A Theoretical-Empirical Approach to Estimating Sample Complexity of DNNs

Devansh Bisla*, Apoorva Nandini Saridena*, Anna Choromanska
Department of Electrical and Computer Engineering,
Tandon School of Engineering, New York University
{db3484, ans609, ac5455}@nyu.edu

Abstract

This paper focuses on understanding how the generalization error scales with the amount of the training data for deep neural networks (DNNs). Existing techniques in statistical learning theory require a computation of capacity measures, such as VC dimension, to provably bound this error. It is however unclear how to extend these measures to DNNs and therefore the existing analyses are applicable to simple neural networks, which are not used in practice, e.g., linear or shallow (at most two-layer) ones or otherwise multi-layer perceptrons. Moreover many theoretical error bounds are not empirically verifiable. In this paper we derive estimates of the generalization error that hold for deep networks and do not rely on unattainable capacity measures. The enabling technique in our approach hinges on two major assumptions: i) the network achieves zero training error, ii) the probability of making an error on a test point is proportional to the distance between this point and its nearest training point in the feature space and at certain maximal distance (that we call radius) it saturates. Based on these assumptions we estimate the generalization error of DNNs. The obtained estimate scales as $\mathcal{O}\left(\frac{1}{\delta N^{1/d}}\right)$*, where* N *is the size of the training data, and is parameterized by two quantities, the effective dimensionality of the data as perceived by the network (*d*) and the aforementioned radius (*δ*), both of which we find empirically. We show that our estimates match with the experimentally-obtained behavior of the error on multiple learning tasks using benchmark data-sets and realistic models. Estimating training data requirements is essential for deployment of safety critical applications such as autonomous driving, medical diagnostics etc. Furthermore, collecting and annotating training data requires a huge amount of financial, computational and human resources. Our empirical estimates will help to efficiently allocate resources.*

1. Introduction

Deep learning (DL) establishes state-of-the-art performances in a number of learning tasks such as image recognition [49, 37], speech recognition [26, 13], and natural language processing [7, 20]. In recent years DL approaches were also shown to outperform humans in classic games such as Go [52]. The performance of DL models depends on three factors: the model's architecture, data set, and training infrastructure, which together constitute the artificial intelligence (AI) trinity framework [3]). It has been observed empirically [56] that increasing size of the training data is an extremely effective way for improving the performance of DL models, more so than modifying the network's architecture or training infrastructure. Generalization ability of a network is defined as difference between the training and test performance of the network. The relationship between the generalization ability of DL model and the size of the training data set constitutes a fundamental characteristic of the AI trinity framework, yet it is poorly described in the literature. Addressing this problem is crucial for safety critical applications where it is desired to understand the sample complexity of the model, or in other words, estimate the amount of data needed to achieve a certain acceptable level of performance. This paper aims at describing this relationship, in the context of a supervised learning setting, through a set of mixed mathematical-empirical tools.

Classic approaches in statistical learning theory for analyzing sample complexity rely on the measures of capacity of a classification model, such Vapnik-Chervonenkis (VC) dimension [59] or Rademacher complexity [9], which are potentially prohibitive to extend to practical DL models and may lead to loose estimates (for example, in the case where these measures approach infinity). The aim of this work is to develop a framework to model the sample complexity of DNNs that is free from such measures and straightforward to use by practitioners. We propose to model the probability of making an error by a DNN on a test sample as a function of the distance in the feature space between that sample (we denote the feature vector of the test sample as $\hat{x}$) and its nearest training sample (we denote the feature vector of this training sample as $x(\hat{x})$). This probability takes the following form:

$$\Phi(\hat{x}) := \min\left(1, \frac{||\hat{x} - x(\hat{x})||_2}{\delta}\right), \tag{1}$$

*Equal Contribution

where δ is a positive constant that we call the radius[1]. Thus, when test and training examples are further than δ in the feature space, the test example will be misclassified according to our model. The choice of the form of $\Phi(\hat{x})$ is also intuitive since as we increase the amount of training data, it becomes more reasonable to expect a test point within a certain distance from the training data point. Our approach is motivated by the fact that fundamentally DNNs can be interpreted as methods of non-linear dimensionality reduction which cluster together input data with similar features while push further apart dis-similar data points [2, 63]. Therefore, a test point far from the closest training point in the feature space is more likely to be incorrectly classified. In addition to assuming the aforementioned error mechanism for DNNs, we also assume that DNNs under consideration can learn perfectly and achieve zero error on the training data set. This assumption is in practice non-restrictive as it was observed empirically that DNNs of sufficient size can fit accurately labelled training data.[2] Finally, in order to estimate the generalization error in the function of the training data size (N), we define the notion of model-dependent effective dimensionality of the data d. It is a minimum dimensionality to which one can compress the feature vector without affecting the performance of the model. We find that this dimensionality is very low in practice. Under the above framework we estimate that the generalization error of a DNN follows a power law and scales inversely to $\delta N^{1/d}$. To the best of our knowledge, our work is distinct from existing work in the literature on the sample complexity of DNNs in a number of ways: i) it provides generalization error estimates, rather than mathematically-rigid error bounds ii) our result is free from complex capacity measures and relies on quantities (d and δ) which can be easily obtained empirically, iii) we perform exhaustive experimental evaluation of our theoretical result, and iv) our estimates can be easily adopted by practitioners to asses the amount of training data needed to meet the performance requirements of their applications.

The paper is organized as follows; Section 2 reviews the related work, Section 3 provides the mathematical derivations, Section 4 reports the experimental results, and finally Section 5 concludes the paper. The Supplement contains additional mathematical derivations and empirical evidence.

2. Background and Related Works

Statistical learning theory typically bounds the generalization error [10, 38, 32] using concentration inequalities, e.g., Hoeffding's inequality [31]. The error bounds depend on the measure of the complexity (capacity) of the hypothesis class that can be learned by a statistical classification algorithm. First existing bounds for simple learning algorithms, such as the histogram classifier, computed the complexity as the cardinality of the hypothesis class [12]. The corresponding bounds were inapplicable to problems involving an infinite class of functions, for which they became very loose. This led to the development of a new capacity measure - the VC dimension [58, 61, 59, 51, 17, 22], which is defined as the cardinality of the largest set of points that the algorithm can shatter and thus does not scale with the size of the hypothesis class. Resolving the VC bounds for neural networks [8, 5] leads to the theoretical guarantees that depend on the number of network parameters. Such bounds are not useful for practical networks.

