Study on Overfitting

A review on Overfitting in Deep Learning models

Mattia Ceccarelli

June 2020

The project aim is to study how overfitting can represent a problem in common data analisys tasks and how to detect and avoid it in Deep Learning Applications.

The datasets used over the whole study are *MNIST-784*, composed of 70.000, 28x28 grey-scale images containing hand-written digits; and *CIFAR-10*, composed of 60.000, 32x32 colored images comprehensive of 10 classes of objects.

The study will also focus on recent discoveries regarding overfitting in deep learning.

All the data obtained in the case of Deep Learning models are obtained by using the famous python library Tensorflow [1]

Overfitting in data analysis

In the context of data analysis overfitting is a common problem that arises from different factor. The main cause of overfitting is noise: indeed, by definition, an analytical or statistical model is "overfitted" when the outcome of the algorithm is overly affected by the noise in the data, which means the model begin to describe the random error in the data rather than the relationship between variables.

This ordinary but important flaw of machine learning and deep learning model brings a series of consequences that have as a secondary effect to reduce the effectiveness of the model, in particular in data it has never seen before: this phenomenon is called lack of Generalizations capabilities.

An example of overfitting in analytical method is shown in figure 1 found when using a simple polynomial fit on randomly generated data from $f(x) = 5 * sin(x * 9/10) + \nu$ where ν is a normally distributed random variable with $\mu = 0$ and $\sigma = 1$. The results obtained in the case of high order polynomial can be considered as a form of overfitting since the model tries to exactly fit every random point.

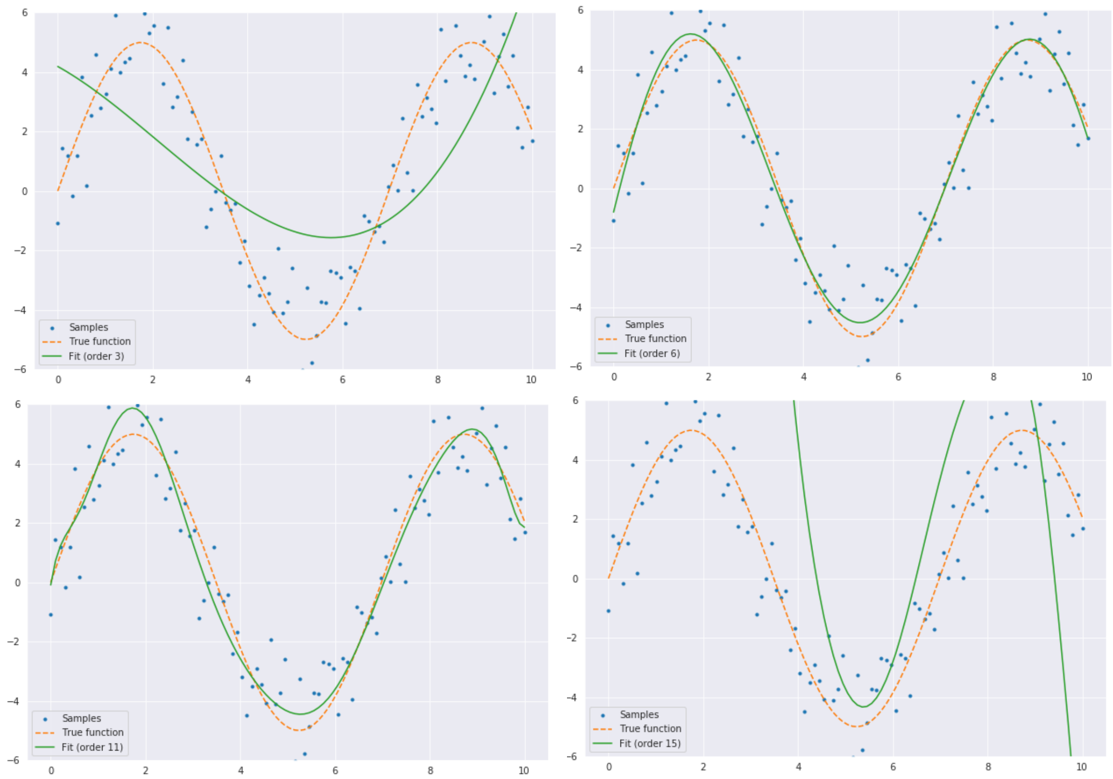


Figure 1: *The image shows in order from left to right and from up to down: underfitting of polynomial with order 3, best fit with polynomial of order 6, and overfitting with polynomial of order 11 and 15*

Given the fact that noise can't be removed from data a priori, many different techniques has been developed to avoid overfitting. The most widely used is the simple divsion between training data and test data, which allow more control over the performances of the models. Figure 2 shows some variation of the same concept: keep train and test data separated. This greatly reduce the risk of overfitting and the introduction of Validation sets helps with the fine-tuning of hyper-parameters. The advantages come at the cost of computational efficency.

