t+1}}{||w'_{t+1}||}||\right]\leq \tfrac{2BL}{n\hat{K}}\textstyle\sum_{i=0}^{t}\lambda_i$$

The result then follows from the inequality

$$|l(\alpha\tfrac{w}{||w||},z)-l(\alpha\tfrac{w'}{||w'||},z)|\leq L\alpha||\tfrac{w}{||w||}-\tfrac{w'}{||w'||}||.$$

We finally prove Theorem 4.5.

Proof of Theorem 4.5: To simplify the text, write $\epsilon(\alpha,\delta):=\mathbb{E}_A L_S^\alpha(A(S))+\epsilon_{stab}^\alpha+(2n\epsilon_{stab}^\alpha+M_\alpha)\sqrt{\frac{\ln(1/\delta)}{2n}}$. For $i\geq 1$, let $\alpha_i=2^{(1-i)}C$ and $\delta_i=\frac{\delta}{2i^2}$. For any fixed i, we have

$$P_S\{\mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S))>\epsilon(\alpha_i,\delta_i)\}<\delta_i.$$

Therefore,

$$\begin{aligned}& P_S\{\forall i,\ \mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S))\leq\epsilon(\alpha_i,\delta_i)\}\\ =\ & 1-P_S\{\exists i,\ \mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S))>\epsilon(\alpha_i,\delta_i)\}\\ \geq\ & 1-\sum_{i=1}^{\infty}P_S\{\mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S))>\epsilon(\alpha_i,\delta_i)\}\\ \geq\ & 1-\sum_{i=1}^{\infty}\delta_i\geq 1-\delta.\end{aligned}$$

The last inequality follows from

$$\sum_{i=1}^{\infty} \delta_i = \frac{\delta}{2} \sum_{i=1}^{\infty} \frac{1}{i^2} = \frac{\delta}{2} \frac{\pi^2}{6} \leq \delta.$$

We want to show that the set

$$\{S : \ \forall i, \ \mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S)) \leq \epsilon(\alpha_i, \delta_i)\}$$

is contained in the set

$$\{S : \ \forall \alpha \in (0, C], \ \mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S)) \leq \mathbb{E}_A \max\left(L_S^{\alpha/2}(A(S)), L_S^{\alpha}(A(S))\right) + \epsilon_{stab}^{\alpha} + (2n\epsilon_{stab}^{\alpha} + M_\alpha)\sqrt{\tfrac{2\ln(\sqrt{2}(2+\log_2 C - \log_2 \alpha)) + \ln(1/\delta)}{2n}}\}.$$

Let S be such that $\forall i, \ \mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S)) \leq \epsilon(\alpha_i, \delta_i)$. Let $\alpha \in (0, C]$. Then, there exists i such that $\alpha_i \leq \alpha \leq 2\alpha_i$. We have

$$\begin{aligned}
\mathbb{E}_A L_{\mathcal{D}}^{0-1}(A(S)) &\leq \mathbb{E}_A L_S^{\alpha_i}(A(S)) + \epsilon_{stab}^{\alpha_i} + (2n\epsilon_{stab}^{\alpha_i} + M_{\alpha_i})\sqrt{\frac{\ln(1/\delta_i)}{2n}} \\
&\leq \mathbb{E}_A L_S^{\alpha_i}(A(S)) + \epsilon_{stab}^{\alpha} + (2n\epsilon_{stab}^{\alpha} + M_{\alpha})\sqrt{\frac{\ln(1/\delta_i)}{2n}} \\
&\leq \mathbb{E}_A L_S^{\alpha_i}(A(S)) + \epsilon_{stab}^{\alpha} + (2n\epsilon_{stab}^{\alpha} + M_{\alpha}) \\
&\sqrt{\frac{2\ln(\sqrt{2}(2+\log_2 C - \log_2 \alpha)) + \ln(1/\delta)}{2n}}
\end{aligned}$$

The second inequality is true since both ϵ_{stab}^{α} and M_α are non-decreasing functions of α and $\alpha_i \leq \alpha$. The last inequality is true since $\frac{1}{\delta_i} = \frac{2i^2}{\delta} \leq \frac{2(2+\log_2 C - \log_2 \alpha)^2}{\delta}$. Finally, the proof is concluded by using the convexity of $L_S^{\alpha}(A(S))$ with respect to α. Indeed, since $\frac{\alpha}{2} \leq \alpha_i \leq \alpha$, we must have

$$L_S^{\alpha_i}(A(S)) \leq \max\left(L_S^{\alpha/2}(A(S)), L_S^{\alpha}(A(S))\right).$$