The aforementioned error bounds are distribution-free and often loose in practice. This motivated work on distribution-dependent capacity measures such as VC entropy [60], covering numbers [1, 65], and Rademacher complexity [35, 9, 41]. Bounds based on covering numbers were derived for a limited family of classifiers, such as linear functional classes or neural networks with identity activation functions [65]. Rademacher complexity measures the ability of functions in the hypothesis space to fit to random labels. It has been recently observed that DNNs are powerful enough to fit any set of random labels [64] thus rendering the Rademacher complexity based bounds inadequate. Other capacity measures for neural networks, not mentioned before, include unit-wise capacities [45]. They led to generalization bounds for two layer ReLU networks. An excellent comparison of existing DNN generalization measures can be found in [33].

Estimating generalization bounds using PAC-Bayesian approaches and margin based analysis [43, 42, 39] is still an active area of research. More recently, several bounds based on PAC-Bayes approach have been presented for stochastic and compressed networks [21, 6, 66] that are computational in nature, by exploring modifications of the standard training procedure to obtain tighter(non-vacuous) generalization guarantees. However, these bounds are still loose ($>> 0$) to practically study the sample complexity of DNNs.

There also exist works that study the generalization phenomenon in DL from the perspective of the behavior of the optimization algorithm that minimizes the training loss. They are focused on the convergence properties of the optimizers and therefore are outside of the focus of this paper, with the exception of [46] that argues the existence of the "inductive bias" imposed by the optimizer, such as SGD, that restricts neural networks to a simple class of functions. This idea is linked with the notion of network's capacity though it is unclear how to use it to obtain sample complexity guarantees.

Above we discussed works that aim at proving theoretical bounds on the generalization error. Existing bounds

[1] Our empirical results were not sensitive to the choice of distance measure. We obtained similar results for squared distance.

[2] We assume that labeling of the training data is consistent and does not contain any mistakes.

that hold only for simplified DNNs typically scale with the training data size as $\mathcal{O}(1/\sqrt{N})$. An empirical family of approaches, that we will discuss next, instead studies the ways of extrapolating the learning curves, i.e. the dependence of the error on the amount of the training data, using parametric models. Among these works, we have linear, logarithmic, exponential, and power law parametric models [24] that were applied to decision trees. A subsequent paper [27] explored a vapor pressure model, the Morgan Mercer-Flodin (MMF) model, and the Weibull model to predict learning curves for classification algorithms such as decision tree and logistic discrimination. Empirical results obtained for a 2-layer neural network on MNIST data set showed that learning curve decays following the power law with a decay factor in the range $[1, 2]$ [16, 15]. This behavior of the learning curve was also observed in other applications [30]. These parametric modeling approaches are not supported by theoretical argument.

Finally, research works that are most closely related to our approach present asymptotic estimates of learning curves for Gaussian processes [54, 62], kernel methods [55], and wide neural networks trained in the regime of neural tangent kernels [14]. These works do not apply to a practical deep learning setting, but provide useful insights into the mathematical modeling of complex learning phenomena.

3. Generalization error estimation

In this section we derive the estimates for the generalization error of a DL model. Our analysis is performed under the assumptions that the model can learn the training data set with perfect accuracy and the probability of making an error on the test examples takes the form given in Equation 1.

3.1. Effective dimensionality

Practical DNNs are over-parameterized, i.e., the number of parameters far exceeds the number of training data samples. This over-parameterization induces redundancy in network weights [19, 44, 28], which particularly manifests itself on the output of the feature extractor of the network (the feature extractor typically precedes the fully connected layers of the model). The data representation there has to be simple enough so that the last layers of the network, which constitute shallow classifier, can perform accurate prediction. It has been noted in past works that this feature vector is low dimensional [47, 50, 48, 4]. We next describe how we define and find effective dimensionality of the feature space.

We introduce a bottleneck network consisting of two linear layers, each followed by the ReLU non-linearity, before the output layer of the network. The bottleneck takes D-dimensional feature vector as input, projects it down to dimensionality $d^{'}$, and then projects it back up to input dimension D. We insert the bottleneck into the trained model and fine-tune the entire network. The effective dimensionality d is the smallest value of $d^{'}$, for which the accuracy of the model with the bottleneck does not differ significantly from the accuracy of the original model without it. Empirical evaluation of d for different networks and data sets is presented in the experimental section.

The existence of small effective dimensionality of the feature space has been observed before in various works. Specifically, [47] defines effective dimensionality of the feature maps in terms of singular values of its co-variance matrix. They observe that as we move from input to the output layer of the network the effective dimensionality first increases and then drops. They report an effective dimensionality at the final layer of the network and show that it is as low as 2 for tiny ImageNet and CIFAR-10 data sets. Furthermore, they also observe a much sharper decline in effective dimensionality for large networks compared to the small ones. Similarly, [50] observed that < 10 singular values of the matrix of vectorized representations are enough to explain $> 99\%$ of the variance. They noted that enforcing even stronger low rank structure for the feature co-variance matrix can lead to better performance and robustness to adversarial examples. [48, 4] utilize an "ID estimator" previously introduced in [23] that relies on the ratio of distances to the nearest and second nearest neighbor of a data point to analyze intrinsic network dimensionality. These authors also observe that the neural network first increases and then decreases its intrinsic dimensionality to as low as 10 when moving towards network's output. Another work [25] reports similar behavior of the mutual information. The mutual information was found to be as low as < 4 nats closer to the final layer of the neural network. Finally, numerous network compression approaches implicitly rely on the existence of small effective dimensionality of the feature space when pruning network connections. They achieve $\approx 90\%$ [67] compression rate with negligible loss of the accuracy of the model.