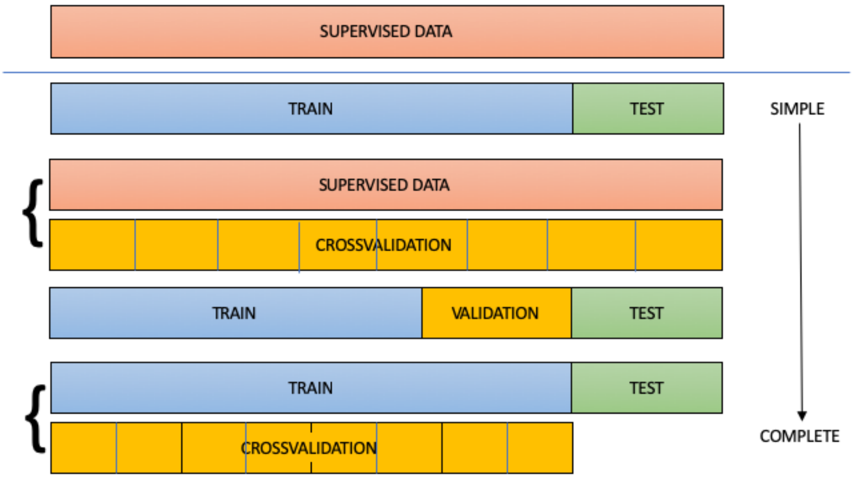


Figure 2: *The image show the most common practices in machine learning application to maintain top generalization performances.*

Other causes for overfitting may come from the dataset:

- individuals in the test set can have bad values in the predicting attributes and/or in the class label
- Lack of representative instances some situations of the real world can be underrepresented, or not represented at all, in the training set. This situation is quite common.

As a rule, a good hypothesis has low generalization error i.e. it works well on examples different from those used in training.

In traditional machine learning methods, the "U-shaped" risk curve in figure 3 is used to measure the bias-variance trade-offs and quantify how generalizable a model is [6]. As the model turns larger (more parameters added), the training error decreases to close to zero, but the test error (generalization error) starts to increase once the model complexity grows to pass the threshold between “underfitting” and “overfitting”

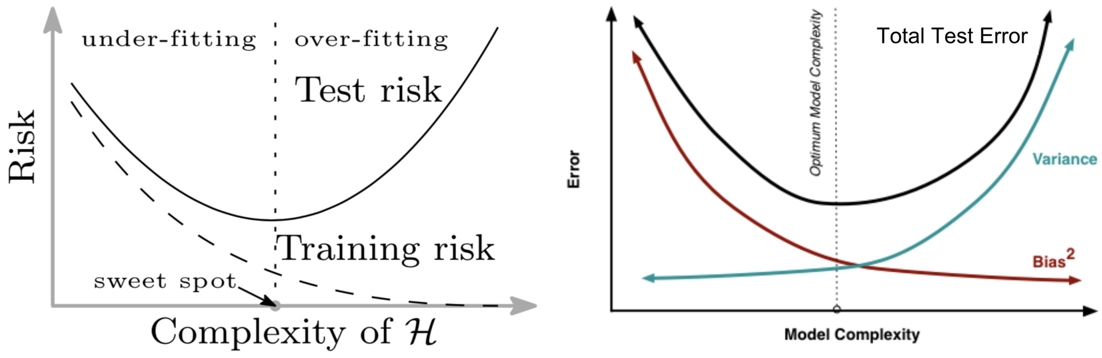


Figure 3: *Classical view on Bias-Variance Trade-off in deep learning. The ideally trained model comes from a balance between under-fitting and over-fitting. Images from [2]*

Overfitting in Deep Learning

In deep learning, overfitting is essentially threated the same as for other machine learning metods. It's easy to see how overfitting can appear in simple applications: in figure 4 is shown the charateristic trend of the loss in a overfitting model.