Appendix B. Proofs for the non-convex case

Proof of Lemma 5.1: The proof is done by induction on the number of layers l. Suppose the result is true for $l-1$ layers. Then we have,

$$
\begin{aligned}
||s_l - s'_l|| &= \alpha_l ||\tilde{W}_l \sigma(s_{l-1}) - \tilde{W}'_l \sigma(s'_{l-1})|| \\
&= \alpha_l ||\tilde{W}_l \sigma(s_{l-1}) - \tilde{W}'_l \sigma(s_{l-1}) - \tilde{W}'_l(\sigma(s'_{l-1}) - \sigma(s_{l-1}))|| \\
&\leq \alpha_l ||\tilde{W}_l \sigma(s_{l-1}) - \tilde{W}'_l \sigma(s_{l-1})|| + \alpha_l ||\tilde{W}'_l(\sigma(s'_{l-1}) - \sigma(s_{l-1}))|| \\
&\leq \alpha_l ||\tilde{W}_l - \tilde{W}'_l||\, ||\sigma(s_{l-1})|| + \alpha_l ||\tilde{W}'_l||\, ||\sigma(s'_{l-1}) - \sigma(s_{l-1})|| \\
&\leq \alpha_l ||\tilde{W}_l - \tilde{W}'_l||\, ||s_{l-1}|| B_{c_{l-1}} + \alpha_l L_{c_{l-1}}\, ||s'_{l-1} - s_{l-1}|| \\
&\leq R\alpha_l ||\tilde{W}_l - \tilde{W}'_l|| \prod_{j=1}^{l-1} B_{c_j} \alpha_j + \alpha_l\, L_{c_{l-1}} ||s'_{l-1} - s_{l-1}|| \\
&\leq R \prod_{j=1}^{l} \alpha_j ||\tilde{W}_l - \tilde{W}'_l|| \prod_{j=1}^{l-1} B_{c_j} + \alpha_l L_{c_{l-1}}\, R \prod_{j=1}^{l-1} \alpha_j \sum_{i=1}^{l-1} \Bigg[\\
&||\tilde{W}_i - \tilde{W}'_i|| \prod_{j=1, j\neq i}^{l-1} \begin{pmatrix} B_{c_j} & if \quad j < i \\ L_{c_{j-1}} & if \quad j > i \end{pmatrix} \Bigg] \\
&= R \prod_{j=1}^{l} \alpha_j \sum_{i=1}^{l} \left[||\tilde{W}_i - \tilde{W}'_i|| \prod_{j=1, j\neq i}^{l} \begin{pmatrix} B_{c_j} & if \quad j < i \\ L_{c_{j-1}} & if \quad j > i \end{pmatrix} \right].
\end{aligned}
$$

The proof is finally concluded by observing that for one layer we have, $||s'_1 - s_1|| \leq R\alpha_1 ||\tilde{W}_1 - \tilde{W}'_1||$.

For the weakly convex case, the following lemma will be key to the proof.

Lemma Appendix B.1. *Under the assumption of $\{\epsilon_j\}$-layerwise weak convexity, for all $j, t, \lambda_t \leq \frac{3}{2\beta}, W_{j,t}, W'_{j,t}$,*

$$
||W_{j,t} - W'_{j,t} - \lambda_t(\nabla^{(j)} L_{B_t}(W_t) - \nabla^{(j)} L_{B_t}(W'_t))|| \leq \frac{1}{\sqrt{1 - 2\lambda_t \epsilon_j}} ||W_{j,t} - W'_{j,t}||. \tag{B.1}
$$

We note that the batch must be the same for both W_t and W'_t for this result to hold.

Proof The proof can be found in Richards and Rabbat (2021) (see proof of Theorem 5 and Lemma 2 in their section $B.3$)

Definition Appendix B.2. Let us introduce some notations. Let $\delta_t^{(j)}(S,z) := ||W_{j,t} - W'_{j,t}||$ and $\Delta_t^{(j)}(S,z) := \mathbb{E}_A[\delta_t^{(j)}(S,z) \,|\, \forall k,\, \delta_{t_0}^{(k)}(S,z) = 0]$. Here, $W_{j,t}$ is obtained when training with S for t iterations and $W'_{j,t}$ is obtained when training with $S^{(i)}$ for t iterations. The condition inside the expectation is that after t_0 iterations, the two networks are still exactly the same. Since we are interested in distances after normalization, we consider $\tilde{\delta}_t^{(j)}(S,z) := ||\frac{W_{j,t}}{||W_{j,t}||} - \frac{W'_{j,t}}{||W'_{j,t}||}||$ and $\tilde{\Delta}_t^{(j)}(S,z) := \mathbb{E}_A[\tilde{\delta}_t^{(j)}(S,z) \,|\, \forall k,\, \delta_{t_0}^{(k)}(S,z) = 0]$. We will further need $\hat{\delta}_t^{(j)}(S,z) := \frac{\delta_t^{(j)}(S,z)}{K_t^{(j)}(S,z)}$ and
$\hat{\Delta}_t^{(j)}(S,z) := \mathbb{E}_A[C_j(S,z)^{T-t}\hat{\delta}_t^{(j)}(S,z) \,|\, \forall k,\, \delta_{t_0}^{(k)}(S,z) = 0]$, where
$K_t^{(j)}(S,z) := \min\{||W_{j,t}||, ||W'_{j,t}||\}$ and $C_j(S,z) := \max_{t_0 \le t \le T-1} \frac{K_t^{(j)}(S,z)}{K_{t+1}^{(j)}(S,z)}$.