We next move to our mathematical modeling of the generalization error. For the purpose of simplifying our analysis, we first consider the case where the effective dimensionality of the feature space is one and then extend the analysis to the general case of arbitrary dimensionality. Let f_{train} and f_{test} denote probability density functions of the train and test feature distributions. Then under the proposed error model defined in Equation 1 the overall probability of making an error on the test set is given by the expectation $\mathbb{E}_{f_{\text{test}}}[\Phi]$.

3.2. Generalization error estimates for one dimensional case (d = 1)

Let $\hat{x}$ be a given test point in the feature space whose immediate nearest training points in the feature space are x_i

and x_j, such that $\hat{x} \in (x_i, x_j)$ and let $\rho(\hat{x}) = |x_j - x_i|$. Intuitively, as we increase the number of training data points sampled from f_{train} the distance between two training samples i.e $\rho(x)$ decreases. Assume that f_{test} is close to a uniform distribution, denoted as u, in the interval (x_i, x_j). Note that this is a realistic assumption, i.e. at the tail of the distribution we observe training samples rarely but at the same time the training distribution there is flat whereas in high-concentration regions, where the training distribution changes quickly, the training samples are observed close to each other. Thus in the latter case the dynamics of the changes of the distribution are compensated by the small distance between samples. Also the more data we have, which is the regime we are mostly interested in analyzing, the more accurate this assumption is. Since the test point $\hat{x}$ is uniformly distributed in the interval (x_i, x_j), the distance from the test point to its closest training point (denoted as $\psi(\hat{x})$) is also uniformly distributed in the range $[0, \frac{\rho(\hat{x})}{2}]$. We can compute the expectation of $\psi(\hat{x})$ as (see Derivations for Equation 2 in the Supplement),

$$\mathbb{E}_u^{\langle x_i, x_j \rangle}[\psi(\hat{x})] = \frac{|x_j - x_i|}{4} = \frac{\rho(\hat{x})}{4}. \tag{2}$$

In the large data regime, we can approximate the distance between two training points ($\rho(\hat{x})$) as the limit of the ratio of length of the interval to number of points lying in the interval as,

$$\rho(\hat{x}) \approx \lim_{\Delta \to 0} \frac{\Delta}{\int_{\hat{x}-\frac{\Delta}{2}}^{\hat{x}+\frac{\Delta}{2}} N f_{\text{train}}(x) dx}$$
$$= \lim_{\Delta \to 0} \frac{\Delta}{N[F_{\text{train}}(\hat{x} + \frac{\Delta}{2}) - F_{\text{train}}(\hat{x} - \frac{\Delta}{2})]} = \frac{1}{N f_{\text{train}}(\hat{x})}$$

The above approximation does not include the local variance of $\rho(\hat{x})$. The effect of local variance results from the fact that neighboring training intervals should roughly have the same length but in practice they do not. Including that effect is crucial in the experiments. Thus we correct $\rho(\hat{x})$ by taking into account this local variance. We denote corrected $\rho(\hat{x})$ as $\rho'(\hat{x})$. The neighboring intervals should have same density function usually, so we calculate $\rho'(\hat{x})$ using K left and K right neighboring intervals of the training interval $\langle x_i, x_j \rangle$. We refer to the lengths of these intervals as $\rho_{-K}(\hat{x}), \rho_{-K+1}(\hat{x}), \ldots, \rho_K(\hat{x})$. Note that

$$\mathbb{E}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})] \coloneqq \frac{\sum_{i=-K}^{K} \rho_i(\hat{x})}{2K+1} = \rho(\hat{x})$$

and

$$\text{Var}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})] \coloneqq \frac{\sum_{i=-K}^{K} \rho_i^2(\hat{x})}{2K+1} - \left(\mathbb{E}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})]\right)^2$$

$$\rho'(\hat{x}) = \sum_{i=-K}^{K} \underbrace{\frac{\rho_i(\hat{x})}{\sum_{j=1}^{K} \rho_j(\hat{x})}}_{\text{prob. of falling into the interval}} \cdot \underbrace{\rho_i(\hat{x})}_{\text{interval length}}$$
$$= \frac{1}{\sum_{j=-K}^{K} \rho_j(x_t)} \sum_{i=-K}^{K} \rho_i(\hat{x})^2 dx = \frac{\sum_{i=-K}^{K} \rho_i(\hat{x})^2}{\sum_{j=-K}^{K} \rho_j(\hat{x})}$$
$$= \frac{\text{Var}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})]}{\mathbb{E}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})]} + \mathbb{E}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})]$$

For 1-dimension case, we empirically verified $\mathbb{E}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})] \approx \frac{Var^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})]}{\mathbb{E}^{\langle \rho_{-K}, \rho_K \rangle}[\rho(\hat{x})]}$, thus:

$$\rho'(\hat{x}) = 2\rho(\hat{x}) = \frac{2}{N f_{\text{train}}(\hat{x})} \tag{3}$$

Using Equations 1, 2, and 3 we can derive the probability of an error on the test set, $\mathbb{E}_{f_{\text{test}}}[\Phi]$, as follows:

$$\mathbb{E}_{f_{\text{test}}}[\Phi] = \int_{-\infty}^{+\infty} \Phi(\hat{x}) f_{\text{test}}(\hat{x}) d\hat{x}$$
$$= \int_{-\infty}^{+\infty} min\left(1, \frac{\psi(\hat{x})}{\delta}\right) f_{\text{test}}(\hat{x}) d\hat{x}$$
$$\approx \int_{-\infty}^{+\infty} min\left(1, \frac{\rho(\hat{x})}{4\delta}\right) f_{\text{test}}(\hat{x}) d\hat{x}$$
$$\approx \int_{-\infty}^{+\infty} min\left(1, \frac{1}{2N f_{\text{train}}(\hat{x})\delta}\right) f_{\text{test}}(\hat{x}) d\hat{x} \tag{4}$$

The integral in Equation 4 can be computed in the closed form for many standard distributions, such as Gaussian or uniform, else it can be computed using Monte Carlo method [11].

