Deep Double Descent: Where Bigger Models and More Data Hurts

Vincent Roduit
EPFL
Electrical Engineering
Sciper: 325140

Fabio Palmisano
EPFL
Electrical Engineering
Sciper: 296708

***Abstract*—Modern machine learning research shows that the classical bias-variance trade-off has its limitations as model size increases. Contrary to the classical machine learning perspective, increasing the complexity of a model does not always lead to overfitting. The purpose of this project is to replicate a research study conducted by engineers from Harvard and OpenAI [3]. This paper unites both sides (classical and modern) by showcasing that there is no contradiction between the two points of view."**

***Keywords*—** Deep Learning, Double Descent, Effective Model Complexity,Bias-Variance Trade-off

I. Introduction

The bias-variance trade-off classically describes phenomena where simpler models suffer from underfitting and complex models suffer from overfitting. Between these two extremes, there exists a point that minimizes the true error. In classical machine learning, this optimum defines the model complexity to choose in order to obtain the best results. Conversely, modern machine learning shows that models initially follow the same trends, but at a specific time, performances increase again along with model complexity. The aforementioned research paper describes this phenomenon as a double descent. Moreover, it is shown that modern deep learning models have two regimes, respectively the classical and modern regime. These two regimes resolve contradictions between classical and modern machine learning. The purpose of this project is to understand the work done in this paper and to reproduce the main figures. In this report, the results of this project are presented, as well as limitations and problems encountered during the development.

II. Theoretical Elements

Some theoretical aspects need to be explained to understand the main interest of this project. The first element has already been pointed out in Section I, and it is the bias-variance trade-off. More formally, the bias-variance trade-off represents a decomposition of the expectation of the true risk. This true risk can be decomposed into three terms, and the resulting equality is [2]:

$$\mathbb{E}_{S\sim\mathbb{D},\epsilon\sim\mathbb{D}_\epsilon}[(f(x_0)+\epsilon-f_S(x_0))^2]$$
$$=\mathbb{V}_{\epsilon\sim\mathbb{D}_\epsilon}[\epsilon] \quad (1)$$
$$+(f(x_0)-\mathbb{E}_{S'\sim\mathbb{D}})^2 \quad (2)$$
$$+\mathbb{E}_{S\sim\mathbb{D}}[(f_S(x_0)-\mathbb{E}_{S'\sim\mathbb{D}}[f_{S'}(x_0)])^2] \quad (3)$$

where (1) is the variance of the noise, (2) is the bias term, and (3) is the variance. From this equation, classical machine learning states that these three terms vary according to the curve presented in Figure 1. It is worth noticing that there is a strict lower bound to this error, which is the noise variance. Even modern machine learning models cannot go beyond this limit. The purpose of this project is to show that bigger models do not necessarily suffer from overfitting in high-degree models. The second point that needs to be discussed is a new definition proposed in [3]. This notion is called *Effective Model Complexity* (EMC) and generalizes the concept of double descent. This quantity not only depends on the number of parameters but also on the data distribution and the training procedure. This *Effective Model Complexity* can be described more formally as follows:

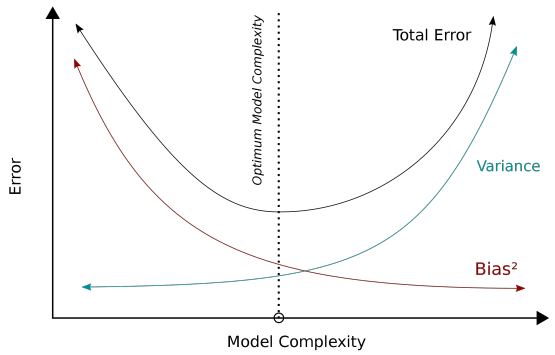


Fig. 1: Typical bias-variance curve

$$\text{EMC}_{\mathbb{D},\epsilon}(\tau) := \max\{n|\mathbb{E}_{S\sim\mathbb{D}^n}[Error_S(\tau))] \leq \epsilon\} \quad (4)$$

where:

- τ represents the training procedure.
- $\mathbb{D}$ is the distribution of the sample.
- $Error_S(M)$ describes the mean error of model M on samples S.

