Understanding the Double Descent

Artur M. Oliveira
Universidade Federal de Minas Gerais
PPGEE
Belo Horizonte, Brasil
oliveiraarturm@hotmail.com

***Abstract*—In recent years, one of the most relevant topics of discussion in the field of machine learning has been the topic of *Double Descent*. *Double Descent* refers to the decay in test error when the model is tested in the overparameterization regime — when there are more parameters than the number of training samples, also known as the region beyond the interpolation point. This behavior has been challenging the classical statistical learning approach, which is widely accepted, in part due to the work *Neural Networks and the Bias/Variance Dilemma* [1], in which the assumption is made that as the complexity of the model increases, so does its capacity, which generally implies a higher test error.**

The main motivation of this work will be to show how deep learning can achieve good performance on tests even in models with high capacity.

I. Introduction

Deep learning has revolutionized a variety of applications in both industry and research. Many applications of DNNs (*Deep Neural Networks*) in fields such as healthcare, robotics, information security, etc., have been developed.

This approach of neural networks, with many parameters, has been studied over the years, even though there have been some gaps in the research. As early as 1965, *Ivakhnenko and Lapa* [2] developed models with polynomial activation units. Another widely adopted neural network architecture today, convolutional networks, which can be considered a class of deep networks, was first developed by *Fukushima* [3] in 1979. The network contained multiple convolutional and *pooling* layers, similar to modern networks. Until that point, there was a certain computational difficulty in training multi-layer architectures.

In 1985, *Rumelhart, Hinton, and Williams* [4] published a paper in which they applied *backpropagation*—already developed in the 1970s but not applied to neural networks—for training multi-layer networks. This paper marked the resurgence of interest in neural networks.

Driven by this work, *LeCun* [5], in 1989, used the *backpropagation* technique in conjunction with convolutional networks for digit recognition. Other relevant works that followed included the development of RNNs (*Recurrent Neural Networks*) and SVMs (*Support Vector Machines*).

Few works followed after this period, as due to the limited computational power at the time and the availability of data, SVMs, which were easy to train and performed well, became the preferred choice. Moreover, problems such as *vanishing gradient* prevented very deep networks from being trained quickly. In 2012, *Krizhevsky, Sutskever, and Hinton* [6] proposed an architecture characterized by the use of ReLUs and dropout for regularization. The network achieved excellent results in the ILSVRC-2012 ImageNet competition.

After this period, DNNs became the dominant models in major *big techs*. However, their high capacity challenged previously widely accepted assumptions in statistical learning, with the most "challenged" being the bias-variance dilemma. In the bias-variance dilemma, the higher the variance of an estimator (e.g., DNNs), the higher the test error typically is. Much of the variance is associated with the complexity of the model, such as the number of parameters. However, deep networks with many parameters still manage to generalize well. This good generalization can be observed through the occurrence of *Double Descent*, which is characterized by a decrease in test error after the interpolation point. This shows that test error is not always tied to model complexity. To demonstrate this behavior, experiments and discussions will be conducted in this paper to explain how this behavior can be explained. It will be argued that the high capacity for generalization in neural networks can be well explained by the training methods, especially *SGD* (*Stochastic Gradient Descent*), which acts as a regularizer, and by the fact that, when there are more parameters than training samples, there are infinite solutions for the network weights, which can have norms similar to those found in the classical region. Other points, such as optimization, will also be discussed.

II. Organization

In Section III, the explanations of the SGD methods and how the increase in the number of parameters in the network can help in learning a dataset will be provided. A review of some works that focused on explaining *Double Descent* will also be included.

In Section IV, some tests will be performed and the demonstration of how these factors, SGD and the increase in parameters, affect the test error, leading to *Double Descent*, will be shown.

Finally, in Section V, the conclusions will be drawn.

III. Understanding *Double Descent*

In this section, the main works discussing *Double Descent* will be mentioned, along with a discussion of the factors that most influence its behavior.

