# STABILITY ANALYSIS OF SGD THROUGH THE NORMALIZED LOSS FUNCTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We prove new generalization bounds for stochastic gradient descent for both the convex and non-convex case. Our analysis is based on the stability framework. 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 favorable 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.

## 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 & Elisseeff, 2002), (Elisseeff et al., 2005). It is argued in Nagarajan & 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 case. 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 investigate a measure of distance respecting invariances in ReLu 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 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 ReLu neural networks. We prove an upper bound on the on-average stability with respect to the normalized loss function which can then be used to give a generalization bound 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 result for neural networks involves 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 increasing the learning rate can lead to a trajectory keeping this quantity smaller during training. Therefore, larger learning rates can lead to a better "actual" stability than what a worst case analysis from uniform stability would indicate. There are two ways to get our data-dependent quantity smaller during training. The first one is by facilitating convergence (having smaller norm for the gradients). The second one 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. In our experiments, larger learning rates are favorable in both regards.

## 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. We might expect generalization bounds on the test error to be better by using the normalized surrogate loss in the analysis. (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 & 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 & Li (2019) provides and extension to deep non-linear homogeneous networks. Nacson et al. (2019) gives similar results for stochastic gradient descent. From the point of view of stability, starting from Hardt et al. (2016) without being exhaustive, a few representative examples are Liu et al. (2017), London (2017), Yuan et al. (2019), Kuzborskij & Lampert (2018).

Since the work of Zhang et al. (2017) showing that currently used deep neural networks are so much 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 & 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, instead of having the bounds involve properties of the initialization (which can be useful to investigate transfer learning), we maintain instead in our bound for neural networks the properties after the "burn-in" period and therefore closer to the final output since we are interested in the effect of the learning rate on the trajectory. 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".

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 & 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. Remark 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 classifier $w$, we will write $L_{\mathcal{D}}^{0-1}(w)$. Furthermore, we will add a superscript $\alpha$ when the normalized losses $l^\alpha$ are under consideration (these will be defined more clearly in the subsequent sections respectively for the convex case and the non-convex case). 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 the algorithm stochastic gradient descent with different batch sizes to minimize the empirical surrogate loss. The update rule of this algorithm for learning rates $\lambda_t$ 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 & 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 & Lampert (2018)). The definitions can be formulated with respect to any loss function but since we will study stability with respect to the $l^\alpha$ losses, we write the definitions in the context of this special case.

**Definition 1** *The algorithm $A$ is said to be $\epsilon_{uni}^\alpha$-uniformly stable if for all $i \in \{1, \dots, n\}$*

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

*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 2** *The algorithm $A$ is said to be $\epsilon_{av}^\alpha$-on-average stable if for all $i \in \{1, \dots, n\}$*

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

*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 Relu neural networks where we can rescale each layer without affecting the classifier. This is why we consider stability with respect to normalized losses instead. Remark 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 proof in Hardt et al. (2016) consists 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 $g$).

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.

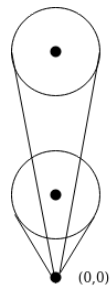



Figure 1: For the same magnitude of step taken (same ball radius), a larger norm of parameters leads to a smaller change in angle.

## 4 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 $\beta$-smoothness.

**Definition 3** *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** *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 1** *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 $\lambda_t$ starting from $w_0$. Denote by $\epsilon_{uni}^{\alpha}$ the uniform stability of this algorithm with respect to $l^{\alpha}$. Then,*

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

What is the difference between our bound and the bound in Hardt et al. (2016) (see theorem 6 in Appendix A) ? Our bound says that it is not enough to use a small learning rate 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 rate is small. 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).

From this result, it is possible to now 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 $A$.

[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 $\rho$ in their theorem 15 do not converge to 0 when the size of the training set grows to infinity.