From this equation, three different states can be drawn. The first regime is the *under-parameterized* regime. It occurs when $EMC_{\mathbb{D},\epsilon}$ is smaller than n. In this case, a small perturbation of τ that increases the complexity leads to a decrease in the test error. The *over-parameterized regime* occurs when $EMC_{\mathbb{D},\epsilon}$ is larger than n. Conversely to the first one, a perturbation that increases the model complexity leads to an increase in the error. The last regime, called *critically parameterized regime*, happens when $EMC_{\mathbb{D},\epsilon}$ is almost equal to n. In this last regime, an increase in complexity could lead to both an increase or decrease in the test error.

III. Experiment Setup

In order to empirically verify the assumptions stated in Section II, several models on various datasets have to be tested. In the original papers, the models were a Residual Network, a Convolutional Neural Network, and transformers. These models were tested on two widely spread datasets, CIFAR10 and CIFAR100 [1]. For this project, only two models are tested (the ResNet and the Convolutional Neural

[1] https://www.cs.toronto.edu/~kriz/cifar.html

Network). For reasons stated in further sections, the datasets used are CIFAR10, CIFAR100, and MNIST[2].

A. Models

1) Convolutional Neural Network: Convolutional Neural Network (CNN) has shown impressive results when dealing with images. Unlike traditional neural networks, CNN has the ability to spatially represent images [4]. For this project, a basic CNN with 4 layers is used. Four convolutional layers are used, with the width being respectively [k, 2k, 4k, 8k], for k varying from 0 to 64.

2) ResNet: Residual Network (ResNet) is a specific type of Convolutional Neural Network that helps train models with a high number of coefficients. ResNet adds a Skip connection between the input and the output [1]. An example of such a structure is presented in Figure 2. The ResNet model used in this project is a variant of the classical ResNet18, with an adjustable width parameter k. This model has four layers, and the width of each layer is given by [k, 2k, 4k, 8k], k being the adjustable parameter [3].

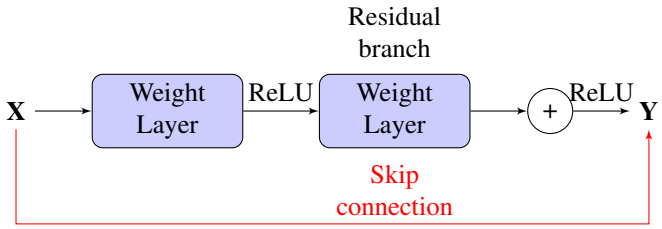


Fig. 2: Example of a Residual block

B. Optimizer, Learning Rate, Loss

The loss used for all experiments is the cross-entropy loss, a widely spread loss for deep learning with multiple classes [5]. Two optimizers are used. The first one is Stochastic Gradient Descent (SGD). It is used with a learning rate $\propto \frac{1}{\sqrt{T}}$ for 500K epochs. The second optimizer is Adam, a variant of SGD, involving a momentum term and second-order statistics. Adam is used with a learning rate of 0.0001 for 4K epochs.

C. Data Augmentation

Several data augmentation processes are used in this process. The first one is RandomHorizontalFlip [4], which flips the image horizontally, with a probability of 0.5 by default. The second one is RandomCrop(32, padding=4) [5], which crops the original image into a new image of size 32. The padding parameter adds 4 pixels on each border.

Fig. 3: Data augmentations

The last data augmentation used is the label noise. With a probability p, the label of an image is replaced with another, creating some noise in the data. Therefore, images have the correct label with probability $(1-p)$ and the wrong label with probability p. Several noise levels are used in this project (from 0% to 20%).

IV. Experiments

This section presents the experiments done in this project. There are two main experiments.

A. Double Descent

The first one is to reproduce the double descent, using ResNet with Adam optimizer on 4K epochs for a width varying from 1 to 64. Different noise ratio are considered (from 0 to 20%). Two datasets are used (CIFAR10 and CIFAR100 and MNIST). The Results sections will explain why MNIST was taken into consideration for this experiment.