3.3. Generalization error estimates for multi-dimensional case

Now we consider multi-dimensional feature distributions. Let $\hat{x}$ be a test point in the feature space whose immediate 2^d nearest training points in the feature space form a set $\bar{X}$ and let $\mathcal{P}$ be a convex hull spanned by these training points. Assume $\mathcal{P}$ contains $\hat{x}$. For the ease of further derivations, we assume training points from $\bar{X}$, sampled from distribution f_{train}, lie on the vertices of a d-dimensional hyper-cube $\mathcal{P}$ with side length $a(\hat{x})$. The side length of the hyper-cube $\mathcal{P}$ depends on the position of the test point $\hat{x}$. This is because in places with higher density of training data points we can construct a tighter convex hull around the test point $\hat{x}$, and hence the length of the side of the hyper-cube $\mathcal{P}$ should decrease then. Furthermore, similar to 1-dimensional case let the test feature distribution be close to uniform, denoted as u, in $\mathcal{P}$.

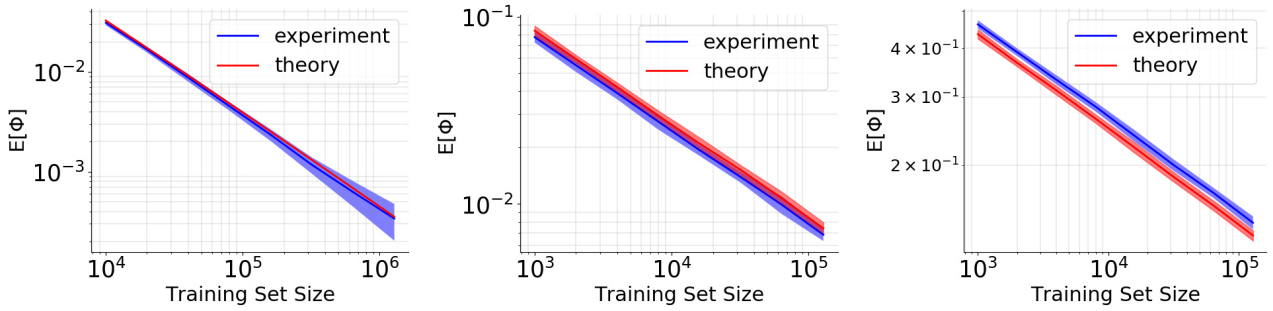

Figure 1. Monte Carlo simulation results (blue curve) confronted with theoretical derivations (red curve) for **(left)** d = 1 **(center)** d=2 and **(right)** d = 4. M = 1K, $f_{train} = f_{test} = \mathcal{N}_d(\mu = 0, \Sigma = I)$. The error bars capture 2 standard deviations.

In the large data regime, we approximate the distance of $\hat{x}$ to its closest training feature vector (denoted as $\psi(\hat{x})$) with the expected value of the distance of $\hat{x}$ to the closest training point in $\mathcal{P}$ (depending on the position in $\mathcal{P}$, the closest training point is one of the vertices of the hyper-cube $\mathcal{P}$). For ease of computation we assume $x(\hat{x})$ lies at the origin of the d-dimensional feature space hence, we can compute $\mathbb{E}_u^{\bar{X}}[\psi(\hat{x})]$ as,

$$\begin{aligned}\mathbb{E}_u^{\bar{X}}[\psi(\hat{x})] &= \int_{\mathcal{P}} \|\hat{x} - x(\hat{x})\|_2 u(\hat{x}) d\hat{x} \\ &= \int_0^{\frac{a(\hat{x})}{2}} \cdots \int_0^{\frac{a(\hat{x})}{2}} \|\hat{x}\|_2 \frac{1}{(\frac{a(\hat{x})}{2})^d} d\hat{x}_1 \dots d\hat{x}_d \\ &= \frac{1}{(\frac{a(\hat{x})}{2})^d} \int_0^{\frac{a(\hat{x})}{2}} \cdots \int_0^{\frac{a(\hat{x})}{2}} \sqrt{\sum_{i=1}^{d} \hat{x}_i^2} d\hat{x}_1 \dots d\hat{x}_d \end{aligned} \quad (5)$$

In the large data regime, we can approximate the distance between two training data points, or in other words the side of the hyper-cube $\mathcal{P}$, $a(\hat{x})$, as the limit of the ratio of the volume of the hyper-cube $\mathcal{P}$ to the number of points lying in $\mathcal{P}$:

$$\begin{aligned} a(\hat{x}) &\approx \left(\lim_{\text{Volume}(\mathcal{P}) \to 0} \frac{\text{Volume}(\mathcal{P})}{\int_{\mathcal{P}} N f_{\text{train}}(\hat{x}) dx} \right)^{1/d} \\ &= \left(\lim_{\Delta \to 0} \frac{1}{N f_{\text{train}}(\hat{x} + \Delta)} \right)^{1/d} \\ &= \frac{1}{(N f_{\text{train}}(\hat{x}))^{1/d}}. \end{aligned} \quad (6)$$

In higher dimensions (> 1), we empirically verified that no correction to $a(\hat{x})$ is required.

Similarly to 1-dimensional case, we can use Equations 1, 5, and 6 to derive the probability of an error on the test set, $\mathbb{E}_{f_{\text{test}}}[\Phi]$, as follows:

$$\mathbb{E}_{f_{\text{test}}}[\Phi] = \int_{-\infty}^{+\infty} \cdots \int_{-\infty}^{+\infty} \min\left(1, \frac{\psi(\hat{x})}{\delta}\right) f_{\text{test}}(\hat{x}) d\hat{x} \quad (7)$$

where,

$$\psi(\hat{x}) = \frac{1}{(\frac{a(\hat{x})}{2})^d} \int_0^{\frac{a(\hat{x})}{2}} \cdots \int_0^{\frac{a(\hat{x})}{2}} ||\hat{x}||_2 d\hat{x}_1 \dots d\hat{x}_d$$

and

$$a(\hat{x}) = \frac{1}{(N f_{train}(\hat{x}))^{\frac{1}{d}}}.$$

The obtained integral cannot be computed in the closed form, however it can be computed using Monte Carlo methods (note that d in our experiments is very small, i.e. it does not exceed 4, which enables accurate Monte Carlo approximations).