**Theorem 2** *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 or equal to $1-\delta$ over draws of training sets $S$:*

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

**Proof:** The proof is an application of McDiarmid's concentration bound. Remark 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 & Elisseeff (2002) and we do not replicate it here. Remark 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, we can make the bound hold uniformly for all $\alpha$'s using standard techniques.

**Theorem 3** *Let $C > 0$. Assume that $l^\alpha(w,z)$ is a convex function of $\alpha$ for all $w, z$ and that $\epsilon_{uni}^{\alpha}$ is a non-decreasing function of $\alpha$. 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 $S$:*

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

In the next section, we investigate the non-convex case (actually we target directly neural networks). We exploit on-average stability to obtain a data-dependent quantity in the bound. Remark that it is also argued in Kuzborskij & Lampert (2018) that the worst case analysis of uniform stability might not be appropriate for deep learning. Finally, observe that if we use only one layer we get a linear classifier and so the results from the next section also applies to logistic regression for example.

## 5 Non-convex case

We now consider ReLu neural networks (although we could easily extend to other Lipschitz non-linearity) 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 $\sigma$ denotes the ReLu function. 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 $s, s', y$,

$$|l(s,y) - l(s',y)| \leq L||s - s'||. \tag{9}$$

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 $L = \sqrt{2}$. Remark that this is slightly different from the Lipschitz assumption of the previous section (given with respect to the weights $w$).

**Lemma 1** *Assume that $||x|| \leq R$. For $1 \leq j \leq l$, write $s_j = W_j(\sigma(\cdots W_2(\sigma(W_1 x))))$ and $s'_j = W'_j(\sigma(\cdots W'_2(\sigma(W'_1 x))))$. We have*

$$||s_l - s'_l|| \leq R \sum_{i=1}^{l} \left[ ||W_i - W'_i|| \prod_{j=1, j\neq i}^{l} \begin{pmatrix} ||W_i|| & if & j < i \\ ||W'_i|| & if & j > i \end{pmatrix} \right]. \tag{10}$$

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

**Definition 5** *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*

$$d(f,g) := \sum_{i=1}^{l} ||\frac{W_i}{||W_i||} - \frac{W'_i}{||W'_i||}||. \tag{11}$$

Remark that this distance function is invariant to rescaling the weights of any layer. This is a desirable property since in a ReLu 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 $l^{\alpha_1,\cdots\alpha_l}$ losses as

$$l^{\alpha_1,\cdots\alpha_l}(f,z) := l(\alpha_l \frac{W_l}{||W_l||}(\sigma(\cdots \alpha_2 \frac{W_2}{||W_2||}(\sigma(\alpha_1 \frac{W_1}{||W_1||}x)))), y), \tag{12}$$

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 $\alpha_i$ 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 $\alpha_i$'s are equal to say $\alpha$ and we will write $l^\alpha(f, z)$. From our definitions and lemma 1, we have that for all $z$ and neural networks $f$ and $g$,

$$|l^\alpha(f,z) - l^\alpha(g,z)| \leq LR\alpha^l d(f,g). \tag{13}$$

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

We will make two main assumptions to prove our main result. The first one is a modification of the concept of $\beta$-smooth functions for neural networks.

**Definition 6** *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 $W' = (W'_1, \cdots, W'_l)$,*

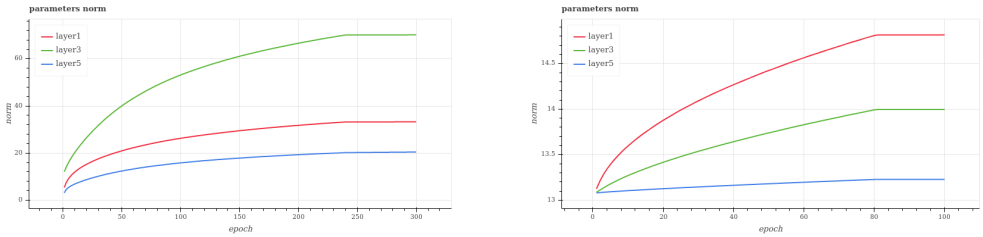
$$||\nabla^{(j)} l(W,z) - \nabla^{(j)} l(W',z)|| \leq \beta_j ||W_j - W'_j||. \tag{14}$$

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

Next, we introduce an assumption meaning informally that with high probability, the norm of parameters of each layer is eventually non-decreasing.