Before proving Theorem 5.4, we establish a lemma. Note that the structure of the proof of the following Lemma and of Theorem 5.4 is similar to the corresponding results in Hardt et al. (2016) and in Kuzborskij and Lampert (2018).

Lemma Appendix B.3. *Suppose that the loss function $l(s,y)$ is L-Lipschitz for all y, non-negative and that $l^\alpha(f,z)$ is bounded above by M_α. Also, assume that $||x|| \le R$. Furthermore, let B denote the batch size and T the number of iterations SGD is being run. Then, for any $t_0 \in \{0,1,2,\ldots,\frac{n}{B}\}$, the on-average stability satisfies*

$$\epsilon_{av}^\alpha \le LR\alpha^l \sum_{j=1}^{l} \tau_j \mathbb{E}_{S,z}\left[\mathbb{E}_A[\tilde{\delta}_T^{(j)}(S,z) \,|\, \forall k,\, \delta_{t_0}^{(k)}(S,z) = 0]\right] + M_\alpha(\frac{Bt_0}{n}).$$

Proof: Write the quantity $|l^\alpha(f,z) - l^\alpha(g,z)|$ as the sum of $|l^\alpha(f,z) - l^\alpha(g,z)|I\{\forall k,\, \delta_{t_0}^{(k)}(S,z) = 0\}$ and $|l^\alpha(f,z) - l^\alpha(g,z)|I\{\exists k :\, \delta_{t_0}^{(k)}(S,z) \neq 0\}$. We bound the first term by using the fact that

$$|l^\alpha(f,z) - l^\alpha(g,z)| \le LR\alpha^l d(f,g) = LR\alpha^l \sum_{j=1}^{l} \tau_j \tilde{\delta}_T^{(j)}(S,z).$$

For the second term, we use that $l^\alpha(f,z)$ is bounded above by M_α and non-negative to write

$$|l^{\alpha}(f,z) - l^{\alpha}(g,z)| \leq M_{\alpha}.$$

The result then follows from the fact that the probability of picking example i in t_0 iterations is smaller than $\frac{Bt_0}{n}$.

Proof of theorem 5.4: From Lemma Appendix A.1, we always have $\tilde{\delta}_t^{(j)}(S,z) \leq \hat{\delta}_t^{(j)}(S,z)$. Therefore, from the previous Lemma,

$$\epsilon_{av}^{\alpha} \leq LR\alpha^l \sum_{j=1}^{l} \tau_j \mathbb{E}_{S,z}\left[\hat{\Delta}_T^{(j)}(S,z)\right] + M_{\alpha}(\frac{Bt_0}{n}).$$

First note that under our definitions,

$$\begin{aligned}
\hat{\delta}_{t+1}^{(j)}(S,z) &= \frac{\delta_{t+1}^{(j)}(S,z)}{K_{t+1}^{(j)}(S,z)} \\
&\leq C_j(S,z)\frac{\delta_{t+1}^{(j)}(S,z)}{K_t^{(j)}(S,z)} \\
&\leq \frac{C_j(S,z)}{K_t^{(j)}(S,z)}\left[\delta_t^{(j)}(S,z) + \lambda_t||\nabla^{(j)}L_{B_t}(W_t) - \nabla^{(j)}L_{B_t'}(W_t')||\right] \\
&= C_j(S,z)\,\hat{\delta}_t^{(j)}(S,z) + \lambda_t C_j(S,z)\frac{||\nabla^{(j)}L_{B_t}(W_t) - \nabla^{(j)}L_{B_t'}(W_t')||}{K_t^{(j)}(S,z)}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
C_j(S,z)^{T-(t+1)}\,\hat{\delta}_{t+1}^{(j)}(S,z) \leq\; & C_j(S,z)^{T-t}\,\hat{\delta}_t^{(j)}(S,z) \\
& + \lambda_t C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t) - \nabla^{(j)}L_{B_t'}(W_t')||}{K_t^{(j)}(S,z)}.
\end{aligned}$$