4. Experiments

We conduct two types of experiments. First, we verify our derivations for the generalization error estimator using Monte Carlo simulations. We use toy data sets generated from the Gaussian distribution. We then move to the main experiments, which are performed on the real data. These experiments involve classification and regression problems. The classification task is performed on the following data sets: MNIST [40], CIFAR [36] and ImageNet [18]). Our experiments utilize popular DNN architectures: LeNet [40], VGG16 [53], ResNet18, and ResNet50 [29]. We used cross entropy loss functions and stochastic gradient descent at training. The regression task is performed on the Udacity [57] data set, which is typically used in the autonomous driving applications. It contains images from left, center and right cameras that are mounted on the vehicle and additional vehicle logs such as speed, steering command etc. The data set is imbalanced and contains mostly samples corresponding to driving straight. We sub-sampled those to balance the data. The final balanced data set contains 38936 training examples, 6552 validation examples, and 8190 test examples. For the Udacity experiments we utilize a network described in Table 7 in the Supplement that takes single image as input and predicts the appropriate steering command. The network was trained using mean squared error loss and Adam optimizer [34].

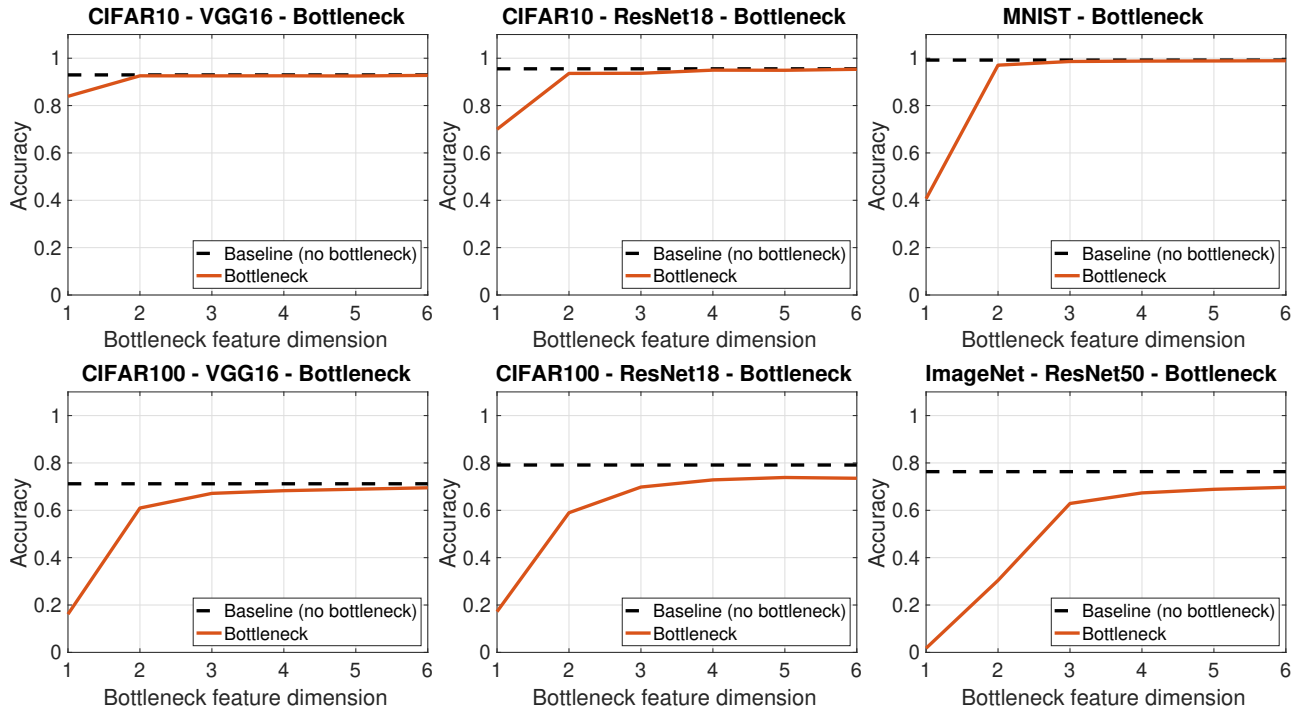

Figure 2. Test accuracy for different classification data sets with bottleneck network inserted before the output layer of the model. **Top Row:** CIFAR-10 trained on **(Left)** VGG16, **(Center)** ResNet-18 and **(Right)** MNIST trained on LeNet. **Bottom Row:** CIFAR-100 trained on **(Left)** VGG16 , **(Center)** ResNet-18 and **(Right)** ImageNet trained on ResNet-50.

4.1. Monte Carlo simulations

Here we assume that both train and test data points are d-dimensional vectors (we explored $d = 1, 2, 4$) that are drawn from a known Gaussian distribution $\mathcal{N}_d(\mu, \Sigma)$. We set $\delta = 1$. We generate the train and test sets containing respectively N and M data points. Since we are interested in this paper in examining how the error scales with N, our experiments are performed on training data sets with a growing size (N). In the simulation, for each test point we find the closest point in the training data set. We count the test point as a failure with the probability obtained using Equation 1. The error rate is computed as a number of failures divided by the size of the test data set (blue curve in Figure 1). We run the simulation for each value of N twenty time with different seeds. We confront the error rate obtained from simulation with the theoretical one obtained using Equations 4 and 7 (red curve in Figure 1). We use Monte Carlo method to compute the integrals in these equations. The results are captured in Figure 1. The experiment shows that simulated and theoretical curves match, which confirms the correctness of our theoretical derivations.

4.2. Real data experiments

4.2.1 Finding effective dimensionality

In Figure 2 and Table 4 in the Supplement we show the experiment capturing the selection of the effective dimensionality involving the injection of the bottleneck to the network

# filters in Conv1	# filters in Conv2		
	4	8	16
2	3	3	3
4	2	2	2
6	2	2	2

Table 1. d values for networks of varying capacity (i.e. varying number of filters in the first (Conv1) and second (Conv2) convolutional layer of the LeNet model.