**Definition 7** *Let $\epsilon \geq 0$. The growing norms assumption (for this value of $\epsilon$) is defined as the following: The distribution $\mathcal{D}$ satisfies the property that there exists $t_0 \in \{0, 1, \cdots, \frac{n}{B}\}$ such that the probability over draws of training sets from $\mathcal{D}^n$ and over the randomness of $A$ of generating a non-decreasing sequence $||W_{j,t}||_{t=t_0}^T$ for all $j$ is greater or equal to $1 - \epsilon$.*

In Figure 2, we can see that on Cifar10 and Mnist this assumption makes sense (more details on experiments in section 6).




(a) Cifar10 (b) Mnist

Figure 2: Norm of parameters for 3 different layers when training a convolutional network on Cifar10 and a fully connected network on Mnist with SGD.

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

**Theorem 4** *Let $\epsilon \geq 0$ and assume that the distribution $\mathcal{D}$ satisfies the growing norms assumption for this value of $\epsilon$. The notation $t_0 \in \{1, 2, \cdots, \frac{B}{n}\}$ will now mean implicitly also that $t_0$ is large*

*enough so that the sequence of norms is non-decreasing. 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 $M_{\alpha}$. 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,*

$$\epsilon_{av}^{\alpha} \leq \inf_{t_0 \in \{1,2,\ldots,\frac{n}{B}\}} \left[ \frac{2BLR\alpha^l}{(n-B)\beta} \left( \frac{T-1}{t_0-1} \right)^{c\beta} \sum_{t=t_0}^{T-1} \zeta_t + M_{\alpha}(\frac{Bt_0}{n} + \epsilon) \right], \tag{15}$$

*where* $\zeta_t := \sum_{j=1}^{l} \mathbb{E}_{A,S,z} \frac{||\nabla^{(j)} L_{B_t}(W_t)||}{\min\{||W_{j,t}||, ||W'_{j,t}||\}}$ *and* $\beta = \max\{\beta_j\}$.

Exploiting theorem 12 in Elisseeff et al. (2005), we can get a probabilistic bound on the test error (holding over the randomness in the training sets and the randomness in the algorithm).

**Theorem 5** *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 $A$:*

$$L_{\mathcal{D}}^{0-1}(A(S)) \leq L_S^{\alpha}(A(S)) + \sqrt{\left(\frac{1}{\delta}\right) \frac{2M_{\alpha}^2 + 12nM_{\alpha}\epsilon_{av}^{\alpha}}{n}}. \tag{16}$$

# 6 Experiments

## 6.1 Learning rates and $\zeta_t$

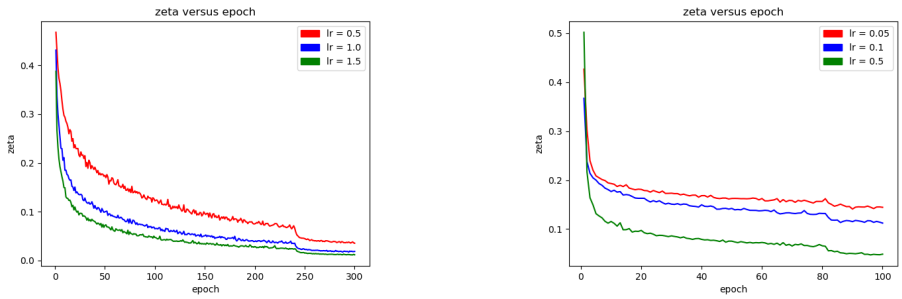
In this section we conduct some experiments on the datasets Cifar10 and Mnist. We consider the scenario where we try to reduce the performance gap between small batch and large batch training by increasing the learning rate. We will give some evidence suggesting that the quantity $\zeta_t$ can be of interest to assess generalization in this case.