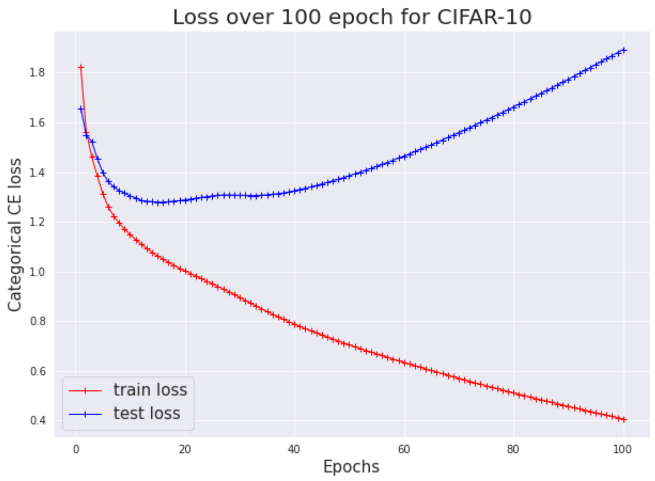


Figure 4: *The image shows a common overfitting behaviour in Deep Learning Applications. The cross entropy loss for the training set keeps a steady descent while the test set loss start increasing.*

In this case, the Gradient Descent is performed on the categorical crossentropy loss, defined as :

$$CE = -\sum_{i=1}^{classes} t_i log(x_i) \tag{1}$$

Where t_i is the true label of the input, while x_i is the predicted label of the input. As shown in the graph below, it is clear how, if not monitored, overfitting can and does lead to contradictory results (very good on training set, very bad on test set).

In figure 5 is reported the behaviour of the classification error in the same experiment. The trend is clearly different from the previous one and the error stays more or less constant after the model reaches the mininum.

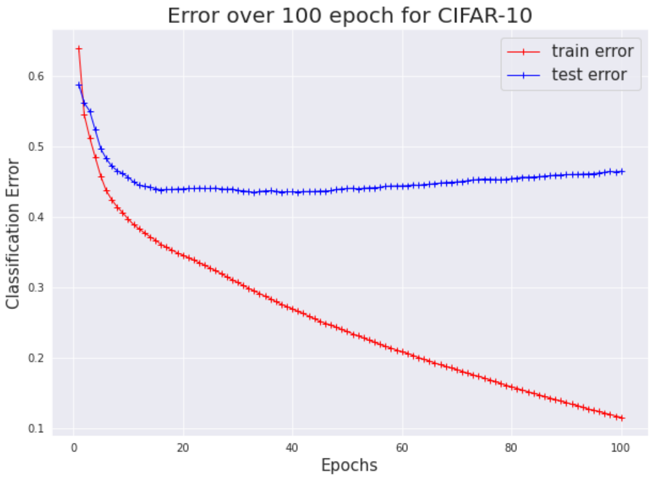


Figure 5: *The image shows a common overfitting behaviour in Deep Learning Applications. The classification error in the test set remains almost constant while the classification error in the training set tends to zero after the "sweet spot"*

The difference in the two graphs is one of the main reasons why accuracy is not a good target function while training deep learning model and, in general, to report machine learning results. Indeed, it only compares a predicted score t to some cutoff c, which is not a proper scoring rule and conceals important information about model fitness. Improper scoring rules such as proportions accuracy, sensitivity, and specificity are not only arbitrary (in choice of threshold) but are improper, i.e., they have the property that maximizing them leads to bad models, inaccurate predictions, and selecting the wrong features.

Cross-entropy loss is more viable than accuracy because it is sensitive to "how wrong" the results are: if the label is 1, but $t = 0.9$, the cross-entropy is lower than when the label is 1 but $t = 0.1$. The phenomenon when comparing these two graphs, accuracy is flat but loss is increasing, happens because $t > c$.

Recent Development

In recent year, the view on overfitting in Deep Learning applications is drastically changing and this is due to new development in the theory of learning. Practitioners routinely use modern machine learning methods, such as large neural networks and other non-linear predictors that have very low or zero training risk. In spite of the high function class capacity and near-perfect fit to training data, these predictors often give very accurate predictions on new data. Indeed, this behavior has guided a best practice in deep learning for choosing neural network architectures, specifically that the network should be large enough to permit effortless zero loss training (called interpolation) of the training data. Moreover, in direct challenge to the bias-variance trade-off philosophy, recent empirical evidence indicates that neural networks and kernel machines trained to interpolate the training data obtain near-optimal test results even when the training data are corrupted with high levels of noise. [2].