Width	10	20	50	100	200	300
d	5	4	2	2	2	2

Table 2. d values for networks of varying capacity (MLP with single hidden layer and varying width).

(it was described in the Section 3) for MNIST, CIFAR10, CIFAR100, and ImageNet data sets. Effective dimensionality d is chosen as the size of the bottleneck for which we start observing saturation. We empirically found (see Figure 11) that this choice of d allows to accurately estimate the learning curve, even when the accuracies of bottleneck models do not reach the accuracies of the original models as is the case for CIFAR100 and ImageNet data sets. Note that the accuracy of the model with the bottleneck saturates at $d = 2$ for MNIST and CIFAR10, $d = 2/d = 3$ for CIFAR100 data set, and $d = 3/d = 4$ for ImageNet data set.

Furthermore, we also extracted feature vectors of different dimensions from the bottleneck model and performed nearest neighbor classification on the low-dimensional features. We found that the performance of the nearest neigh-

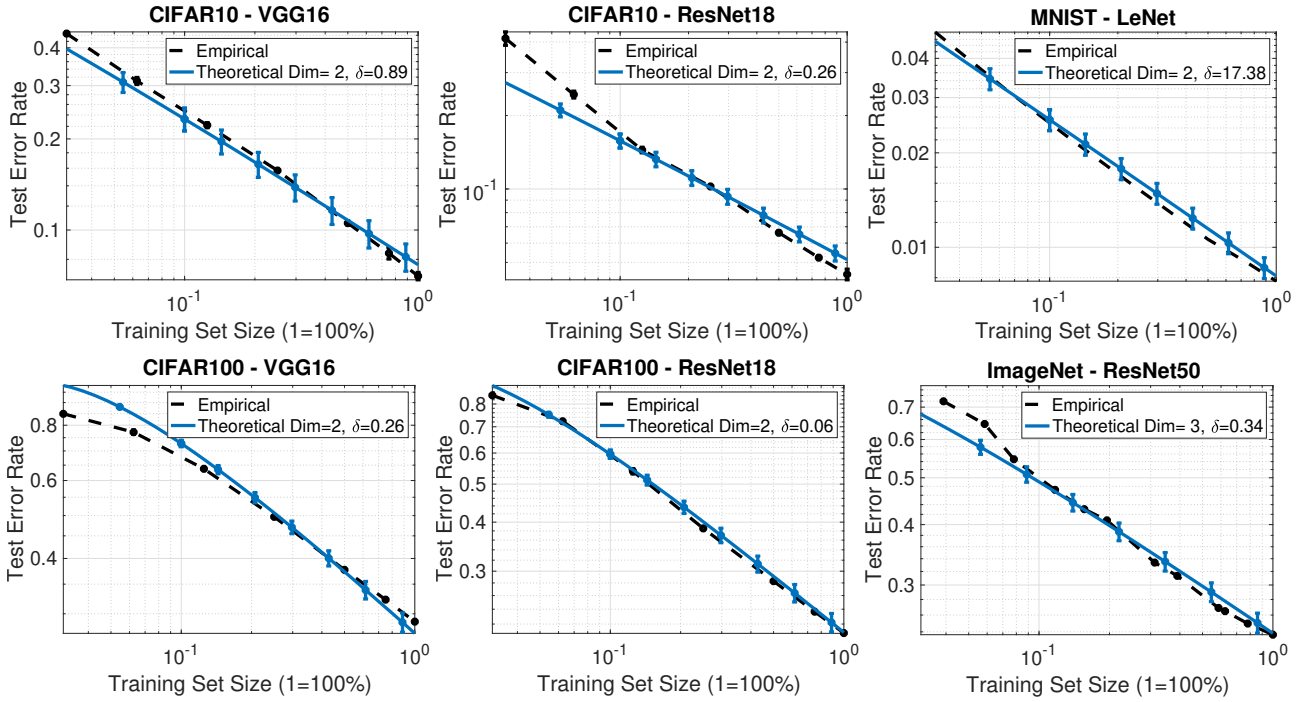

Figure 3. Theoretical and empirical learning curves for classification experiments **Top Row:** CIFAR-10 trained on (**left**) VGG16, (**center**) ResNet-18 and (**right**) MNIST trained on LeNet. **Bottom Row:** CIFAR-100 trained on **(left)**VGG16 and **(center)** ResNet-18, and **(right)** ImageNet trained on ResNet-50 .

bor saturates for the same values of d as described above. These results are highlighted in Figure 5 and Table 5 in the Supplement. This ensures us that the dimensionality found using the bottleneck indeed captures enough variety in the data to perform accurate prediction. Finally, for the Udacity, the effective dimensionality we found was equal to 1. Apart from training data set, the effective dimensionality of the feature space indirectly depends on the capacity of the neural network which in turn depends on network design. We verify this claim by training multiple LeNet and MLP models with varying capacity on MNIST data set and computing the effective dimensionality for each of the model. The LeNet model consists of two convolution layers with 6 and 16 filters respectively. We control the capacity of the network by decreasing the number of filters in each convolutional layer. For MLP, we use single hidden layer and vary its width. As the capacity of the network decreases we observe an increase in the effective dimensionality. The results are highlighted in Table 1 and 2.

4.2.2 Learning curves

The empirical learning curves were obtained by testing DNNs trained on increasingly larger data sets. Thus we sampled MNIST, CIFAR and Udacity data set to obtain training data sets of size equal to 3.125%, 6.25%, 12.5%, 25%, 50%, 75% and 100% of the entire data set. For ImageNet we obtained training data sets of size equal to

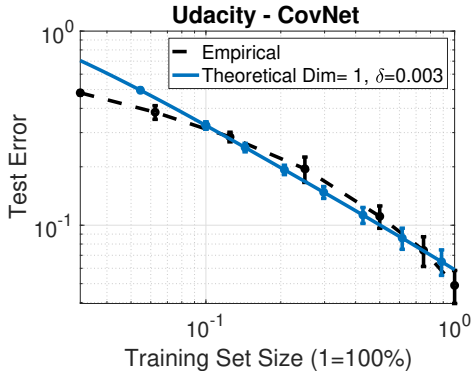

Figure 4. Theoretical and empirical learning curves for Udacity data set.