Remark that the bound on stability for neural networks hold for learning rates satisfying $\lambda_t \leq c/t$. This is not a very practical schedule for deep learning and so we use a global learning rate being decayed one time by a factor of 10 instead in our experiments. We use no weight decay or momentum to stay closer to our theoretical analysis of SGD. Remark 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 result 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 $\zeta_t$ is relevant to investigate empirically. We approximate its value on a training set $S$ with the quantity $\hat{\zeta}_t(S) := \sum_{j=1}^{l} \frac{||\nabla^j L_{B_t}(W_t)||}{||W_{j,t}||}$. 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-layers convolutional 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 is reaching 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 3. We can see that it is better during all training when increasing the learning rate.

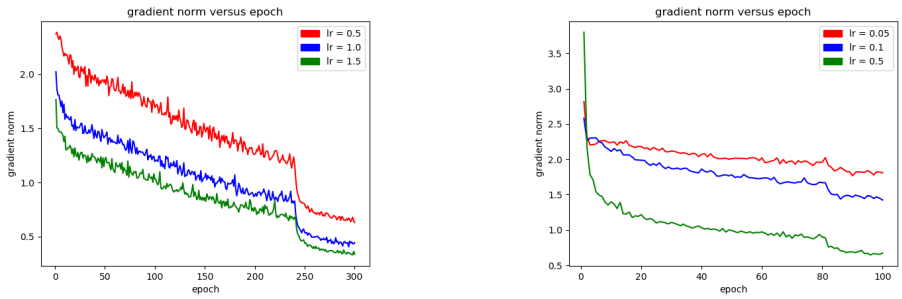
To compare with the analysis from Hardt et al. (2016), the quantity $\zeta_t$ would be replace with a global Lipschitz constant which would not be affected by the actual trajectory of the algorithm. Therefore,

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 $\zeta_t$ 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 4 and figure 5). Remark 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 & Lampert (2018) would not benefit from the norm of the paramters.



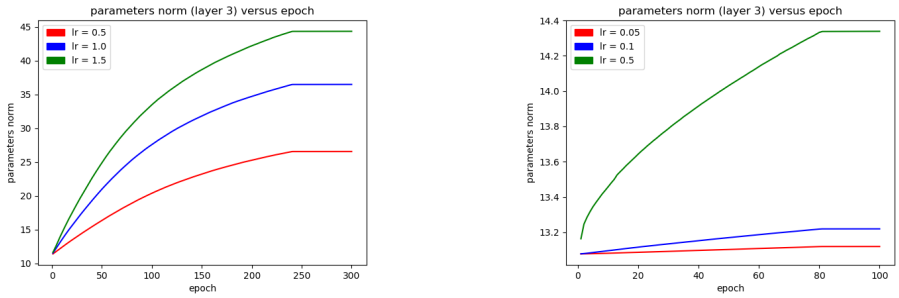

(a) Cifar10 (b) Mnist

Figure 3: $\hat{\zeta}_t(S)$ when training a convolutional network on Cifar10 and a fully connected network on Mnist.




(a) Cifar10 (b) Mnist

Figure 4: Norm of the gradient when training a convolutional network on Cifar10 and a fully connected network on Mnist.




(a) Cifar10 (b) Mnist

Figure 5: Norm of the parameters (layer 3) when training a convolutional network on Cifar10 and a fully connected network on Mnist.

### 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 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 $\epsilon_{av}$ 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 gradient obtained during training. For the normalized loss (our actual theorem 5), we upper bound $\epsilon_{av}^{\alpha}$ by $LR\alpha^{l}\mathbb{E}d(f,g)$ (see equation 13). 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 6.

