Why neural networks find simple solutions: the many regularizers of geometric complexity

Anonymous Author(s)
Affiliation
Address
email

Abstract

In many contexts, simpler models are preferable to more complex models and the control of this model complexity is the goal for many methods in machine learning such as regularization, hyperparameter tuning and architecture design. In deep learning, it has been difficult to understand the underlying mechanisms of complexity control, since many traditional measures are not naturally suitable for deep neural networks. Here we develop the notion of geometric complexity, which is a measure of the variability of the model function, computed using a discrete Dirichlet energy. Using a combination of theoretical arguments and empirical results, we show that many common training heuristics such as parameter norm regularization, spectral norm regularization, flatness regularization, implicit gradient regularization, noise regularization and the choice of parameter initialization all act to control geometric complexity, providing a unifying framework in which to characterize the behavior of deep learning models.

1 Introduction

Regularization is an essential ingredient in the deep learning recipe and understanding its impact on the properties of the learned solution is a very active area of research [28, 36, 62, 73]. Regularization can assume a multitude of forms, either added explicitly as a penalty term in a loss function [28] or implicitly through our choice of hyperparameters [6, 65, 68, 70, 72], model architecture [49, 51, 52] or initialization [26, 31, 32, 50, 59, 81, 82]. These forms are generally not designed to be analytically tractable, but in practice, regularization is often invoked in the control of model complexity, putting a pressure on a model to discover simple solutions more so than complex solutions.

To understand regularization in deep learning, we need to precisely define model 'complexity' for deep neural networks. Complexity theory provides many techniques for measuring the complexity of a model, such as a simple parameter count, or a parameter norm measurement [4, 18, 54, 61] but many of these measures can be problematic for neural networks [38, 79]. The recently observed phenomena of 'double-descent' [8, 9, 60] illustrates this clearly: neural networks with high model complexity, as measured by a parameter count, can fit training data closely (sometimes interpolating the data exactly), while simultaneously having low test error [60, 79]. Classically, we expect that interpolation of training data is evidence of overfitting, but yet, neural networks seem to be capable of interpolation while also having low test error. It is often suggested that some form of implicit regularization or explicit regularization is responsible for this, but how should we account for this in theory, and what complexity measure is most appropriate?

In this work, we develop a measure of model complexity, called Geometric Complexity (GC), that has properties that are suitable for the analysis of deep neural networks. We use theoretical and empirical techniques to demonstrate that many different forms of regularization and other training heuristics

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

can act to control geometric complexity through different mechanisms. We argue that the geometric complexity provides a convenient proxy for neural network performance.

Our primary contributions are:

- We develop a computationally tractable notion of complexity (Section 2), which we call *Geometric Complexity* (GC), that has many close relationships with many areas in deep learning and mathematics including harmonic function theory, Lipschitz smoothness (Section 2), and regularization theory (Section 4).
- We provide evidence that common training heuristics keep the geometric complexity low, including: (i) common initialization schemes (Section 3) (ii) the use of overparametrized models with a large number of layers (Fig. 2) (iii) large learning rates, small batch sizes, and implicit gradient regularization (Section 5 and Fig. 4) (iv) explicit parameter norm regularization, spectral norm regularization, flatness regularization, and label noise regularization (Section 4 and Fig. 3)
- We show that the geometric complexity captures the double-descent behaviour observed in the test loss as model parameter count increases (Section 6 and Fig. 5).

The aim of this paper is to introduce geometric complexity, explore its properties and highlight its connections with existing implicit and explicit regularizers. To disentangle the effects studied here from optimisation choices, we use stochastic gradient descent without momentum to train all models. We also study the impact of a given training heuristic on geometric complexity in isolation of other techniques to avoid masking effects. For this reason we do not use data augmentation or learning rate schedules in the main part of the paper. In the Supplementary Material (SM) we redo most experiments using SGD with momentum (Section C.7) and Adam (Section C.8) with very similar conclusion. We also observe the same behavior of the geometric complexity in a setting using learning rate schedule, data augmentation, and explicit regularization in conjunction to improve model performance (Section C.6). The exact details of all experiments in the main paper are listed in SM Section B. All additional experiment results and details can be found in SM Section C.

2 Geometric complexity and Dirichlet energy

Although many different forms of complexity measures have been proposed and investigated (e.g., [17, 38, 61]), it is not altogether clear what properties they should have, especially for deep learning. For instance, a number of them like the Rademacher complexity [44], the VC dimension [76], or the simple model parameter count focus on measuring the entire hypothesis space, rather than a specific function, which can be problematic in deep learning [38, 79]. Other measures like the number of linear pieces for ReLU networks [3, 69] or various versions of the weight matrix norms [61] measure the complexity of the model function independently from the task at hand, which is not desirable [64]. An alternative approach is to learn a complexity measure directly from data [48] or to take the whole training procedure over a dataset into account [60]. Recently, other measures focusing on the model function complexity over a dataset have been proposed in [22] and [54] to help explain the surprising generalization power of deep neural networks. Following that last approach and motivated by frameworks well established in the field of geometric analysis [40], we propose a definition of complexity related to the theory of harmonic functions and minimal surfaces. Our definition has the advantage of being computationally tractable and implicitly regularized by many training heuristics in the case of neural networks. It focuses on measuring the complexity of individual functions rather than that of the whole function space, which makes it different from the Radamacher or VC complexity.

Definition 2.1. *Let $g_\theta : \mathbb{R}^d \to \mathbb{R}^k$ be a neural network parameterized by θ. We can write $g_\theta(x) = a(f_\theta(x))$ where a denotes the last layer activation, and f_θ its logit network. The GC of the network over a dataset D is defined to be the discrete Dirichlet energy of its logit network:*

$$\langle f_\theta,\, D\rangle_G = \frac{1}{|D|}\sum_{x\in D} \|\nabla_x f_\theta(x)\|_F^2, \tag{1}$$

where $\|\nabla_x f_\theta(x)\|_F$ is the Frobenius norm of the network Jacobian.