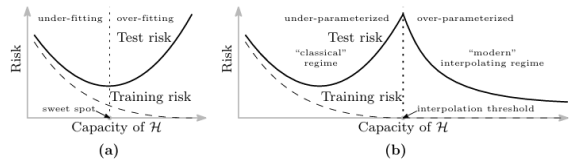

Fig. 1. Curves for training risk (dashed line) and test risk (solid line). (a) The classical U-shaped curve, resulting from the trade-off between bias and variance. (b) The *double descent* risk curve, combining the U-shaped curve (classical regime) with the behavior observed when using high-capacity function classes (modern interpolation regime), separated by the interpolation threshold. Predictors to the right of this threshold have training risk equal to zero.

The bias-variance dilemma and *double descent* have been and continue to be studied by several authors in the field of statistical learning and *machine learning*. One of the most widely cited and recognized works in the area is the work of *Belkin* [7].

In this work, *Belkin* attempts to provide evidence for the existence and ubiquity of *double descent*. According to the author, the classical theory of statistical learning focused on finding the optimal point of the error curve, typically found when a classifier belongs to a function class $\mathcal{H}$, which is not small enough to have the problem of *underfitting* nor large enough to have the problem of *overfitting*. However, modern machine learning methods are complex enough to interpolate the training data, meaning they achieve near-zero or zero training error and still have good generalization error, defined as the difference between the test error and the training error on data never seen by the trained model. This behavior can be seen in Fig. 1 [7].

Belkin states that by going beyond the interpolation point in a high-capacity function class, there are more solutions, and thus it is possible to find functions with norms comparable to those of the optimal point in the classical regime even in the overparameterization regime. According to him, there is no guarantee that the minimal norm solution is found in the classical regime. This can be interpreted as having a more biased solution in the *double descent* regime. His work mentions that SGD can result in a non-optimal solution beyond the interpolation point, as in a non-convex surface, the algorithm may converge to a local minimum. However, in the final sections, it is stated that SGD is easy to optimize and frequently finds a global minimum in the overparameterization regime.

Other relevant works have been done to understand the behavior of *double descent*. According to Nakkiran et al. [8], other factors may influence *dd*. According to the paper, *double descent* can occur due to the new complexity metric they define, EMC (Effective Model Complexity). When EMC is sufficiently larger than the number of examples, *dd* is more likely to occur. The authors also mention that *dd* is more likely to occur in situations where there is noise in the labels and less likely to occur when there is implicit regularization, such as early stopping.

In [9], the authors discuss how DNN models manage to generalize well. According to them, model capacity measures alone, such as VC dimension and Rademacher complexity, are incapable of explaining why DNNs can achieve small test errors and still have zero training error. The authors conducted a series of experiments and showed that DNN models can memorize datasets with inputs and outputs that have no correlation and still generalize well. This means, according to the authors, that factors other than model complexity are the true reasons for the small generalization error of DNNs. As in [7], the role of SGD as a regularizer is highlighted, and how the large number of parameters in DNNs is not an optimization problem in terms of training time.

IV. Experiments

A. Methodology

Inspired by the referenced works, the goal is to reiterate through some experiments the most relevant points cited by the authors as explanations for the *dd* phenomenon. Among the main objectives are:

- Verify SGD as a regularizer and as a fast training method. To do this, a comparison will be made with another training method, the pseudoinverse for ELMs.
- Verify the test error for the Breast Wisconsin, Seno, and MNIST datasets.
- The behavior of *dd* on the datasets.
- Verify the norm of the solutions in the overparameterization regime.
- Verify how explicit regularization affects *dd*.

B. Results

1) Breast Wisconsin: The first experiment was to reproduce *dd* on the Breast Cancer dataset [10]. The dataset consists of 30 features and a binary output, indicating whether the cancer is malignant or benign. ELMs were trained with 1, 5, 10, 30, 50, 100, 300, 398, 1000, 10000, 50000, and 200000 units in the hidden layer. The number of samples considered for testing was 398. As can be seen in Figure 2, the test error for each number of units in the hidden layer (considered here as the unit, minus the average plus the standard deviation of correct classifications for 10 trainings) exhibits the *dd* behavior. For this training method, time is a limiting factor, as it took about 7 minutes to generate the figure. More complex networks and a larger training set would take much longer.