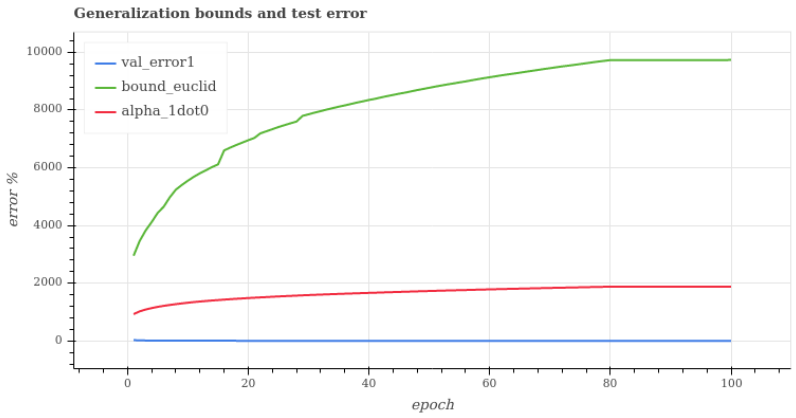



Figure 6: The bound obtained from the euclidean distance is much worst than the bound obtained from our normalized distance. The network is a 6-layer fully connected network and the training set is MNIST.

## 7 Conclusion

We investigated the 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. Our experimental results show that stability might not be as bad as expected when using larger learning rates in training deep neural networks. We hope that our analysis will be a helpful step in understanding generalization in deep learning. Future work could investigate the on-average stability with respect to $l^{\alpha}$ losses for different optimization algorithms.

## REFERENCES


Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, 2002.

André Elisseeff, Theodoros Evgeniou, and Massimiliano Pontil. Stability of randomized learning algorithms. *J. Mach. Learn. Res.*, 6:55–79, 2005.

Priya Goyal, Piotr Dollár, Ross B. Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: training imagenet in 1 hour. *CoRR*, abs/1706.02677, 2017. URL http://arxiv.org/abs/1706.02677.

Moritz Hardt, Ben Recht, and Yoram Singer. 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*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1225–1234, 2016. URL http://proceedings.mlr.press/v48/hardt16.html.

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

Elad Hoffer, Itay Hubara, and Daniel Soudry. 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, 2017.

Stanislaw Jastrzebski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho*, and Krzysztof Geras*. The break-even point on optimization trajectories of deep neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1g87C4KwB.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 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*, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

Ilja Kuzborskij and Christoph H. Lampert. 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*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2820–2829. PMLR, 2018.

Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539. URL https://doi.org/10.1038/nature14539.

Qianli Liao, Brando Miranda, Andrzej Banburski, Jack Hidary, and Tomaso A. Poggio. A surprising linear relationship predicts test performance in deep networks. *CoRR*, abs/1807.09659, 2018. URL http://arxiv.org/abs/1807.09659.

Tongliang Liu, Gábor Lugosi, Gergely Neu, and Dacheng Tao. Algorithmic stability and hypothesis complexity. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2159–2167. PMLR, 2017.

Ben London. 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, 2017.

Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *CoRR*, abs/1906.05890, 2019. URL http://arxiv.org/abs/1906.05890.

Mor Shpigel Nacson, Nathan Srebro, and Daniel Soudry. 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*, volume 89 of *Proceedings of Machine Learning Research*, pp. 3051–3059. PMLR, 2019.

Vaishnavh Nagarajan and J. Zico Kolter. 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, 2019.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. 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, 2015. URL `http://proceedings.mlr.press/v40/Neyshabur15.html`.

Tomaso A. Poggio, Andrzej Banburski, and Qianli Liao. Theoretical issues in deep networks: Approximation, optimization and generalization. *CoRR*, abs/1908.09375, 2019. URL `http://arxiv.org/abs/1908.09375`.