Note that this definition is well-defined for any differentiable model, not only a neural network, and incorporates both the model function and the dataset over which the task is determined. Next, we discuss how GC relates to familiar concepts in deep learning.

Geometric complexity and linear models: Consider a linear transformation $f(x) = Ax + b$ from $\mathbb{R}^d$ to $\mathbb{R}^k$ and a dataset $D = \{x_i\}_{i=1}^N$ where $x_i \in \mathbb{R}^d$. At each point $x \in D$, we have that $\|\nabla_x f(x)\|_F^2 = \|A\|_F^2$, hence the GC for a linear transformation is $\langle f_\theta, D\rangle_G = \|A\|_F^2$. Note, this implies that GC for linear transformations (and more generally, affine maps), is independent of the dataset D and zero for constant functions. Furthermore, note that the GC in this setting coincides precisely with the L2 norm of the model weight matrix. Thus, enforcing an L2 norm penalty is equivalent to regularizing the GC for linear models (see Section 4 for more on that point).

Geometric complexity and ReLU networks: For a ReLU network $g_\theta : \mathbb{R}^d \to \mathbb{R}^k$ as defined in Definition 2.1, the GC over a dataset D has a very intuitive form. Since a ReLU network parameterizes piece-wise linear functions [3], the domain can be broken into a partition of subsets $X_i \subset \mathbb{R}^d$ where f_θ is an affine map $A_i x + b_i$. Now denote by D_i the points in the dataset D that fall in the linear piece defined on X_i. For every point x in X_i, we have that $\|\nabla_x f_\theta(x)\|_F^2 = \|A_i\|_F^2$. Since the D_i's partition the dataset D, we obtain

$$\langle f_\theta, D\rangle_G = \sum_i \left(\frac{n_i}{|D|}\right) \|A_i\|_F^2, \tag{2}$$

where n_i is the number of points in the dataset D falling in X_i. We see from Eqn. (2) that for ReLU networks the GC over the whole dataset coincides exactly with the GC on a batch $B \subset D$, provided that the proportion of points in the batch falling into each of the the linear pieces are preserved. This makes the evaluation of the GC on large enough batches a very good proxy to the overall GC over the dataset, and computationally tractable during training.

Geometric complexity and Lipschitz smoothness: One way to measure the smoothness of a function $f : \mathbb{R}^d \to \mathbb{R}^k$ on a subset $X \subset \mathbb{R}^d$ is by its Lipschitz constant; i.e., the smallest $f_L \geq 0$ such that $\|f(x_1) - f(x_2)\| \leq f_L \|x_1 - x_2\|$, for all $x_1, x_2 \in X$. Intuitively, the constant f_L measures the maximal amount of variation allowed by f when the inputs change by a given amount. Using the Lipschitz constant, one can define a complexity measure of a function f as the Lipschitz constant f_L of the function over the input domain $\mathbb{R}^d$. Since $\|\nabla_x f(x)\|_F^2 \leq \min(k, d)\|\nabla_x f(x)\|_{op}^2 \leq \min(k, d) f_L^2$ where $\|\cdot\|_{op}$ is the operator norm, we obtain a general bound on the GC by the Lipschitz complexity:

$$\langle f, D\rangle_G \;=\; \frac{1}{|D|} \sum_{x \in D} \|\nabla_x f(x)\|_F^2 \;\leq\; \min(k, d) f_L^2. \tag{3}$$

While the Lipschitz smoothness of a function provides an upper bound on GC, there are a few fundamental differences between the two quantities. Firstly, GC is data dependent: a model can have low GC while having high Lipschitz constant due to the model not being smooth in parts of the space where there is no training data. Secondly, the GC can be computed exactly given a model and dataset, while for neural networks only loose upper bounds are available to estimate the Lipschitz constant.

Geometric complexity, arc length and harmonic maps: Let us start with a motivating example: Consider a dataset consisting of 10 points lying on a parabola in the plane and a large ReLU deep neural network trained via gradient descent to learn a function $f : \mathbb{R} \to \mathbb{R}$ that fits this dataset (Fig. 1). Throughout training the model function seems to attain minimal arc length for a given level of training error. Recalling the formula for arc length of f, which is the integral of over $[-1, 1]$ of $\sqrt{1 + f'(x)^2}$, and using the Taylor approximation $\sqrt{1 + x^2} \approx 1 + \frac{x^2}{2}$, it follows that minimizing the arc length is equivalent to minimizing the classic Dirichlet energy:

$$E(f) = \frac{1}{2} \int_\Omega \|\nabla_x f(x)\|^2 dx. \tag{4}$$

where $\Omega = [-1, 1]$. The Dirichlet energy can be thought intuitively of as a measure of the variability of the function f on Ω. Its minimizers, subject to a boundary condition $f_{|\partial\Omega} = h$, are called *harmonic maps*, which are maps causing the "least intrinsic stretching" of the domain Ω [74]. The geometric

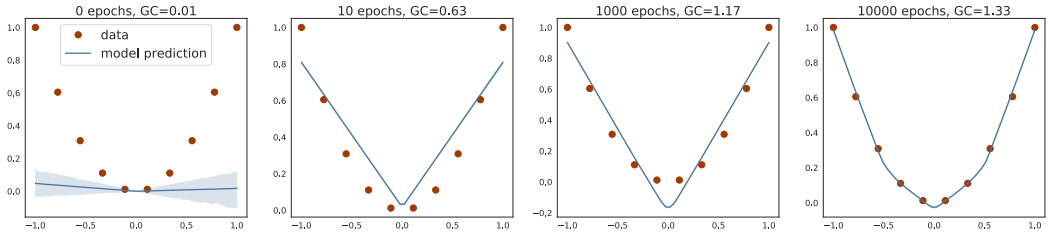

Figure 1: For a large MLP fitting 10 points, the complexity of the function being learned gradually grows in training, while avoiding unnecessary complexity by keeping the function arc length minimal.