Here, B_t denotes the batch of samples at iteration t when training on S and B_t' denotes the batch of samples at iteration t when training on $S^{(i)}$. When $B_t = B_t'$, we will use $\{\beta_j\}_{j=1}^{l}$-layerwise smoothness to bound $||\nabla^{(j)}L_{B_t}(W_t) - \nabla^{(j)}L_{B_t'}(W_t')||$. Otherwise, we use simply the triangular inequality. Let $p(B,n)$ be the probability of picking the example i in a mini-batch of size B (this is smaller than $\frac{B}{n}$). For $t \geq t_0$, we have

$$\begin{aligned}
\hat{\Delta}_{t+1}^{(j)}(S,z) \leq (1-p(B,n))(1+\beta_j\lambda_t)\hat{\Delta}_t^{(j)}(S,z) + p(B,n)\Bigg(\hat{\Delta}_t^{(j)}(S,z) + \\
\lambda_t\mathbb{E}_A[C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t)||}{K_t^{(j)}(S,z)} + C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t'}(W_t')||}{K_t^{(j)}(S,z)}]\Bigg).
\end{aligned}$$

Define $\hat{\Delta}_t^{(j)} := \mathbb{E}_{S,z}\hat{\Delta}_t^{(j)}(S,z)$ and
$\zeta_t^{(j)} := \mathbb{E}_{A,S,z}C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t)||}{K_t^{(j)}(S,z)}$ for any t. Taking the expectation over S and z on both sides of the previous inequality, we get

$$\hat{\Delta}_{t+1}^{(j)} \leq (1-p(B,n))(1+\beta_j\lambda_t)\hat{\Delta}_t^{(j)} + p(B,n)(\hat{\Delta}_t^{(j)} + 2\lambda_t\zeta_t^{(j)}).$$

This is true since
$\mathbb{E}_{A,S,z}C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t)||}{K_t^{(j)}(S,z)} = \mathbb{E}_{A,S,z}C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B'_t}(W'_t)||}{K_t^{(j)}(S,z)}$. Rearranging terms and using $1+x \leq \exp(x)$, we get

$$\begin{aligned}\hat{\Delta}_{t+1}^{(j)} &\leq [1+(1-p(B,n))\beta_j\lambda_t]\hat{\Delta}_t^{(j)} + 2p(B,n)\lambda_t\zeta_t^{(j)} \\ &\leq \exp((1-p(B,n))\beta_j\lambda_t)\hat{\Delta}_t^{(j)} + 2p(B,n)\lambda_t\zeta_t^{(j)}.\end{aligned}$$

Developing the recursion yields

$$\begin{aligned}\hat{\Delta}_T^{(j)} &\leq \sum_{t=t_0}^{T-1} 2p(B,n)\lambda_t\zeta_t^{(j)} \prod_{k=t+1}^{T-1}\exp\left((1-p(B,n))\frac{c\beta_j}{k}\right) \\ &\leq \frac{2B}{n}\sum_{t=t_0}^{T-1}\lambda_t\zeta_t^{(j)}\exp\left((1-p(B,n))c\beta_j\sum_{k=t+1}^{T-1}\frac{1}{k}\right) \\ &\leq \frac{2B}{n}\sum_{t=t_0}^{T-1}\lambda_t\zeta_t^{(j)}\exp\left((1-p(B,n))c\beta_j\log(\frac{T-1}{t})\right) \\ &\leq \frac{2Bc}{n}\sum_{t=t_0}^{T-1}\frac{\zeta_t^{(j)}}{t}\left(\frac{T-1}{t}\right)^{(1-p(B,n))c\beta_j} \\ &\leq \frac{2Bc}{n}\sum_{t=t_0}^{T-1}\frac{\zeta_t^{(j)}}{t}\left(\frac{T-1}{t}\right)^{(1-p(B,n))c\beta} \\ &\leq \frac{2Bc}{n}\max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\}(T-1)^{(1-p(B,n))c\beta}\sum_{t=t_0}^{T-1}\left(\frac{1}{t}\right)^{(1-p(B,n))c\beta-1} \\ &\leq \frac{2Bc}{nc(1-p(B,n))\beta}\max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\}\left(\frac{T-1}{t_0-1}\right)^{(1-p(B,n))c\beta} \\ &\leq \frac{2B}{(n-B)\beta}\left(\frac{T-1}{t_0-1}\right)^{c\beta}\max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\}.\end{aligned}$$

Therefore, ϵ_{av}^{α} is upper bounded by

$$\inf_{t_0\in\{1,2,\dots,\frac{n}{B}\}}\left[\frac{2BLR\alpha^l}{(n-B)\beta}\left(\frac{T-1}{t_0-1}\right)^{c\beta}\sum_{j=1}^{l}\tau_j\max_{t_0\le t\le T-1}\{\zeta_t^{(j)}\}+M_\alpha(\frac{Bt_0}{n})\right].$$