In Table I, it can be observed that beyond the interpolation point, the norm of the network weights decreases. This helps in *dd*. However, the optimal solution still remains in the classical regime.

2) Breast Wisconsin with SGD and Backpropagation: The second experiment was to train a one-layer neural network with a variable number of units in the hidden layer. Specifically, the values 2, 3, 4, 5, 6, 7, 15, 20, 50, 100, 1000, 10000 were used. The training algorithm used was backpropagation with SGD to optimize the *Binary Cross Entropy*. The dataset is the same as the previous example. The test error was defined as the ratio between the total number of classification errors

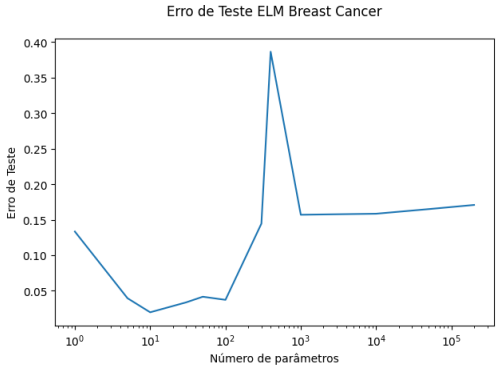

Fig. 2. ELM Breast Test Error

TABLE I
BREAST ELM RESULTS WITH DIFFERENT PARAMETERS

Norm	Accuracy	Number of Parameters
3.301	0.860	1
5.091	0.947	5
6.659	0.978	10
14.081	0.966	30
23.847	0.962	50
71.869	0.965	100
628.384	0.841	300
13735.467	0.579	398
395.550	0.819	1000
90.429	0.833	10000
39.471	0.831	50000
19.666	0.830	200000

in the test set and the total number of predictions made. The results of test error vs. number of parameters can be seen in Figure 3. The time spent plotting the figure was 1min29s.

In Table II, the total weight norm, accuracy, and number of parameters for this network configuration can be observed. It is noted that the weights do not increase significantly as the number of parameters increases, and in the overparameterization regime, the network still achieves high accuracy. This behavior is a strong indication of SGD as a regularizer.

3) Sine with SGD: The third experiment was the approximation of the sine function with added noise having a mean

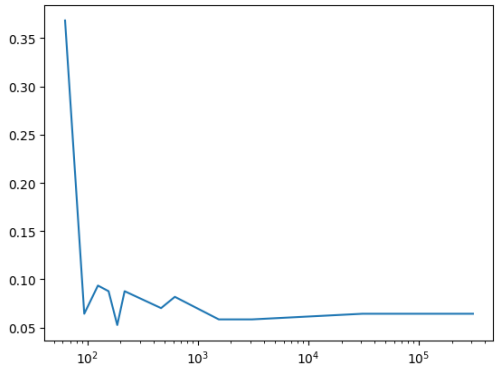

Fig. 3. SGD Breast Test Error

TABLE II
BREAST SGD RESULTS WITH DIFFERENT PARAMETERS

Norm	Accuracy	Number of Parameters
3.387	0.631	63
6.030	0.935	94
6.334	0.906	125
6.231	0.912	156
6.960	0.947	187
7.064	0.912	218
8.692	0.930	466
9.466	0.918	621
10.341	0.942	1551
11.005	0.942	3101
11.808	0.936	31001
11.933	0.936	310001