complexity in Definition 2.1 is an unbiased estimator of a very related quantity

$$\mathbb{E}_X(\|\nabla_x f_\theta(x)\|_F^2) = \int_{\mathbb{R}^d} \|\nabla_x f_\theta(x)\|_F^2 \, p_X(x) dx, \tag{5}$$

where the domain Ω is replaced by the probability distribution of the features p_X and the boundary condition is replaced by the dataset D. We could call the quantity defined in Eqn. (5) the *theoretical geometric complexity* as opposed to the *empirical geometric complexity* in Definition 2.1.

In the remaining sections we provide evidence that common training heuristics do indeed keep the GC low, encouraging neural networks to find intrinsically simple solutions.

3 Impact of initialization on geometric complexity

Parameter initialization choice is an important factor in deep learning. Although we are free to specify exact parameter initialization values, in practice, a small number of default initialization schemes have emerged to work well across a wide range of tasks [26, 32]. Here, we explore the relationship between some of these initialization schemes and GC.

To begin, consider the one dimensional regression example that we introduced in Figure 1. In this experiment, we employed a standard initialization scheme to initialise the parameters: we sample them from a truncated normal distribution with variance inversely proportional to the number of input units and the bias terms were set to zero. We observe that the initialised function on the interval $[-1, 1]$ is very close to the zero function (Fig. 1), and the zero function has zero GC. This observation suggests that initialization schemes that have low initial GC are useful for deep learning.

To explore this further, we consider deep ReLU networks with larger input and output spaces initialized using the same scheme as above, and measure the GC of the resulting model. Specifically, consider the initialized ReLU network given by $f_{\theta_0} : \mathbb{R}^d \to \mathbb{R}^k$ with $d = 150528$ and $k = 1000$, with parameter initialization θ_0, and varying network depth. We measure the ReLU network output size by recording the mean and maximum output values, evaluated using input values along the

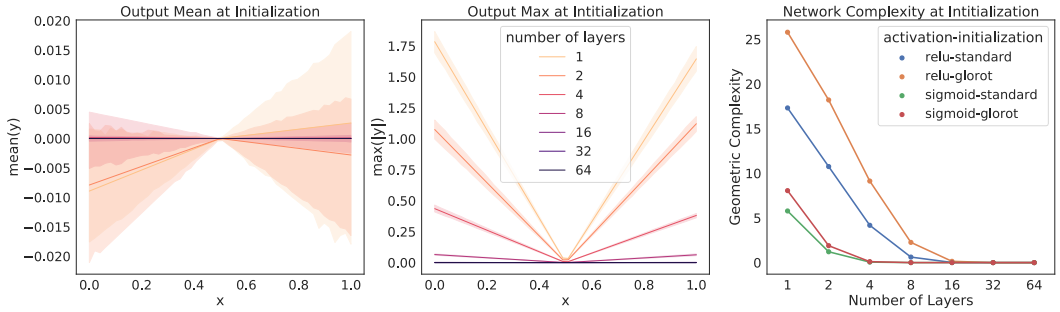

Figure 2: MLP's given by $y = f_{\theta_0}(x)$ initialize closer to the zero function and closer to zero GC, as the number of layers increases. **Left and Middle:** The ReLU MLP's initialized with the standard scheme are evaluated along a range of normalised input values. **Right:** GC is computed on a dataset D of 100 normalized data points. All MLP's have 500 neurons per layer.

line $P_1 + (P_2 - P_1)x$ with $x \in [0, 1]$ between two diagonal points P_1 and P_2 of the hyper-cube $[-1, 1]^d$. We observe that these ReLU networks initialize to functions close to the zero function, and become progressively closer to a zero valued function as the number of layer increases (Fig. 2). For ReLU networks, this is not entirely surprising. With biases initialised to zero, we can express a ReLU network in a small neighborhood of a given point $x \in \mathbb{R}^d$ as a product of matrices $f_{\theta_0}(x) = W_1 P_2 W_2 P_3 W_3 \cdots P_l W_l x$, where the W_i's are the weight matrices and the P_i's are diagonal matrices with 0 and 1 on their diagonals. This representation makes it clear that at initialization the ReLU network passes through the origin; i.e., $f_{\theta_0}(0) = 0$. Furthermore, with weight matrices initialised around zero, using a scaling that can reduce the spread of the distribution as the matrices grow, we can expect that deeper ReLU networks generated by multiplying a large number of small-valued weights, can produce output values close to zero (for input values taken from a hyper-cube $[-1, 1]^d$). We extend these results further to include additional initialization schemes and experimentally confirm that the GC can be brought close to zero with a sufficient number of layers. In fact, this is true not only for ReLU networks with the standard initialization scheme, but for a number of other common activation functions and initialization setups [26, 32], and even on domains much larger than the normalized hyper-cube (Fig. 2 and SM Section C.1).

4 Impact of explicit regularization on geometric complexity

Next, we explore the relationship between GC and various forms of explicit regularization using a combination of theoretical and empirical results.