To complete the proof, we will use that $\max_{t_0\le t\le T-1}\{\zeta_t^{(j)}\}\le\sum_{t=t_0}^{T-1}\zeta_t^{(j)}$ and reverse the sum order. With the definition $\zeta_t:=\sum_{j=1}^{l}\tau_j\zeta_t^{(j)}$, we then have

$$\epsilon_{av}^{\alpha}\le\inf_{t_0\in\{1,2,\dots,\frac{n}{B}\}}\left[\frac{2BLR\alpha^l}{(n-B)\beta}\left(\frac{T-1}{t_0-1}\right)^{c\beta}\sum_{t=t_0}^{T-1}\zeta_t+M_\alpha(\frac{Bt_0}{n})\right].$$

Proof of Theorem 5.6: The beginning of the proof is similar to the proof of Theorem 5.4. We get

$$\begin{aligned}\hat{\Delta}_{t+1}^{(j)}(S,z)\le(1-p(B,n))\mathbb{E}_A\Bigg[&\\ &\frac{C_j(S,z)^{T-t}}{K_t^{(j)}(S,z)}||W_{j,t}-W'_{j,t}-\lambda_t(\nabla^{(j)}L_{B_t}(W_t)-\nabla^{(j)}L_{B_t}(W'_t))||\Bigg]\\ &+p(B,n)\Bigg(\hat{\Delta}_t^{(j)}(S,z)+\\ &\lambda_t\mathbb{E}_A\left[C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t)||}{K_t^{(j)}(S,z)}+C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B'_t}(W'_t)||}{K_t^{(j)}(S,z)}\right]\Bigg).\end{aligned}$$

When $B_t=B'_t$, we exploit Lemma Appendix B.1 and obtain

$$\begin{aligned}\hat{\Delta}_{t+1}^{(j)}(S,z)\le&\frac{(1-p(B,n))}{\sqrt{1-2\lambda_t\epsilon_j}}\hat{\Delta}_t^{(j)}(S,z)\\ &+p(B,n)\Bigg(\hat{\Delta}_t^{(j)}(S,z)+\lambda_t\mathbb{E}_A\Bigg[C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t)||}{K_t^{(j)}(S,z)}\\ &+C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B'_t}(W'_t)||}{K_t^{(j)}(S,z)}\Bigg]\Bigg).\end{aligned}$$

As in the previous proof, define $\hat{\Delta}_t^{(j)}:=\mathbb{E}_{S,z}\hat{\Delta}_t^{(j)}(S,z)$ and $\zeta_t^{(j)}:=\mathbb{E}_{A,S,z}C_j(S,z)^{T-t}\frac{||\nabla^{(j)}L_{B_t}(W_t)||}{K_t^{(j)}(S,z)}$ for any t. Taking the expectation over S and z on both sides of the previous inequality, we get

$$\hat{\Delta}_{t+1}^{(j)}\le\tfrac{(1-p(B,n))}{\sqrt{1-2\lambda_t\epsilon_j}}\hat{\Delta}_t^{(j)}+p(B,n)(\hat{\Delta}_t^{(j)}+2\lambda_t\zeta_t^{(j)}).$$

Rearranging terms, denoting $z = 1 - 2\lambda_t\epsilon_j$, $p = p(B,n)$ and using $1 + x \leq \exp(x)$, we get

$$\begin{aligned}
\hat{\Delta}_{t+1}^{(j)} &\leq \sqrt{\frac{(1-p+p\sqrt{z})^2}{z}}\hat{\Delta}_t^{(j)} + 2p\lambda_t\zeta_t^{(j)} \\
&\leq \sqrt{1 + \frac{(1-p+p\sqrt{z})^2 - z}{z}}\hat{\Delta}_t^{(j)} + 2p\lambda_t\zeta_t^{(j)} \\
&\leq \sqrt{1 + \frac{1-z}{z}}\hat{\Delta}_t^{(j)} + 2p\lambda_t\zeta_t^{(j)} \\
&\leq \exp\left(\frac{1-z}{2z}\right)\hat{\Delta}_t^{(j)} + 2p\lambda_t\zeta_t^{(j)} \\
&= \exp\left(\frac{\lambda_t\epsilon_j}{1-2\lambda_t\epsilon_j}\right)\hat{\Delta}_t^{(j)} + 2p\lambda_t\zeta_t^{(j)},
\end{aligned}$$

where we used that $\sqrt{z} \leq 1$ to get the third inequality. Therefore, Developing the recursion yields