In figure 6 is shown the new Risk-Model Complexity curve proposed from the authors of the paper.

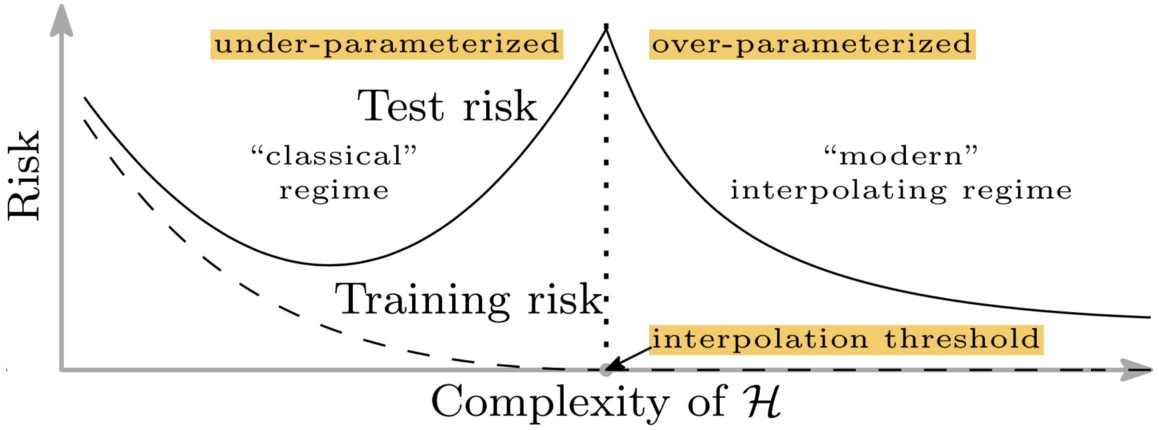


Figure 6: *New bias-variance curve proposed in [2].*

The double descent risk curve, which incorporates te U-shaped risk curve ("classical regime") with the observed behaviour from using high capacity function classes ("modern regime"), seprated by a interpolation threshold. The predictors to the right have zero training risk.

In figure 7 is shown the loss and the miss-classification error during the training of a one hidden-layer fully-connected neural network as a function of the

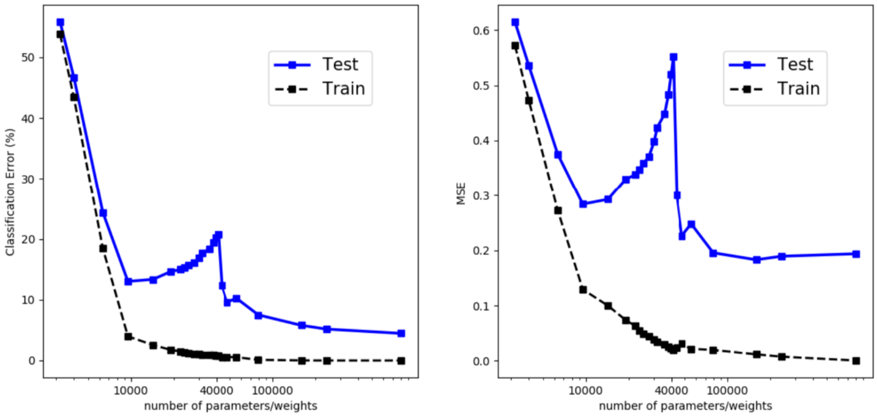


Figure 7: *Training and evaluation errors of a one hidden layer fc network of different numbers of hidden units, trained on 4000 data points sampled from MNIST. From [2]*

The paper claimed it is likely due to two reasons :