L2 regularization: In L2 regularization, the Euclidean norm of the parameter vector is explicitly added to a loss function, so as to penalize solutions that have excessively large parameter values. This is one of the simplest, and most widely used forms of regularization. For linear models ($f_\theta(x) = Ax + b$), we saw in Section 2 that the GC coincides with the Frobenius norm of the matrix A. This means that standard L2 norm penalties on the weight matrix coincide in this case with a direct explicit regularization of the GC of the linear model. For non-linear deep neural networks, we cannot directly identify the L2 norm penalty with the model GC. However, we can demonstrate this relationship empirically, by training a selection of neural networks with L2 regularization, each with a different regularization strength. We measure the GC for each network at the time of maximum test accuracy. We observe empirically that strengthening L2 regularization coincides with a decrease in GC values (Fig. 3).

Lipschitz regularization via spectral norm regularization: A number of explicit regularization schemes have been used to tame the Lipschitz smoothness of the model function and produce smoother models with reduced test error [2, 3, 19, 21, 30, 43, 57, 64, 78]. Smoothness regularization has also been shown to be beneficial outside the supervised learning context, in GANs [12, 57, 80] and reinforcement learning [10, 27]. One successful approach to regularising the Lipschitz constant of a neural network with 1-Lipschitz activation functions (e.g. ReLU, ELU) is to constrain the spectral norm of each layer of the network (i.e., the maximal singular values $\sigma_{\max}(W_i)$ of the weight matrices W_i), since the product of the spectral norms of the networks weight matrices is an upper bound to the Lipschitz constant of the model: $f_L \leq \sigma_{\max}(W_1) \cdots \sigma_{\max}(W_l)$. Using inequality (3), we see

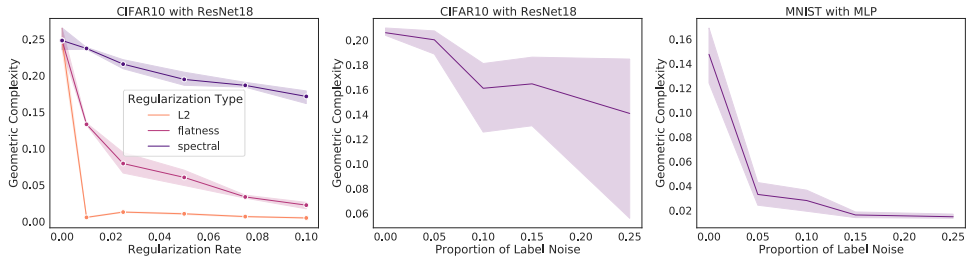

Figure 3: Explicit regularization and GC. **Left:** As the L2, flatness, and spectral regularization increase, GC decreases. (See SM Section C.5 for additional experiments on explicit regularization with ranges targeted to each regularization type.) **Middle and Right:** As label noise regularization increases, GC decreases.

that any approach that constrains the Lipschitz constant of the model constrains GC. To confirm this theoretical prediction, we train a ResNet18 model on CIFAR10 [46] and regularize using spectral regularization [78], an approach which adds the regularizer $\frac{\lambda}{2}\sum_l(\sigma_{\max}(W_l))^2$ to the model loss function. We observe that GC decreases as the strength of spectral regularization λ increases (Fig. 3).

Noise regularization: The addition of noise during training is known to be an effective form of regularization. For instance, it has been demonstrated [11] that the addition of noise to training labels during the optimisation of a least-square loss using SGD exerts a regularising pressure on $\sum_{(x,y)\in D}\|\nabla_\theta f_\theta(x)\|^2/|D|$. Here, we demonstrate empirically that the GC of a ResNet18 trained on CIFAR10 reduces as the proportion of label noise increases (Fig. 3, middle). The same is true of an MLP trained on MNIST [15] (Fig. 3, right). In Section 5, we provide a theoretical argument which justifies these experiments, showing that a regularizing pressure on $\|\nabla_\theta f_\theta(x)\|^2$ transfers to a regularizing pressure on $\|\nabla_x f_\theta(x)\|^2$. Thus, label noise in SGD in turn translates into a regularizing pressure on the GC in the case of neural networks.

Flatness regularization: In flatness regularization, an explicit gradient penalty term, $\|\nabla_\theta L_B\|^2$ is added to the loss (where L_B is the loss evaluated across a batch B). It has been observed in practice that flatness regularization can be effective in many deep learning settings, from supervised learning [23, 70] to GAN training [5, 56, 58, 63, 65]. Flatness regularization penalizes learning trajectories that follow steep slopes across the loss surface, thereby encouraging learning to follow shallower slopes toward flatter regions of the loss surface. We demonstrate empirically that GC decreases as the strength of flatness regularization increases (Fig. 3). In the next section, we will provide a theoretical argument that flatness regularization can control GC.

Explicit GC regularization and Jacobian regularization: All the forms of regularization above have known benefits for improving the test accuracy in deep learning. As we saw, they all also implicitly regularize GC. This raises the question as whether regularizing for GC directly and independently of any other mechanism is sufficient to improve model performance. Namely, we can add GC computed on the batch to the loss $L_{\text{reg}}(\theta) = L_B(\theta) + \alpha/B\sum_{x\in B}\|\nabla_x f_\theta(x)\|_F^2$. This is actually a known form of explicit regularization, called **Jacobian regularization**, which has been correlated with increased generalization but also robustness to input shift [36, 73, 77, 78]. [73] specifically shows that adding Jacobian regularization to the loss function can lead to an increase in test set accuracy (their Tables III, IV, and V). In SM Section C.4 we train a MLP on MNIST and a ResNet18 on CIFAR10 regularized explicitly with the geometric complexity. We observe an increase of test accuracy and a decrease of GC with more regularization. Related regularizers include *gradient penalties* of the form $\sum_x(||\nabla_x f_\theta(x)|| - K)^2$ which have been used for GAN training [21, 30, 42]. We leave the full investigation of the importance of GC outside supervised learning for future work.