$$\begin{aligned}
\hat{\Delta}_T^{(j)} &\leq \sum_{t=t_0}^{T-1} 2p(B,n)\lambda_t\zeta_t^{(j)} \prod_{k=t+1}^{T-1} \exp\left(\frac{\lambda_k\epsilon_j}{1-2\lambda_k\epsilon_j}\right) \\
&\leq \frac{2B}{n}\sum_{t=t_0}^{T-1} \lambda_t\zeta_t^{(j)} \exp\left(\sum_{k=t+1}^{T-1} \frac{\lambda_k\epsilon_j}{1-2\lambda_k\epsilon_j}\right) \\
&\leq \frac{2B}{n} \max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\} \sum_{t=t_0}^{T-1} \lambda_t \exp\left(\sum_{k=t+1}^{T-1} \frac{\lambda_k\epsilon_j}{1-2\lambda_k\epsilon_j}\right).
\end{aligned}$$

In the case of constant learning rates $\lambda = \lambda_t$, we get

$$\begin{aligned}
\hat{\Delta}_T^{(j)} &\leq \frac{2B\lambda}{n} \max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\} \sum_{t=t_0}^{T-1} \exp\left(\frac{(T-1-t)\lambda\epsilon_j}{1-2\lambda\epsilon_j}\right) \\
&\leq \frac{2B\lambda(T-t_0)}{n} \max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\} \exp\left(\frac{(T-1-t_0)\lambda\epsilon_j}{1-2\lambda\epsilon_j}\right).
\end{aligned}$$

Therefore, ϵ_{av}^{α} is upper bounded by

$$\inf_{t_0\in\{1,2,\ldots,\frac{n}{B}\}} \left[\frac{2BLR\alpha^l\lambda(T-t_0)}{n} \exp\left(\frac{(T-1-t_0)\lambda\epsilon}{1-2\lambda\epsilon}\right) \sum_{j=1}^{l} \tau_j \max_{t_0\leq t\leq T-1}\{\zeta_t^{(j)}\} + M_\alpha(\frac{Bt_0}{n})\right].$$

To complete the proof, we will use that $\max_{t_0 \leq t \leq T-1}\{\zeta_t^{(j)}\} \leq \sum_{t=t_0}^{T-1} \zeta_t^{(j)}$ and reverse the sum order. With the definition $\zeta_t := \sum_{j=1}^{l} \tau_j \zeta_t^{(j)}$, we then have

$$\epsilon_{av}^{\alpha} \leq$$
$$\inf_{t_0 \in \{1,2,\ldots,\frac{n}{B}\}} \left[\frac{2BLR\alpha^l \lambda (T - t_0)}{n} \exp\left(\frac{(T-1-t_0)\lambda\epsilon}{1-2\lambda\epsilon} \right) \sum_{t=t_0}^{T-1} \zeta_t + M_\alpha(\frac{Bt_0}{n}) \right].$$

Proof of Theorem 5.7: The beginning of the proof is the same as the proof of Theorem 5.4. However, in the case where smoothness is not assume (for example, ReLU neural networks), it is not possible to exploit the property of layer-wise smoothness. Instead, only the triangular inequality is used to bound $||\nabla^{(j)} L_{B_t}(W_t) - \nabla^{(j)} L_{B_t'}(W_t')||$. This leads to the inequality

$$\hat{\Delta}_{t+1}^{(j)} \leq \hat{\Delta}_t^{(j)} + 2\lambda_t \zeta_t^{(j)}.$$

Solving the recursion then yields

$$\hat{\Delta}_T^{(j)} \leq 2 \sum_{t=t_0}^{T-1} \lambda_t \zeta_t^{(j)}.$$

As a consequence,

$$\epsilon_{av}^{\alpha} \leq \inf_{t_0 \in \{1,2,\ldots,\frac{n}{B}\}} \left[2LR\alpha^l \sum_{t=t_0}^{T-1} \lambda_t \zeta_t + M_\alpha \frac{Bt_0}{n} \right],$$

where $\zeta_t = \sum_{j=1}^{l} \tau_j \zeta_t^{(j)}$, concluding the proof.