B. Adam vs SGD

The second experiments proposes to show differences between Adam and SGD on CNN. Raw CIFAR10 (i.e. no label noise and no data augmentation) is used for this experiment. Adam is trained on 4K epochs, while SGD on 500K.

V. Results

This section presents the results obtained from the two experiments and discusses the challenges encountered during the project. One notable challenge is the significant time required for image generation. Both experiments involve a large number of epochs. The first experiment necessitates $n_{epochs,exp1} \times n_k \times n_{noise} = 768 \times 10^3$ epochs,where $n_k = 64$[6] and $n_{nosie} = 3$[7] while the second experiment requires $(n_{epochs,Adam} + n_{epochs,SGD}) \times n_k \approx 32 \times 10^6$ epochs. Table I provides a summary of the required time using the fastest available running environment, namely the V100 from Google.

TABLE I: Required time

Dataset	Time per epoch [s]	Exp. 1 [h]	Exp. 2 [h]
CIFAR10	13	$2.8 * 10^3$	$1.1 * 10^5$
CIFAR100	13	$2.8 * 10^3$	$1.1 * 10^5$
MNIST	10	$2.1 * 10^3$	$90 * 10^3$

Analysis of Table I reveals that conducting the experiments with the proposed parameters is impractical. Even when using a simpler dataset like MNIST, the required computation time remains prohibitively large. Solutions must be explored to overcome these challenges. The initial approach involved reducing the number of epochs and opting for the MNIST Dataset. Despite attempts to maximize the number of epochs, computational constraints restrict us to running the model for a maximum of 20 epochs under original conditions, which corresponds to 10 hours of computations. However, this approach introduces a significant problem: reducing the number of epochs to only 20 results in a suboptimal solution, as illustrated in Figure 4.

Furthermore, the Figure 7 (Figure 4 in the referenced paper) suggests that observing a double descent is improbable due to the relatively low number of epochs.

Plotting results for an non optimal problem makes no sense. Other solutions have to be found. These solutions involve reducing the size of the dataset to decrease the computational time per epochs, but also to improve the convergence.

Two improvements are done to improve the trade-off between fast convergence and small number of epochs. An Early Stopping[8] parameter is added to prevent unnecessary loss step. Finally, a scheduler

[2] http://yann.lecun.com/exdb/mnist

[3] if k=64, the model is the standard ResNet

[4] PyTorch documentation

[5] PyTorch documentation

[6] the number of width to test

[7] resp. 0,0.1 and 0.2

[8] Pytorch webstite

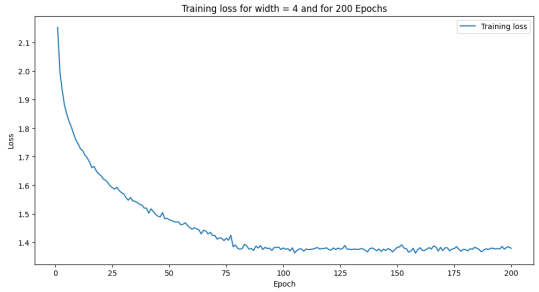
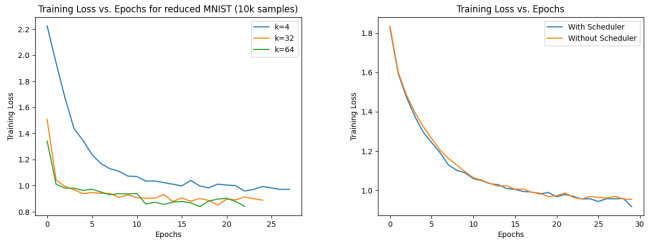


Fig. 4: Convergence of the loss with 200 Epochs for width = 4

is added to improve performance. Differences is shown in Figure 5b. For the same number of epochs on CIFAR10 with 20% label noise, adding a scheduler improves the accuracy of 5% [9]