5 Impact of implicit regularization on geometric complexity

Implicit regularization is a hidden form of regularization that emerges as a bi-product of model training. Unlike explicit regularization, it is not explicitly added to a loss function. In deep learning settings where no explicit regularization is used, it is the only form of regularization. Here, we use a combination of theoretical and empirical results to argue that some recently identified implicit regularization mechanisms in gradient descent [6, 53, 70] exert a regularization pressure on geometric complexity. Our argument proceeds as follows: 1) we identify a mathematical term (the implicit gradient regularization term) that characterizes implicit regularization in gradient descent, 2) we demonstrate that this term depends on model gradients, 3) we identify the conditions where model gradient terms apply a regularization pressure on geometric complexity.

Step 1: The implicit regularization that we consider emerges as a bi-product of the discrete nature of gradient descent updates. In particular it has been shown that a discrete gradient update $\theta' = \theta - h\nabla_\theta L_B(\theta)$ over a batch of data implicitly minimizes a modified loss, $\widetilde{L}_B = L_B + \frac{h}{4}\|\nabla_\theta L_B\|^2$, where the second term is called the Implicit Gradient Regularizer (IGR) [6]. Gradient descent optimisation is better characterized as a continuous flow along the gradient of the modified loss,

rather than the original unmodified loss. By inspection, the IGR term implicitly regularizes training toward trajectories with smaller loss gradients toward flatter region on the loss surface [6] .

Step 2: Next, we develop this implicit regularizer term for a multi-class classification cross-entropy loss term. We can write (see SM Section A.2 for details):

$$\widetilde{L}_B = L_B + \frac{h}{4B}\left(\frac{1}{B}\sum_{x,i}\epsilon_x^i(\theta)^2\|\nabla_\theta f_\theta^i(x)\|^2\right) + \frac{h}{4}A_B(\theta) + \frac{h}{4}C_B(\theta), \tag{6}$$

where $C_B(\theta)$ measures the *batch gradient alignment*

$$C_B(\theta) = \frac{1}{B^2}\sum_{(x,y)\neq(x',y')}\langle\nabla_\theta L(x,y,\theta), \nabla_\theta L(x',y',\theta)\rangle\,. \tag{7}$$

and $A_B(\theta)$ measures the *label gradient alignment*:

$$A_B(\theta) = \frac{1}{B^2}\sum_{x\in B}\sum_{i\neq j}\langle\epsilon_x^i\nabla_\theta f_\theta^i(x), \epsilon_x^j\nabla_\theta f_\theta^j(x)\rangle. \tag{8}$$

where $\epsilon_x^i(x) = a(f_\theta(x))^i - y^i$ is the signed residual and a denotes the activation function of the last layer. Note that this residual term arises in our calculation using $\nabla_\theta L(x,y,\theta) = \epsilon_x^1(\theta)\nabla_\theta f_\theta^1(x) + \cdots + \epsilon_x^k(\theta)\nabla_\theta f_\theta^k(x)$ (see SM Section A.1, Eqn. (14)). This development also extends to other widely used loss functions such as the cross entropy and least-square loss.

Step 3: Now, from the modified loss in Eqn. (6), we can observe the conditions under which SGD puts an implicit pressure on the gradient norms $\|\nabla_\theta f_\theta^i(x)\|^2$ to be small: the batch gradient alignment and label gradient alignment terms must be small relative to the gradient norms, or positive valued.

We also derive conditions under which this implicit pressure on $\|\nabla_\theta f_\theta(x)\|_F^2$ transfers to a regularizing pressure on $\|\nabla_x f_\theta(x)\|_F^2$ (see SM Section A.3 for the proof):.

Theorem 5.1. *Consider a logit network $f_\theta : \mathbb{R}^d \to \mathbb{R}^k$ with l layers, then we have the following inequality*

$$\|\nabla_x f_\theta(x)\|_F^2 \leq \frac{\|\nabla_\theta f_\theta(x)\|_F^2}{T_1^2(x) + \cdots + T_l^2(x)}, \tag{9}$$

where T_i is the transfer function for layer i given by

$$T_i(x,\theta) = \frac{1}{\sqrt{\min(d,k)}}\frac{\sqrt{1+\|h_i(x)\|_2^2}}{\sigma_{\max}(W_i)\sigma_{\max}(h_i'(x))}, \tag{10}$$

where h_i is the subnetwork to layer i and $\sigma_{\max}(A)$ is the maximal singular value of matrix A (i.e., its spectral norm).

Here, we can see that for any settings where the sum $T_1^2(x) + \cdots + T_l^2(x)$ of squared transfer functions diminishes slower than the gradient norm $\|\nabla_\theta f_\theta(x)\|_F^2$ during training, we expect that implicit gradient regularization will apply a regularization pressure on GC.

An immediate prediction arising under these conditions is that the size of the regularization pressure on geometric complexity will depend on the implicit regularization rate h/B in Eqn. (6) (Note that the ratio h/B has been linked to implicit regularization strength in many instances [29, 71, 55, 47, 53]). Specifically, under these conditions, networks trained with larger learning rates or smaller batch sizes, or both, will apply a stronger regularization pressure on geometric complexity,

We test this prediction by performing experiments on ResNet18 trained on CIFAR10 and show results in Figure 4. The results show that while all models achieve a zero train loss, consistently the higher the learning rate, the higher the test accuracy and the lower the GC. Similarly, we observe that the lower the batch size, the lower the GC and the higher the test accuracy.

The increased performance of lower batch sizes has been long studied in deep learning, under the name 'the generalisation gap' [35, 41]. Crucially however, this gap was recently bridged [24], showing that full batch training can achieve the same test set performance as mini-batch gradient descent. To obtain these results, they use an explicit regularizer similar to the implicit regularizer in Eqn. (6), introduced to compensate for the diminished implicit regularization in the full batch case. Their results further strengthen our hypothesis that implicit regularization via GC results in improved generalisation.