Samuel L. Smith and Quoc V. Le. 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*, 2018. URL `https://openreview.net/forum?id=BJij4yg0Z`.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, and Nathan Srebro. 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*, 2018. URL `https://openreview.net/forum?id=r1q7n9gAb`.

Zhuoning Yuan, Yan Yan, Rong Jin, and Tianbao Yang. Stagewise training accelerates convergence of testing error over SGD. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 2604–2614, 2019.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. URL `https://openreview.net/forum?id=Sy8gdB9xx`.

## A Appendix: Proofs for the convex case

We start by proving a few lemmas.

**Lemma 2** *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 3** *Assume that the update rule $G$ is $\eta-$expansive. Furthermore, assume that $||G(v)||$ and $||G(w)||$ are larger than $\hat{K}\eta$. Then,*

$$||\frac{G(w)}{||G(w)||} - \frac{G(v)}{||G(v)||}|| \leq \frac{||v-w||}{\hat{K}}.$$

**Proof:** For $v$ and $w$ satisfying the assumptions, we have

$$||\frac{G(v)}{||G(v)||} - \frac{G(w)}{||G(w)||}|| \leq \frac{||G(v)-G(w)||}{\hat{K}\eta} \leq \frac{||v-w||}{\hat{K}}.$$

**Lemma 4** *Assume that the update rule is $\sigma-$bounded, that is $||G(w) - w|| \leq \sigma$ for all $w$. Furthermore, assume that $||G(v)|| \geq \hat{K}$ and $||G(w)|| \geq \hat{K}$. Then,*

$$||\frac{G(w)}{||G(w)||} - \frac{G(v)}{||G(v)||}|| \leq \frac{||v-w||}{\hat{K}} + 2\frac{\sigma}{\hat{K}}.$$

**Lemma 5** *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 $\lambda_t$ 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 6** *(Theorem 3.8 in Hardt et al. (2016)) Assume that the loss function $f(\cdot; z)$ is $\beta$-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 proposition 1.

**Proof of theorem 1:** 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)). Furthermore, it follows directly form our lemma 5 that the assumptions in both lemma 3 and lemma 4 are satisfied (we can use $\sigma = L\lambda_t$ at iteration $t$ in lemma 4). Since the probability of

picking the example $i$ in a mini-batch of size $B$ is smaller than $\frac{B}{n}$ (sampling with replacement), we have

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

Remark that this is true since $\frac{\mathbb{E}||w_t-w'_t||}{\hat{K}} \leq \frac{\mathbb{E}||w_t-w'_t||}{\hat{K}}+2\frac{L\lambda_t}{\hat{K}}$. From the result in Hardt et al. (2016) (with a slight change to allow any batch size $B$), we have

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

Therefore,

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

The result then follows from the inequality

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

We finally prove theorem 3.

**Proof of theorem 3:** 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

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

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 $\epsilon_{stab}^{\alpha}$ and $M_\alpha$ are non-decreasing functions of $\alpha$ 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 $\alpha$. Indeed, since $\frac{\alpha}{2} \leq \alpha_i \leq \alpha$, we must have