Appendix C. More Experiments

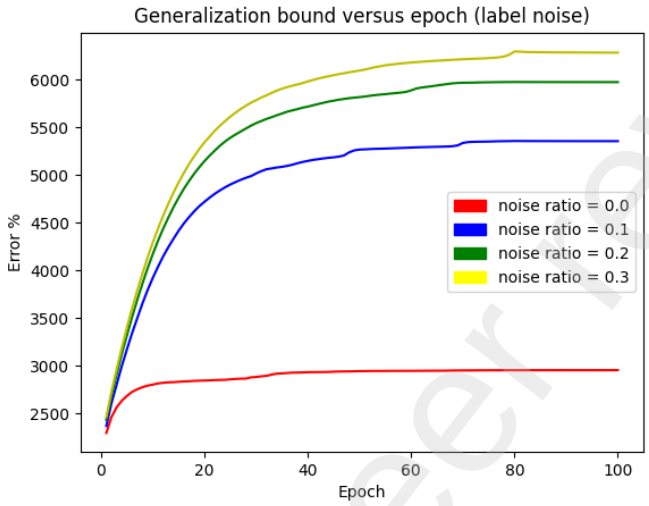


Figure C.5: The generalization bound from Theorem 5.8 ($\alpha = 1.0$) when training with different amounts of label noise. The network is a 6-layer fully connected network trained on MNIST with SGD. The final test accuracies are: 98.56% (label noise ratio = 0.0), 96.06% (label noise ratio = 0.1), 91.28% (label noise ratio = 0.2) and 84.22% (label noise ratio = 0.3).

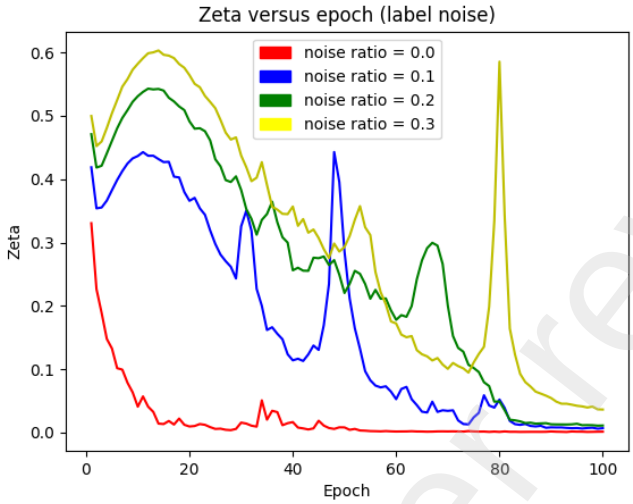


Figure C.6: $\hat{\zeta}_t(S)$ when training with different amounts of label noise. The network is a 6-layer fully connected network trained on MNIST with SGD.

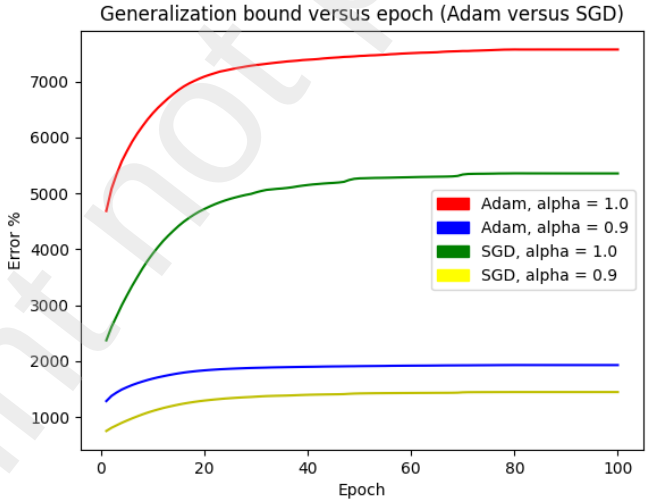


Figure C.7: The generalization bound from Theorem 5.8 ($\alpha = 1.0$ and $\alpha = 0.9$) is estimated when training with Adam and SGD. The network is a 6-layer fully connected network trained on MNIST with 10% of label noise. For Adam the test accuracy is 95.38% and for SGD, the test accuracy is 96.06%. Those are the best test accuracies that could be obtained after tuning the learning rate for each respective algorithm.

References

Bassily, R., Feldman, V., Guzmán, C., Talwar, K., 2020. Stability of stochastic gradient descent on nonsmooth convex losses, in: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (Eds.), Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.

Bousquet, O., Elisseeff, A., 2002. Stability and generalization. J. Mach. Learn. Res. 2, 499–526.

Elisseeff, A., Evgeniou, T., Pontil, M., 2005. Stability of randomized learning algorithms. J. Mach. Learn. Res. 6, 55–79.

Goyal, P., Dollár, P., Girshick, R.B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K., 2017. Accurate, large minibatch SGD: training imagenet in 1 hour. CoRR abs/1706.02677. URL: `http://arxiv.org/abs/1706.02677`, `arXiv:1706.02677`.

Hardt, M., Recht, B., Singer, Y., 2016. Train faster, generalize better: Stability of stochastic gradient descent, in: Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016, pp. 1225–1234. URL: `http://proceedings.mlr.press/v48/hardt16.html`.