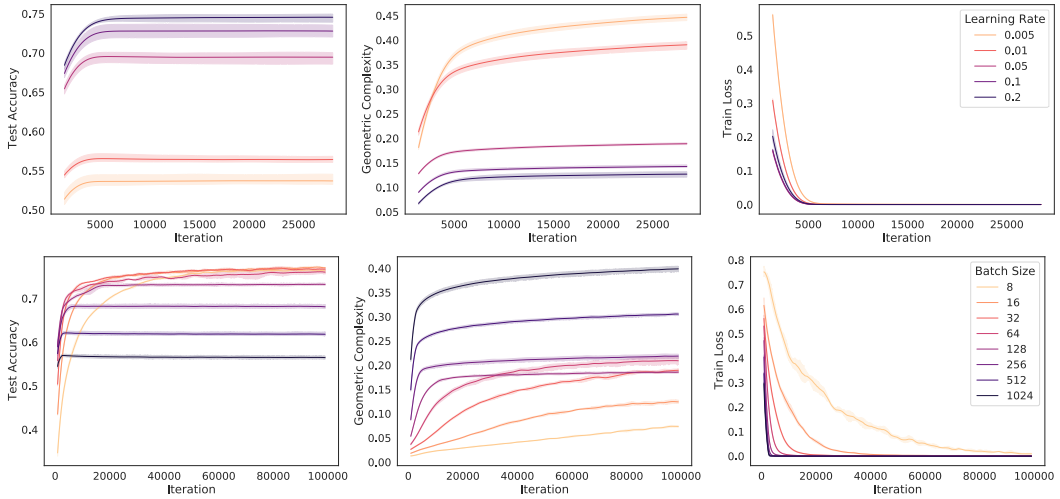

Figure 4: Impact of IGR when training ResNet18 on CIFAR10. **Top row:** As IGR increases through higher learning rates, GC decreases. **Bottom row:** Similarly, lower batch size leads to decreased GC.

6 Geometric complexity and double descent

When complexity is measured using a simple parameter count, a double descent phenomena has been consistently observed: as the number of parameters increases, the test accuracy decreases at first, before increasing again, followed by a second descent toward a low test error value [8, 9, 60]. An excellent overview of the double descent phenomena in deep learning can be found in [8], together with connections to smoothness as an inductive bias of deep networks and the role of optimisation.

To explore the double descent phenomena using GC we follow the set up introduced in [60]: we train multiple ResNet18 networks on CIFAR10 with increasing layer width, and show results in Fig. 5. We make two observations: first, like the test loss, GC follows a double descent curve as width increases; second, when plotting GC against the test loss, we observe a U-shape curve, recovering the traditional expected behaviour of complexity measures. Importantly, we observe the connection between the generalisation properties of overparametrized models and GC: after the critical region, increase in model size leads to a decrease in GC. This provides further evidence that GC is able to capture model capacity in a meaningful way, suggestive of a reconciliation of traditional complexity theory and deep learning.

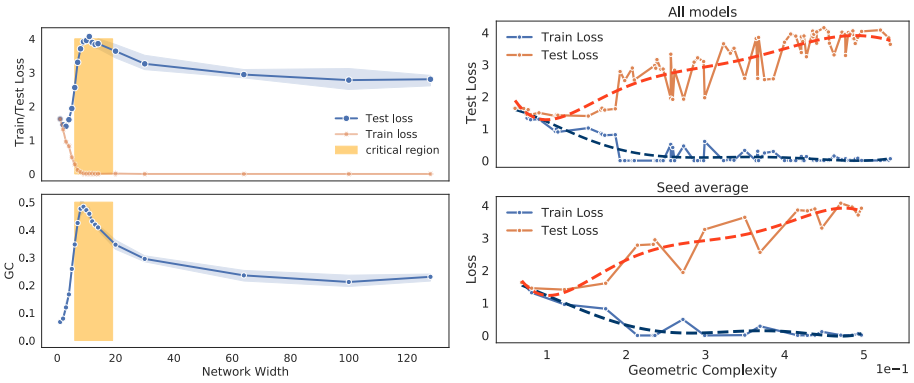

Figure 5: Double descent and GC. **Left:** GC captures the double descent phenomenon. **Right:** GC captures the traditional U-shape curve, albeit with some noise, showing (top) GC vs Test Loss for all models and (bottom) GC vs Test Loss averaged across different seeds. We fit a 6 degree polynomial to the curves to showcase the trend.

7 Related and Future Work

The aim of this work is to introduce a measure which captures the complexity of neural networks. There are many other approaches aiming at doing so, ranging from naive parameter count and more data driven approaches [37, 48, 54] to more traditional measures such as the VC dimension and Rademacher complexity [7, 38, 45, 75] which focus on the entire hypothesis space. As we saw in Fig. 1, the interpolating ReLU network with minimal GC is also the piecewise linear function with minimal volume or length [16]. In 1D this minimal function can be described only with the information given by the data points. This description with minimal information is reminiscent of the Kolmogorov complexity [67] and the minimum description length [34], and we believe that the exact relationship between these notions, GC, and minimal volume is worth investigating. Similarly, [1] argues that flat solutions have low information content, and for neural networks these flat regions are also the regions of low loss gradient and thus of low GC, as explained by the Transfer Theorem in Section 5. Another recent measure of complexity is the *effective dimension* which is defined in relation to the training data, but is computed using the spectral decomposition of the loss Hessian [54], making flat regions in the loss surface also regions of low complexity w.r.t. this measure. This hints toward the effective dimension being related to GC. Note that similarly to GC, the effective dimension can also capture the double descent phenomena. The effective dimension is an efficient mechanism for model selection [54], which our experiments seem also to indicate may be the case for GC.