(a) Early Stopping (b) Scheduler

Fig. 5: Effects of the improvement parameters

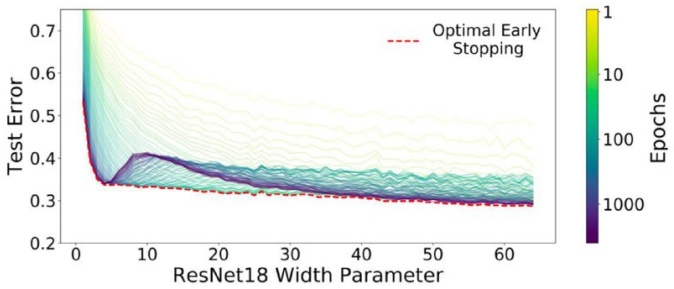
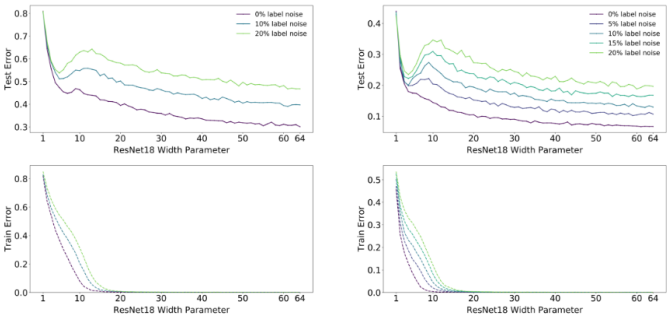


Fig. 6: Double descent phenomena according to number of epochs

A. *Experiment 1 (Fig. 4)*

The conditions of the first experiment become far from the original. The differences are summarize in Table II. Therefore, expecting the results to be similar to the one found in Figure 7 is unrealistic. Figure 8 presents the obtained results. From these figures it can be seen that increasing the width of the model does not implies overfitting. Furthermore, by looking at Figure 8b, it can be seen that even the train set experiences a plateau in the error for label noise greater than zero, while in the original figure all the errors tend to zero. It can be explained by two main factors, the number of epochs and the reduced size of the dataset. Regarding the shape of the test error in Figure 8a, it can be seen that it follows the general trend of the original, while the double descent is missing for reasons already stated.

[9]See notebook for more details (Github link)



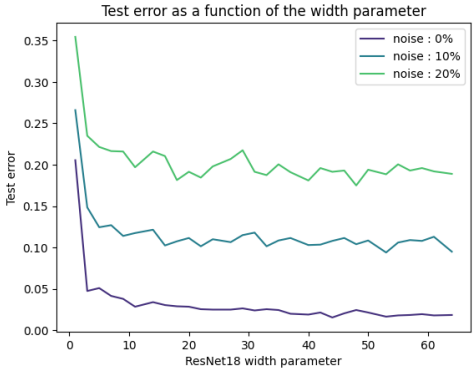
(a) **CIFAR-100.** There is a peak in test error even with no label noise.

(b) **CIFAR-10.** There is a "plateau" in test error around the interpolation point with no label noise, which develops into a peak for added label noise.

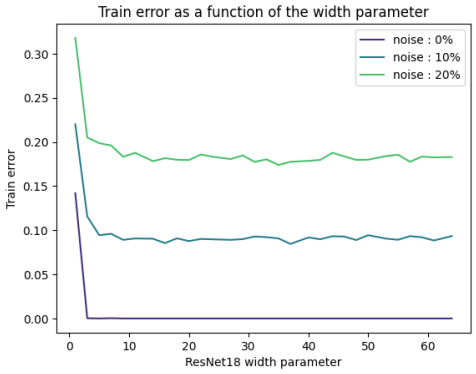
Fig. 7: Double descent phenomena according to number of epochs

TABLE II: Differences of parameter setup

Parameter	Original	Reproduced
Samples	50k	10k
Epochs	4k	[0;100]
Scheduler	×	✓
Early Stopping	×	✓