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

## B Appendix: Proofs for the non-convex case

**Proof of lemma 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,

$$\begin{aligned}
||s_l - s_l'|| &= ||W_l\sigma(s_{l-1}) - W_l'\sigma(s_{l-1}')|| \\
&= ||W_l\sigma(s_{l-1}) - W_l'\sigma(s_{l-1}) - W_l'(\sigma(s_{l-1}') - \sigma(s_{l-1}))|| \\
&\leq ||W_l\sigma(s_{l-1}) - W_l'\sigma(s_{l-1})|| + ||W_l'(\sigma(s_{l-1}') - \sigma(s_{l-1}))|| \\
&\leq ||W_l - W_l'||\,||\sigma(s_{l-1})|| + ||W_l'||\,||\sigma(s_{l-1}') - \sigma(s_{l-1})|| \\
&\leq ||W_l - W_l'||\,||s_{l-1}|| + ||W_l'||\,||s_{l-1}' - s_{l-1}|| \\
&\leq R||W_l - W_l'|| \prod_{i=j}^{l-1} ||W_j|| + ||W_l'||\,||s_{l-1}' - s_{l-1}|| \\
&\leq R||W_l - W_l'|| \prod_{j=1}^{l-1} ||W_j|| + ||W_l'||\, R\sum_{i=1}^{l-1}\left[||W_i - W_i'|| \prod_{j=1, j\neq i}^{l-1} \begin{pmatrix} ||W_i|| & if & j<i \\ ||W_i'|| & if & j>i \end{pmatrix}\right] \\
&= R\sum_{i=1}^{l}\left[||W_i - W_i'|| \prod_{j=1, j\neq i}^{l} \begin{pmatrix} ||W_i|| & if & j<i \\ ||W_i'|| & if & j>i \end{pmatrix}\right].
\end{aligned}$$

We exploited the fact that the ReLu non-linearity cannot increase the norm of a vector and also that it is 1-Lipschitz. The proof is finally concluded by observing that for one layer we have, $||s_1' - s_1|| \leq R||W_1 - W_1'||$.

**Definition 8** *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 $\hat{\Delta}_t^{(j)}(S,z) := \mathbb{E}_A[\hat{\delta}_t^{(j)}(S,z)\,|\,\forall k,\ \delta_{t_0}^{(k)}(S,z) = 0]$, where $K_t^{(j)}(S,z) := \min\{||W_{j,t}||, ||W_{j,t}'||\}$.*

Before proving theorem 4, we establish a lemma. Remark that the structure of the proof of the following lemma and of theorem 4 is similar to the corresponding results in Hardt et al. (2016) and in Kuzborskij & Lampert (2018).

**Lemma 6** *Let $\epsilon \geq 0$ and assume that the distribution $\mathcal{D}$ satisfies the growing norms assumption for this value of $\epsilon$. The notation $t_0 \in \{0, 1, 2, \cdots, \frac{B}{n}\}$ will now mean implicitly also that $t_0$ is large enough so that the sequence of norms is non-decreasing. 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 $M_\alpha$. Also, assume that $||x|| \leq R$. 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*

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

**Proof:** Since $l^\alpha(f, z)$ is bounded above by $M_\alpha$ and non-negative, we have

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

Therefore, by adding a term $M_\alpha \epsilon$ to the bound, we can then only consider the case where the growing norm assumption is satisfied. We won't write that the expectation is then conditional on that assumption to lighten the statements. By a similar line of reasoning, we can further condition inside the expectation with the property that after $t_0$ iterations, the two networks are still exactly the same. To make this clearer, 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

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

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

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

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 4:** From lemma 2, we always have $\tilde{\delta}_t^{(j)}(S, z) \leq \hat{\delta}_t^{(j)}(S, z)$. Therefore, from the previous lemma,

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

First remark that under our assumption,

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

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

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

Define $\hat{\Delta}_t^{(j)} := \mathbb{E}_{S,z}\hat{\Delta}_t^{(j)}(S,z)$ and $\zeta_t^{(j)} := \mathbb{E}_{A,S,z}\frac{||\nabla^{(j)} L_{B_t}(W_t)||}{K_t^{(j)}(S,z)}$ for any $t$. Taking the expectation over $S$ and $z$ on both side 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 $\mathbb{E}_{A,S,z}\frac{||\nabla^{(j)} L_{B_t}(W_t)||}{K_t^{(j)}(S,z)} = \mathbb{E}_{A,S,z}\frac{||\nabla^{(j)} L_{B'_t}(W'_t)||}{K_t^{(j)}(S,z)}$. 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, we get

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

Therefore,

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

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 $\zeta_t := \sum_{j=1}^{l}\zeta_t^{(j)}$, we then have

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