GC is close to considerations about smoothness [64] and the Sobolev norm implicit regularization [53]. While GC and Lipchitz smoothness are connected, here we focused on a tractable quantity for neural networks and its connections with existing regularizers. Smoothness regularization has been particularly successful in the GAN literature [13, 30, 42, 57, 80], and the connection between Lipschitz smoothness and GC begs the question of whether their success is due to their implicit regularization of GC, but we leave the application of GC outside supervised learning for future work.

The Dirichlet energy in Eqn. (4) is a well-known quantity in harmonic function theory [20] and a fundamental concept throughout mathematics, physics and, more recently, 3D modeling [74], manifold learning [14], and image processing [25, 66]. Minimizers of the Dirichlet energy are harmonic and thus enjoy certain guarantees in regularity; i.e., any such solution is a smooth function. It has been demonstrated that for mean squared error regression on wide-shallow ReLU networks, gradient descent is biased towards smooth functions at interpolation [39]. Similarly, our work suggests that neural networks, through a mechanism of implicit regularization of GC, are biased towards minimal Dirichlet energy and thus encourage smooth interpolation. It may be interesting to understand how the relationship between GC, Dirichlet energy, and harmonic theory can help futher improve the learning process, in particular in transfer learning, or OOD generalization.

8 Discussion

In terms of *limitations*, the theoretical arguments presented in this work have a focus on ReLU activations and DNN architectures, with log-likelihood losses coming from the exponential family, such as multi-dimensional regression with least-square losses, or multi-class classification with cross-entropy loss. The experimental results are obtained on the image datasets MNIST and CIFAR using DNN and ResNet architectures.

In terms of *societal impact*, we are not introducing new training methods, but focus on providing a better understanding of the impact of common existing training methods. We hope that this understanding will ultimately lead to more efficient training techniques. While existing training methods may have their own risk, we do not foresee any potential negative societal impact of this work.

In conclusion, altogether, geometric complexity provides a useful lens for understanding deep learning and sheds light into why neural networks are able to achieve low test error with highly expressive models. We hope this work will encourage further research around this new connection, and help to better understand current best practices in model training as well as discover new ones.

References

[1] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–9, 2018.

[2] Michael Arbel, Danica J Sutherland, Mikołaj Bińkowski, and Arthur Gretton. On gradient regularizers for mmd gans. *Advances in neural information processing systems*, 31, 2018.

[3] Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018.

[4] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning*, pages 254–263. PMLR, 2018.

[5] David Balduzzi, Sebastien Racaniere, James Martens, Jakob Foerster, Karl Tuyls, and Thore Graepel. The mechanics of n-player differentiable games. In *International Conference on Machine Learning*, 2018.

[6] David G.T. Barrett and Benoit Dherin. Implicit gradient regularization. In *International Conference on Learning Representations*, 2021.

[7] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[8] Mikhail Belkin. Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation. 2021.

[9] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32), 2019.

[10] Johan Bjorck, Carla P Gomes, and Kilian Q Weinberger. Towards deeper deep reinforcement learning. *arXiv preprint arXiv:2106.01151*, 2021.

[11] Guy Blanc, Neha Gupta, Gregory Valiant, and Paul Valiant. Implicit regularization for deep neural networks driven by an ornstein-uhlenbeck like process. In *Conference on Learning Theory*, 2020.

[12] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[13] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2018.

[14] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[15] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

[16] Benoit Dherin, Michael Munn, and David G.T. Barrett. The geometric occam's razor implicit in deep learning. In *Proceedings on "Optimization for Machine Learning" at NeurIPS Workshops*, 2021.

[17] Gintare Karolina Dziugaite, Alexandre Drouin, Brady Neal, Nitarshan Rajkumar, Ethan Caballero, Linbo Wang, Ioannis Mitliagkas, and Daniel M Roy. In search of robust measures of generalization. *Advances in Neural Information Processing Systems*, 33:11723–11733, 2020.

[18] Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

[19] Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. *Advances in neural information processing systems*, 31, 2018.

[20] Lawrence C Evans. *Partial differential equations*, volume 19. American Mathematical Soc., 2010.

[21] William Fedus, Mihaela Rosca, Balaji Lakshminarayanan, Andrew M Dai, Shakir Mohamed, and Ian Goodfellow. Many paths to equilibrium: Gans do not need to decrease a divergence at every step. *arXiv preprint arXiv:1710.08446*, 2017.

[22] Tianxiang Gao and Vladimir Jojic. Degrees of freedom in deep neural networks. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, 2016.

[23] Jonas Geiping, Micah Goldblum, Phil Pope, Michael Moeller, and Tom Goldstein. Stochastic training is not necessary for generalization. In *International Conference on Learning Representations*, 2022.

[24] Jonas Geiping, Micah Goldblum, Phillip E Pope, Michael Moeller, and Tom Goldstein. Stochastic training is not necessary for generalization. *arXiv preprint arXiv:2109.14119*, 2021.

[25] Pascal Getreuer. Rudin-osher-fatemi total variation denoising using split bregman. *Image Processing On Line*, 2:74–95, 2012.

[26] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010.

[27] Florin Gogianu, Tudor Berariu, Mihaela C Rosca, Claudia Clopath, Lucian Busoniu, and Razvan Pascanu. Spectral normalisation for deep reinforcement learning: an optimisation perspective. In *International Conference on Machine Learning*, pages 3734–3744. PMLR, 2021.

[28] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016.

[29] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[30] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[31] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pages 1832–1841. PMLR, 2018.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[34] Geoffrey E Hinton and Richard Zemel. Autoencoders, minimum description length and helmholtz free energy. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6. Morgan-Kaufmann, 1994.

[35] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in neural information processing systems*, 30, 2017.

[36] Judy Hoffman, Daniel A. Roberts, and Sho Yaida. Robust learning with jacobian regularization. 2020.