3.90%, 5.85%, 7.80%, 11.70%, 15.61%, 19.51%, 31.22%, 39.02%, 58.54%, 62.44%, 78.05% and 100% of the entire data set. In the obtained training data sets, all classes are equally well-represented (they are balanced). For classification problems, we plot the experimental learning curve by computing the test error rate, i.e number of samples incorrectly classified by the model (see Figure 3 and Table 3; theoretical curves for various settings of effective dimensionality are reported in Figure 11 in the Supplement). For regression task, we count the test sample as misclassified if the predicted steering command deviates from the label by more than 0.1 (in the Udacity the steering command is typically in the range (-0.5, 0.5); see Figure 4 for the results).

The theoretical learning curves were obtained according

Data set	Model	f_{train}		f_{test}		d	δ
		μ	diag(Σ)	μ	diag(Σ)		
MNIST	LeNet	$\begin{bmatrix}0.020\\0.012\end{bmatrix}$	$\begin{bmatrix}377.855\\264.029\end{bmatrix}$	$\begin{bmatrix}-0.061\\-0.036\end{bmatrix}$	$\begin{bmatrix}384.357\\270.399\end{bmatrix}$	2	17.375
CIFAR-10	VGG-16	$\begin{bmatrix}-0.004\\0.024\end{bmatrix}$	$\begin{bmatrix}98.192\\80.676\end{bmatrix}$	$\begin{bmatrix}0.009\\-0.060\end{bmatrix}$	$\begin{bmatrix}87.691\\76.163\end{bmatrix}$	2	0.890
	ResNet-18	$\begin{bmatrix}-0.003\\-0.001\end{bmatrix}$	$\begin{bmatrix}3.717\\2.789\end{bmatrix}$	$\begin{bmatrix}0.007\\0.003\end{bmatrix}$	$\begin{bmatrix}3.726\\2.662\end{bmatrix}$	2	0.262
CIFAR-100	VGG-16	$\begin{bmatrix}0.023\\0.0464\end{bmatrix}$	$\begin{bmatrix}123.572\\121.108\end{bmatrix}$	$\begin{bmatrix}-0.0584\\-0.116\end{bmatrix}$	$\begin{bmatrix}102.838\\97.932\end{bmatrix}$	2	0.265
	ResNet-18	$\begin{bmatrix}-0.001\\-0.010\end{bmatrix}$	$\begin{bmatrix}3.648\\3.449\end{bmatrix}$	$\begin{bmatrix}0.003\\0.0260\end{bmatrix}$	$\begin{bmatrix}2.914\\2.811\end{bmatrix}$	2	0.056
ImageNet	ResNet-50	$\begin{bmatrix}-0.002\\-0.007\\-0.020\end{bmatrix}$	$\begin{bmatrix}19.940\\14.308\\12.331\end{bmatrix}$	$\begin{bmatrix}-0.002\\-0.007\\-0.020\end{bmatrix}$	$\begin{bmatrix}19.940\\14.308\\12.331\end{bmatrix}$	3	0.340
Udacity	CovNet	-0.0111	4.3415	0.0265	6.1000	1	0.003

Table 3. The effective dimensionality d and δ parameter for different data sets and model architectures. The train and test feature distributions are denoted as f_{train} and f_{test}. μ denotes the mean of the distribution and $diag(\Sigma)$ denotes the diagonal elements of the co-variance matrix (off-diagonal elements are equal to 0).

to Equations 4 and 7, where the integral were computed using Monte Carlo method. Note that our generalization error estimate is dependent on the feature train and test distributions. In order to obtain the features for the distribution estimation, we train the DNN on a subset of the training data, i.e. 50%. Next, we process this subset as well as the subset of the test data with this DNN. The obtained features are then projected via PCA to the effective dimensionality d. We assume single d-dimensional Gaussian distribution for both the train and test data, whose parameters (mean and covariance) we estimate via maximum likelihood approach (it has been previously observed that features space learned by DNNs exhibit a simple clustering structure [25]). Finally, we treat δ as a hyperparameter of the error estimate. It was obtained under small data regime ($\leq 50\%$ of training data) by minimizing the distance between theoretical and empirical curve. Therefore, after training network on small amount of data, which is computationally much faster than training on the entire corpus, we estimate δ and predict the behavior of the learning curve in large data regime. Table 3 and 6 in the Supplement summarizes the choice of hyperparameters for different data sets and architectures. As can be seen in Table 3, δ heavily depends on the considered combination of data set and architecture (difference is often in order of magnitudes).

Figure 3, Figure 4 and Table 3 report the results confronting the theoretical and empirical learning curves. Note that among all our data sets, only ImageNet does not satisfy the assumption of zero training error (see Figures 9, 6, 7, 8; for Udacity data set the training error is close to zero as can be seen in Figure 10), nevertheless even for this data set we could well-model the behavior of the learning curve using our theoretical framework. According to [30] the learning curve can be broken down into three regions: low data region, power law region, and the saturation region. In our experiments we observe first two regions. In low data regime we observe over-fitting. In this case we observe a mismatch between the theoretical and empirical curves (recall that our estimates of the generalization error become more accurate with increasing N). In the power law region, as we increase the amount of training data the performance of the network consistently improves. Our theoretical framework estimates the empirical learning curve in this region very well.

5. Conclusion

In this paper we address the problem of describing the behavior of the generalization error of DL models with the growing size of the training data. We attempt to reconcile the dichotomy between existing theoretical approaches, which rely on capacity measures that are potentially impossible to obtain for practical DNNs, and existing empirical approaches that model the behavior of the error by fitting it to a parameterized curve and lack any theoretical description. Our error estimates stem from a simple model of a DL machine that we propose and analyze. Our approach relies on modeling assumptions, which are however not unrealistic and gives rise to the estimates of the generalization error curves that closely resemble the ones empirically observed. We verify our approach on several learning tasks involving various realistic architectures and data sets.