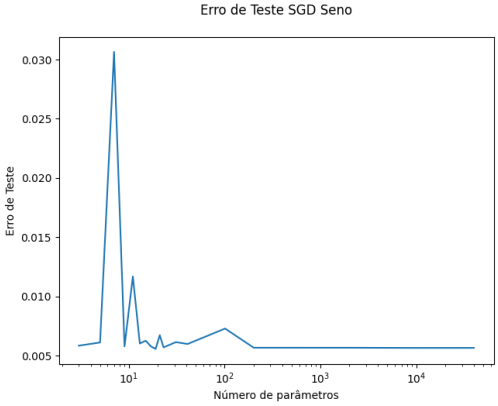

Fig. 4. SGD Test Error for Sine

of 0 and a standard deviation of 0.05. A total of 1000 training samples and 200 test samples were considered. The number of units in the hidden layer was varied according to the following values: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 15, 20, 50, 100, 1000, 5000, 20000. This experiment was similar to the second one. However, since the data is continuous, the mean squared error cost function was used to define the parameters. The evaluation metric for the test set was also the mean squared error. The test error graph can be observed in Fig 4. The time spent to plot the figure was 1 minute and 46 seconds.

In Table III, the total weight norm, accuracy, and number of parameters for this network configuration are shown.

4) Sine with SGD and Regularization: The fourth experiment was to perform the same sine function from the previous experiment but using weight decay (L2 regularization). The weight decay factor was set to 0.001. The result of the test error, defined in the same way as the previous experiment, is shown in Fig 5.

In Table IV, the total weight norm, accuracy, and number of parameters for this network configuration are shown.

5) MNIST Trained with SGD and Backpropagation: The final experiment was the classification of digits using the MNIST dataset. The network configuration was a hidden layer with a number of units varying according to the following values: 2, 3, 4, 5, 6, 7, 15, 20, 50, 55, 60, 65, 70, 100. The output layer remained fixed with 10 outputs. SGD was used

TABLE III
SINE SGD RESULTS WITH DIFFERENT PARAMETERS

Norm	Accuracy	Number of Parameters
1.788	0.994	3
2.267	0.994	5
1.579	0.969	7
3.005	0.994	9
3.041	0.988	11
2.705	0.994	13
2.384	0.994	15
2.493	0.994	17
2.925	0.994	19
2.904	0.993	21
3.068	0.994	23
3.073	0.994	31
2.810	0.994	41
2.902	0.993	101
3.138	0.994	201
3.003	0.994	2001
3.068	0.994	10001
3.050	0.994	40001

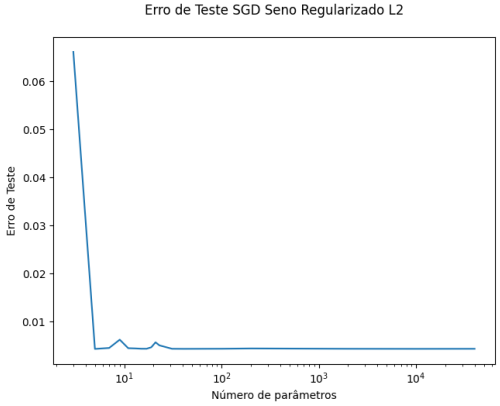

Fig. 5. SGD Regularized Sine Test Error

TABLE IV
SINE SGD REGULARIZED RESULTS WITH DIFFERENT PARAMETERS

Norm	Accuracy	Number of Parameters
2.127	0.934	3
2.603	0.996	5
2.908	0.996	7
2.477	0.994	9
2.780	0.996	11
2.572	0.996	13
2.666	0.996	15
2.988	0.996	17
2.482	0.995	19
2.291	0.994	21
2.721	0.995	23
2.870	0.996	31
2.889	0.996	41
3.159	0.996	101
2.924	0.996	201
2.998	0.996	2001
3.005	0.996	10001
3.001	0.996	40001

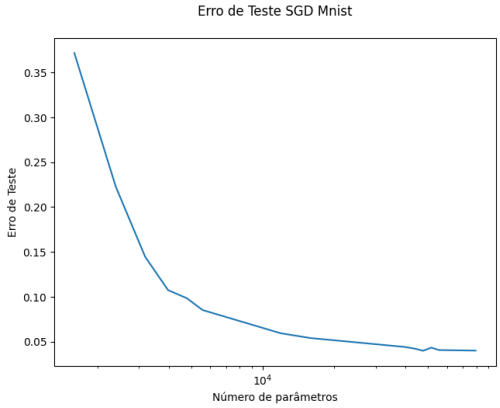

Fig. 6. SGD MNIST Test Error

as the optimizer and the cost function to minimize was *sparse categorical crossentropy*. The test error was defined as the ratio between the total number of classification errors in the test set and the total number of predictions made. The results of test error vs. number of parameters can be observed in Fig 6. The time spent performing the calculations was 11 minutes, a reasonable time considering the time spent by ELM and the number of data in the MNIST dataset. The training error was 0.9649.