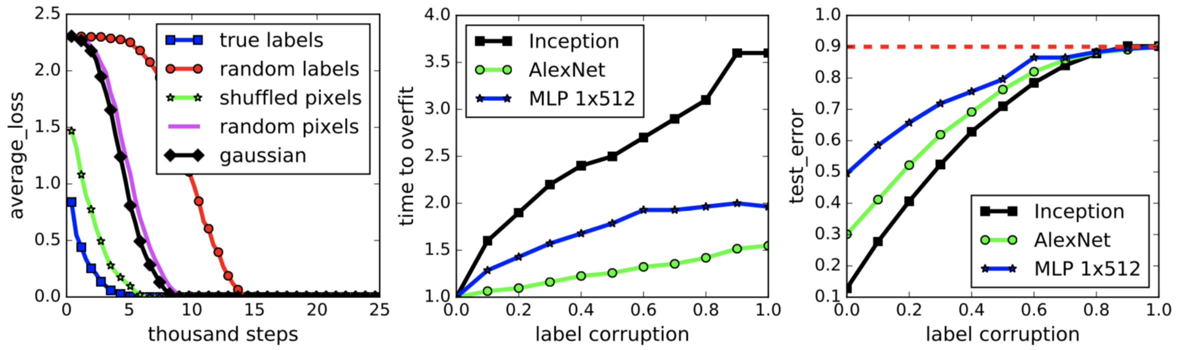
- The number of parameters is not a good measure of inductive bias, defined as the set of assumptions of a learning algorithm used to predict for unknown samples.
- Equipped with a larger model, we might be able to discover larger function classes and further find interpolating functions that have smaller norm and are thus “simpler”.

Regularization is a common way to control overfitting and improve model generalization performance. Interestingly some research [7] has shown that explicit regularization (such as data augmentation, weight decay and dropout) is neither necessary or sufficient for reducing generalization error.

model	# params	random crop	weight decay	train accuracy	test accuracy
Inception	1,649,402	yes	yes	100.0	89.05
		yes	no	100.0	89.31
		no	yes	100.0	86.03
		no	no	100.0	85.75
(fitting random labels)		no	no	100.0	9.78

Figure 8: *The training and test accuracy (percentage) of Inception models on the CIFAR10 dataset (in the original paper more models are considered). Performance with and without data augmentation and weight decay are compared. The results of fitting random labels are also included. From [7].*

Two-layer neural networks are universal approximators, so given a sufficiently complex structure, they are able to learn any function. It has been proven that they are able to learn unstructured random noise as well, as shown in [7] and [5]. If labels of image classification dataset are randomly shuffled, the high expressivity power of deep neural networks can still empower them to achieve near-zero training loss. These results do not change with regularization terms added. Figure 9 shows some example of this behaviour.



(a) learning curves (b) convergence slowdown (c) generalization error growth

Figure 9: *Fit models on CIFAR10 with random labels or random pixels: (a) learning curves; (b-c) label corruption ratio is the percentage of randomly shuffled labels. Image from [7]*

An interesting interpretation of those behaviour is given by the lottery ticket hypothesis [4]. The article states that only a subset of network parameters have impact on the model performance and thus the network is not overfitted. The lottery ticket hypothesis suggest that a randomly initialized, dense, feed-forward network contains a pool of subnetworks and among them only a subset are “winning tickets” which can achieve the optimal performance when trained in isolation. Namely :

“**The Lottery Ticket Hypothesis.** *A randomly-initialized, dense neural network contains a subnetwork that is initialized such that — when trained in isolation — it can match the test accuracy of the original network after training for at most the same number of iterations.*” [4]

The paper identify a method to find "winning tickets" :

1. random initialization of a neural network with parameters θ_0
2. Train the net for j iterations, reaching state θ_j
3. Prune a percentage of parameters θ_j.
4. Reset remaning parameters to values in θ_0, creating the winning ticket

The authors successfully found "winning tickets" for fully-connected and convolutional structure for MNIST and CIFAR-10 across different optimization strategy (SGD, Momentum and Adam). This hypothesis opens up new ways about interpreting deep neural networks results.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine learning practice and the bias-variance trade-off. *ArXiv e-prints*, Sept. 2019.

[3] R. Caruana, S. Lawrence, and C. Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. volume 13, pages 402–408, 01 2000.

[4] J. Frankle and M. Carbin. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. *ArXiv e-prints*, Mar. 2019.

[5] T. Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Boix, J. Hidary, and H. Mhaskar. Theory of Deep Learning III: explaining the non overfitting puzzle. *ArXiv e-prints*, Jan. 2018.

[6] L. Weng. Are deep neural networks dramatically overfitted? *lilianweng.github.io/lil-log*, 2019.

[7] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding Deep learning requires Rethinking Generalization. *ArXiv e-prints*, Feb. 2017.