References

[1] Noga Alon, Shai Ben-David, Nicolo Cesa-Bianchi, and David Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the ACM (JACM)*, 44(4):615–631, 1997. 2

[2] Rana Ali Amjad and Bernhard Claus Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 2

[3] Anima Anandkumar. The AI Trinity: Data + Algorithms + Infrastructure. https://www.youtube.com/watch?v=Mzior-Jmp8A, 2018. 1

[4] Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. Intrinsic dimension of data representations in deep neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 6109–6119. Curran Associates, Inc., 2019. 3

[5] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999. 2

[6] Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *35th International Conference on Machine Learning, ICML 2018*, pages 390–418. International Machine Learning Society (IMLS), 2018. 2

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[8] Peter L. Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019. 2

[9] 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. 1, 2

[10] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Summer School on Machine Learning*, pages 169–207. Springer, 2003. 2

[11] Russel E. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta Numerica*, 7:1–49, 1998. 4

[12] Rui Castro. Statistical learning theory. In *Lecture Notes*, April 2018. 2

[13] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE, 2018. 1

[14] Omry Cohen, Or Malka, and Zohar Ringel. Learning curves for deep neural networks: a gaussian field theory perspective. *arXiv preprint arXiv:1906.05301*, 2019. 3

[15] Corinna Cortes, Lawrence D Jackel, and Wan-Ping Chiang. Limits on learning machine accuracy imposed by data quality. In *Advances in Neural Information Processing Systems*, pages 239–246, 1995. 3

[16] Corinna Cortes, Lawrence D Jackel, Sara A Solla, Vladimir Vapnik, and John S Denker. Learning curves: Asymptotic values and rate of convergence. In *Advances in Neural Information Processing Systems*, pages 327–334, 1994. 3

[17] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995. 2

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009. 5

[19] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in neural information processing systems*, pages 1269–1277, 2014. 3

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. 1

[21] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence, UAI 2017, Sydney, Australia, August 11-15, 2017*, 2017. 2

[22] Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247 – 261, 1989. 2

[23] Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific reports*, 7(1):12140, 2017. 3

[24] Lewis J. Frey and Douglas H. Fisher. Modeling decision tree performance with the power law. In *Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics, AISTATS 1999, Fort Lauderdale, Florida, USA, January 3-6, 1999*, 1999. 3

[25] Ziv Goldfeld, Ewout Van Den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating information flow in deep neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2299–2308, Long Beach, California, USA, 09–15 Jun 2019. PMLR. 3, 8

[26] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013. 1

[27] Baohua Gu, Feifang Hu, and Huan Liu. Modelling classification performance for large data sets. In *International Conference on Web-Age Information Management*, pages 317–328. Springer, 2001. 3

[28] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015. 3

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[30] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory F. Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *CoRR*, abs/1712.00409, 2017. 3, 8

[31] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*, pages 409–426. Springer, 1994. 2

[32] Yiding Jiang, Behnam Neyshabur, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. 2

[33] Yiding Jiang, Behnam Neyshabur, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020. 2

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 5

[35] Vladimir Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001. 2

[36] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 5

[37] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. 1

[38] Sanjeev R. Kulkarni and Gilbert Harman. Statistical learning theory: a tutorial. *WIREs Computational Statistics*, 3(6):543–556, 2011. 2

[39] John Langford and John Shawe-Taylor. Pac-bayes & margins. In *Advances in neural information processing systems*, pages 439–446, 2003. 2

[40] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. 5

[41] Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with lipschitz functions. *Journal of Machine Learning Research*, 5(Jun):669–695, 2004. 2

[42] David McAllester. Simplified pac-bayesian margin bounds. In *Learning theory and Kernel machines*, pages 203–215. Springer, 2003. 2

[43] David A McAllester. Pac-bayesian model averaging. In *COLT*, volume 99, pages 164–170. Citeseer, 1999. 2

[44] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 3

[45] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. The role of over-parametrization in generalization of neural networks. In *International Conference on Learning Representations*, 2019. 2

[46] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015. 2

[47] Kavya Ravichandran, Ajay Jain, and Alexander Rakhlin. Using effective dimension to analyze feature transformations in deep neural networks. 2019. 3

[48] Stefano Recanatesi, Matthew Farrell, Madhu Advani, Timothy Moore, Guillaume Lajoie, and Eric Shea-Brown. Dimensionality compression and expansion in deep neural networks. *CoRR*, abs/1906.00443, 2019. 3

[49] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 1

[50] Amartya Sanyal, Varun Kanade, and Philip H. S. Torr. Low rank structure of learned representations. *CoRR*, abs/1804.07090, 2018. 3

[51] Norbert Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145–147, 1972. 2

[52] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, Jan. 2016. 1

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 5

[54] Peter Sollich. Gaussian process regression with mismatched models. In *Advances in Neural Information Processing Systems*, pages 519–526, 2002. 3

[55] Stefano Spigler, Mario Geiger, and Matthieu Wyart. Asymptotic learning curves of kernel methods: empirical data vs teacher-student paradigm. *arXiv preprint arXiv:1905.10843*, 2019. 3

[56] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017. 1

[57] Udacity. Udacity self-driving car driving data 10/3/2016 (dataset-2-2.bag.tar.gz). 5

[58] Vladimir Vapnik. *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag, Berlin, Heidelberg, 1982. 2

[59] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013. 1, 2

[60] Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998. 2

[61] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pages 11–30. Springer, 2015. 2

[62] Christopher KI Williams and Francesco Vivarelli. Upper and lower bounds on the learning curve for gaussian processes. *Machine Learning*, 40(1):77–102, 2000. 3

[63] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[64] 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. 2

[65] Tong Zhang. Covering number bounds of certain regularized linear function classes. *Journal of Machine Learning Research*, 2(Mar):527–550, 2002. 2

[66] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: a PAC-bayesian compression approach. In *International Conference on Learning Representations*, 2019. 2

[67] Michael Zhu and Suyog Gupta. To prune, or not to prune: Exploring the efficacy of pruning for model compression. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018. 3