V. CONCLUSIONS

As shown by the first experiment using ELMs, the test error decreases after the interpolation point, when the number of parameters is equal to 398, along with the norm of the weights. This indicates that the method seeks solutions with smaller norms when there is overparameterization. In other words, in the parameterization regime, the model becomes more biased, as the higher complexity of the model is reduced by the small norm associated with the parameters, resulting in a model with lower complexity. This fact can be observed in Table I.

In the second experiment, SGD was used together with backpropagation. It can be observed that SGD acts as an implicit regularizer, preventing a peak in the test error when the interpolation point is reached. Another point to highlight is the calculation time. With SGD, calculations with similar numbers of parameters were completed in 1min29s, compared to 7 minutes for the ELM. A similar result occurred in experiment 3.

The fourth experiment involved a network trained by backpropagation, regularized, and trained with SGD. It can be observed that the test error curve quickly stabilizes. There is no peak in the test error at the interpolation point. This is because regularization prevents the model's variance from increasing too much, as the norm of the weights is penalized, making the model more biased.

The final experiment conducted was digit classification on the MNIST dataset with the network trained by backpropagation and SGD. The result shows that SGD again acts as an implicit regularizer, seeking models with small norms even

after the interpolation point, when the number of parameters exceeds 60,000. Additionally, the high performance on the training set did not result in poor performance on the test set.

It can be concluded, therefore, that as mentioned in [9], the number of parameters and the capacity of the model are not good measures for generalization error. The implicit regularization of SGD, by selecting random fractions of the training set, prevents all the complexity of the data from being captured. This translates into regularization that prevents the model from having too many significant parameters (with high weights). The same happens with ELMs, which manage to find solutions with smaller norms than those found just before the interpolation point. However, there is no guarantee that the solution with the lowest test error lies in the overparameterization region, as stated by [7]. According to [8], *dd* is less likely to occur when there is implicit regularization, which was observed in the experiments. No *dd* occurred when SGD was applied to MNIST, Breast Wisconsin, and sine datasets. With explicit regularization, *dd* did not occur, as shown in experiment 4.

To conclude, *dd* does not contradict the bias-variance dilemma, since the number of parameters alone cannot define the flexibility of a model. Larger models but with smaller norms have lower variance. The overparameterization regime does not necessarily imply a lower test error.

REFERENCES

[1] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma," *Neural Computation*, vol. 4, no. 1, pp. 1–58, 1992.

[2] A. G. Ivakhnenko and V. G. Lapa, *Cybernetic Predicting Devices*. CCM Information Corporation, 1965.

[3] K. Fukushima, "Neural network model for a mechanism of pattern recognition unaffected by shift in position—neocognitron," *Trans. IECE*, vol. J62-A, no. 10, pp. 658–665, 1979, the first deep convolutional neural network architecture, with alternating convolutional layers and downsampling layers. In Japanese. English version.

[4] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[7] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, p. 15849–15854, Jul. 2019. [Online]. Available: http://dx.doi.org/10.1073/pnas.1903070116

[8] P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever, "Deep double descent: Where bigger models and more data hurt," 2019. [Online]. Available: https://arxiv.org/abs/1912.02292

[9] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2017. [Online]. Available: https://arxiv.org/abs/1611.03530

[10] e. a. Wolberg, "Breast Cancer Wisconsin (Diagnostic)," UCI Machine Learning Repository, 1993, DOI: https://doi.org/10.24432/C5DW2B.