(a) Test Error



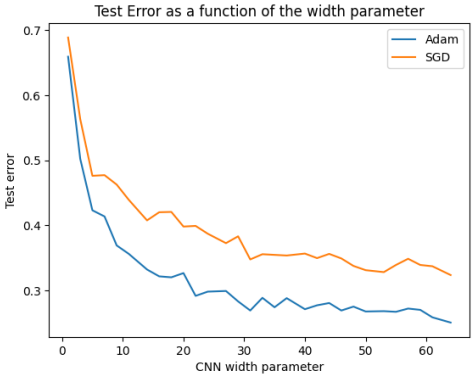
(b) Train Error

Fig. 8: Reproduced Fig 4

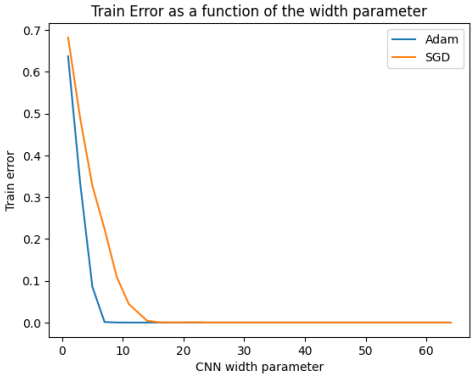
B. *Experiment 2 (Fig 6)*

The second experiment involves the comparison of the two optimizer performances, respectively Adam and SGD. Originally, the number of epochs is different for the two optimizers, with a ratio

being $\frac{1}{200}$ in favor of SGD. Keeping this ratio is not acceptable regarding the conditions of the project, since the number of epochs does not exceed 100. Performing only 1 epochs lacks of sense since, no convergence will be observed. The same number of epochs is set (100 epochs) with the tuning presented in II. CIFAR10 dataset has been used for this experiment. Complete details can be found in the notebook on Github. The results are presented in Figure 9. Comparing with the original, it can be seen that the general shape emerges for the train error, while the two curves are switched for the test error, with Adam outperforming SGD. It's not surprising as Adam can have faster convergence than SGD in small number of epochs. The results could be very different if the number of epochs were differrent for the two optimizers.



(a) Test Error



(b) Train Error

Fig. 9: Reproduced Fig 6

VI. Critical Analysis

This section aims to critique the results obtained from the different experiments. Due to the limitations of computational resources, the original experimental conditions could not be fully reproduced. Nevertheless, some conclusions can be drawn from the figures.

A. Experiment 1

The double descent phenomenon is not clearly observed in the figure, but the general pattern emerges, indicating that very large models are advantageous compared to simpler ones. This observation aligns with the concept of *over-* and *under-parametrized regimes* derived from Equation 4. Notably, the *critically parametrized regime* (i.e., double descent) is missing, as mentioned in Section V, resulting in a single descent. Additionally, the general shape of the noise curves corresponds; as more noise is applied, the models make more errors.

B. Experiment 2

The general shape found in this project corresponds to the one obtained in the original paper. The main difference lies in the distance between the two curves. In this project, the two optimizers are closer to each other than in the original experiment. This can be partly explained by the fact that the number of epochs is the same in this project, while it differed for both optimizers in the original paper (Section IV). Furthermore, reducing the size of the dataset decreases the quality of the data and, consequently, the quality of the results, explaining the less convincing outcomes observed for both experiments.

VII. Summary

This project has highlighted the significant challenges associated with replicating experiments. Discrepancies in the experimental setups prevented a thorough verification of all statements. Despite these differences, the project successfully demonstrated the general behavior, indicating that larger models do not necessarily suffer from overfitting. To achieve a more accurate reproduction of the experiment, it would be essential to utilize better computational resources to reduce the required time.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. arXiv:1512.03385.

[2] Nicolas Flammarion Martin Jaggi. Bias-variance decomposition, 2023. EPFL Machine Learning Course, Fall 2023, Week 4.

[3] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *CoRR*, abs/1912.02292, 2019. URL: http://arxiv.org/abs/1912.02292, arXiv:1912.02292.

[4] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. arXiv:1511.08458.

[5] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/f2925f97bc13ad2852a7a551802feea0-Paper.pdf.