He, F., Liu, T., Tao, D., 2019. Control batch size and learning rate to generalize well: Theoretical and empirical evidence, in: Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 1143–1152.

Hoffer, E., Hubara, I., Soudry, D., 2017. Train longer, generalize better: closing the generalization gap in large batch training of neural networks, in: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pp. 1731–1741.

Jastrzebski, S., Szymczak, M., Fort, S., Arpit, D., Tabor, J., Cho*, K., Geras*, K., 2020. The break-even point on optimization trajectories of deep neural networks, in: International Conference on Learning Representations. URL: `https://openreview.net/forum?id=r1g87C4KwB`.

Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P., 2017. On large-batch training for deep learning: Generalization gap and sharp minima, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. URL: https://openreview.net/forum?id=H1oyR1Ygg.

Krizhevsky, A., 2009. Learning multiple layers of features from tiny images. Technical Report.

Kuzborskij, I., Lampert, C.H., 2018. Data-dependent stability of stochastic gradient descent, in: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018, PMLR. pp. 2820–2829.

LeCun, Y., Bengio, Y., Hinton, G.E., 2015. Deep learning. Nature 521, 436–444. URL: https://doi.org/10.1038/nature14539, doi:10.1038/nature14539.

LeCun, Y., Cortes, C., 2010. MNIST handwritten digit database URL: http://yann.lecun.com/exdb/mnist/.

Lei, Y., Ying, Y., 2020. Fine-grained analysis of stability and generalization for stochastic gradient descent. CoRR abs/2006.08157. URL: https://arxiv.org/abs/2006.08157, arXiv:2006.08157.

Liao, Q., Miranda, B., Banburski, A., Hidary, J., Poggio, T.A., 2018. A surprising linear relationship predicts test performance in deep networks. CoRR abs/1807.09659. URL: http://arxiv.org/abs/1807.09659, arXiv:1807.09659.

Liu, T., Lugosi, G., Neu, G., Tao, D., 2017. Algorithmic stability and hypothesis complexity, in: Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, PMLR. pp. 2159–2167.

London, B., 2017. A pac-bayesian analysis of randomized learning with application to stochastic gradient descent, in: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA, pp. 2931–2940.

Lyu, K., Li, J., 2019. Gradient descent maximizes the margin of homogeneous neural networks. CoRR abs/1906.05890. URL: http://arxiv.org/abs/1906.05890, arXiv:1906.05890.

Mou, W., Wang, L., Zhai, X., Zheng, K., 2018. Generalization bounds of SGLD for non-convex learning: Two theoretical viewpoints, in: Bubeck, S., Perchet, V., Rigollet, P. (Eds.), Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018, PMLR. pp. 605–638. URL: http://proceedings.mlr.press/v75/mou18a.html.

Nacson, M.S., Srebro, N., Soudry, D., 2019. Stochastic gradient descent on separable data: Exact convergence with a fixed learning rate, in: The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan, PMLR. pp. 3051–3059.

Nagarajan, V., Kolter, J.Z., 2019. Uniform convergence may be unable to explain generalization in deep learning, in: Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pp. 11611–11622.

Neyshabur, B., Tomioka, R., Srebro, N., 2015. Norm-based capacity control in neural networks, in: Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015, pp. 1376–1401. URL: http://proceedings.mlr.press/v40/Neyshabur15.html.

Poggio, T.A., Banburski, A., Liao, Q., 2019. Theoretical issues in deep networks: Approximation, optimization and generalization. CoRR abs/1908.09375. URL: http://arxiv.org/abs/1908.09375, arXiv:1908.09375.

Richards, D., Rabbat, M., 2021. Learning with gradient descent and weakly convex losses. URL: https://arxiv.org/abs/2101.04968, doi:10.48550/ARXIV.2101.04968.

Shalev-Shwartz, S., Ben-David, S., 2014. Understanding Machine Learning - From Theory to Algorithms. Cambridge University Press.

Smith, S.L., Le, Q.V., 2018. A bayesian perspective on generalization and stochastic gradient descent, in: 6th International Conference on Learning

Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. URL: https://openreview.net/forum?id=BJij4yg0Z.

Soudry, D., Hoffer, E., Nacson, M.S., Srebro, N., 2018. The implicit bias of gradient descent on separable data, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. URL: https://openreview.net/forum?id=r1q7n9gAb.

Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O., 2017. Understanding deep learning requires rethinking generalization, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. URL: https://openreview.net/forum?id=Sy8gdB9xx.

Zhou, Y., Liang, Y., Zhang, H., 2022. Understanding generalization error of SGD in nonconvex optimization. Mach. Learn. 111, 345–375. URL: https://doi.org/10.1007/s10994-021-06056-w, doi:10.1007/s10994-021-06056-w.