[37] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*, 2018.

[38] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

[39] Hui Jin and Guido Montúfar. Implicit bias of gradient descent for mean squared error regression with wide neural networks. *arXiv preprint arXiv:2006.07356*, 2020.

[40] Jürgen Jost and Jèurgen Jost. *Riemannian geometry and geometric analysis*, volume 42005. Springer, 2008.

[41] 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. *arXiv preprint arXiv:1609.04836*, 2016.

[42] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[43] Naveen Kodali, James Hays, Jacob Abernethy, and Zsolt Kira. On convergence and stability of GANs. In *Advances in Neural Information Processing Systems*, 2018.

[44] V. Koltchinskii and D. Panchenko. Rademacher processes and bounding the risk of function learning. In *High Dimensional Probability II*, pages 443–459. Birkhäuser, 1999.

[45] Vladimir Koltchinskii and Dmitry Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50, 2002.

[46] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[47] Yoonho Lee, Juho Lee, Sung Ju Hwang, Eunho Yang, and Seungjin Choi. In *Advances in Neural Information Processing Systems*, 2020.

[48] Yoonho Lee, Juho Lee, Sung Ju Hwang, Eunho Yang, and Seungjin Choi. Neural complexity measures. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9713–9724. Curran Associates, Inc., 2020.

[49] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[50] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *Advances in Neural Information Processing Systems*, 31, 2018.

[51] Henry W Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.

[52] Chao Ma, Lei Wu, et al. The quenching-activation behavior of the gradient descent dynamics for two-layer neural network models. *arXiv preprint arXiv:2006.14450*, 2020.

[53] Chao Ma and Lexing Ying. The sobolev regularization effect of stochastic gradient descent. 2021.

[54] Wesley J Maddox, Gregory Benton, and Andrew Gordon Wilson. Rethinking parameter counting in deep models: Effective dimensionality revisited. *arXiv preprint arXiv:2003.02139*, 2020.

[55] Sam McCandlish, Jared Kaplan, Dario Amodei, and OD Team. An empirical model of large-batch training. arxiv 2018. *arXiv preprint arXiv:1812.06162*, 2018.

[56] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. In *Advances in Neural Information Processing Systems*, 2017.

[57] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[58] Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In *Advances in neural information processing systems*, 2017.

[59] Vaishnavh Nagarajan and J Zico Kolter. Generalization in deep networks: The role of distance from initialization. *arXiv preprint arXiv:1901.01672*, 2019.

[60] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

[61] Behnam Neyshabur. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*, 2017.

[62] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In *Advances in Neural Information Processing Systems*, 2019.

[63] Chongli Qin, Yan Wu, Jost Tobias Springenberg, Andy Brock, Jeff Donahue, Timothy Lillicrap, and Pushmeet Kohli. Training generative adversarial networks by solving ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[64] Mihaela Rosca, Theophane Weber, Arthur Gretton, and Shakir Mohamed. A case for new neural network smoothness constraints. In *Proceedings on "I Can't Believe It's Not Better!" at NeurIPS Workshops*, volume 137 of *Proceedings of Machine Learning Research*, 2020.

[65] Mihaela Rosca, Yan Wu, Benoit Dherin, and David G.T. Barrett. Discretization drift in two-player games. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, 2021.

[66] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[67] J. Schmidhuber. Discovering neural nets with low kolmogorov complexity and high generalization capability. *Neural networks : the official journal of the International Neural Network Society*, 10 5, 1997.

[68] Sihyeon Seong, Yegang Lee, Youngwook Kee, Dongyoon Han, and Junmo Kim. Towards flatter loss surface via nonmonotonic learning rate scheduling. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence*, 2018.

[69] Thiago Serra, Christian Tjandraatmadja, and Srikumar Ramalingam. Bounding and counting linear regions of deep neural networks, 2018.

[70] Samuel L Smith, Benoit Dherin, David G.T. Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. In *International Conference on Learning Representations*, 2021.

[71] Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*, 2018.

[72] Samuel L Smith and Quoc V. Le. A bayesian perspective on generalization and stochastic gradient descent. 2018.

[73] Jure Sokolic, Raja Giryes, Guillermo Sapiro, and Miguel R. D. Rodrigues. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65:4265–4280, 2017.

[74] Justin Solomon, Leonidas Guibas, and Adrian Butscher. Dirichlet energy for analysis and synthesis of soft maps. In *Computer Graphics Forum*, volume 32, pages 197–206. Wiley Online Library, 2013.

[75] Eduardo D Sontag et al. Vc dimension of neural networks. *NATO ASI Series F Computer and Systems Sciences*, 168:69–96, 1998.

[76] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, XVI(2):264–280, 1971.

[77] Dániel Varga, Adrián Csiszárik, and Zsolt Zombori. Gradient regularization improves accuracy of discriminative models. 2018.

[78] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *ArXiv*, abs/1705.10941, 2017.

[79] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[80] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019.

[81] Yaoyu Zhang, Zhi-Qin John Xu, Tao Luo, and Zheng Ma. A type of generalization error induced by initialization in deep neural networks. In *Mathematical and Scientific Machine Learning*, pages 144–164. PMLR, 2020.

[82] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Gradient descent optimizes over-parameterized deep relu networks. *Machine Learning*, 109(3):467–492, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The data is widely available (MNIST and CIFAR10). The code we use to train the model is standard training code using standard neural network architectures (MLP and ResNet18) available out-of-the-box in most ml libraries. We included detailed instructions with all the hyper-parameter used in the supplemental material to reproduce them.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We use the CIFAR10 and MNIST datasets for some of our experimental results. We cite both in the paper, for CIFAR following the guidance on Krizhevsky's homepage: https://www.cs.toronto.edu/ kriz/cifar.html.

(b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]