Data Science Semester Project

Author: Julien Heitmann

Understanding neural network training dynamics via pruning

Content

1 Introduction

One of the suprising results of training deep neural networks with many more parameters than training samples is that one can achieve zero-training loss but still get good generalisation [3]. While statistical learning theory suggests that such heavily over-parametrised networks generalise poorly without further regularisation, experiments show that in most cases, increasing the number of parameters does not worsen the capability to generalise. This paradox is one of the big unresolved questions of deep learning, and understanding the phenomena behind it would deliver a lot of insights about why neural networks work so well. The question arises why the network doesn't exploit its full expressive power to overfit the training dataset. Modern network architectures, which can have up to 100x more trainable parameters than training samples [4], most certainly are capable of learning the entire dataset, independently of the labels. It has been shown that zero training loss can even be achieved when the training labels are randomly shuffled [5], thus preventing the learning of underlying features of the data that are sufficient to solve the classification task at hand.

Surely one must look at the interplay between the dataset properties, the network architecture and the optimization algorithm, but the latter in particular seems to play a crucial role, as it somehow consistently favors solutions in the optimization landscape that generalise, over solutions that overfit the data and in some cases do not learn a representation of the data. Is it a property of the optimization algorithm, does stochastic gradient descent optimization lead to sparse solutions? If yes, how do these sparse solutions look like? Do trained neural networks rely on single components (nodes or filters), even with a growing number of parameters? Or does each of the components contribute equally to the estimated function, thus reducing the contribution when there are more parameters?

An interesting way to measure the importance of a trained network's individual components is pruning. When single components are removed from a network's architecture, i. e. they do not contribute anymore to the output of the network, the difference in validation accuracy can be measured, which is a good indicator of the network's performance. Pruning can therefore be used as a means to get a better understanding of what is happening in a trained network, to identify important parts and not so important ones. But

pruning can also be seen as an end, if done properly it might yield smaller architectures, which result in computation speed-ups and smaller memory footprints at evaluation. Assuming that overparametrisation is necessary to achieve good performance, and that some components of an overparametrised network are obsolete after training or can be removed without a negative impact on validation accuracy if the network is retrained, then pruning might even be necessary to get efficient architectures that scale.

This project aims to explore different hypotheses about neural network training dynamics, with a particular focus on pruning. The goal is to identify both similarities and differences in the theories, and provide evidence in favor or against them in a series of experiments. Moreover, some visualisation tools will be introduced that will help getting a better intuition for the complex process that is neural network training.

2 Related work

If good generalisation can be achieved by an overparametrised network, there must be some mechanism that prevents the model to overfit the dataset, and learn for instance features that would be considered as noise when it comes to the classification task, because they do not provide any information about a sample's affiliation to a certain class. With an increasing number of parameters, more complex decision boundaries can be represented, so in order to generalise, the solutions obtained by the optimization algorithm of choice must be "sparse" in a way, which is yet to be defined. We can ask specific questions when reasoning about sparsity: within an overparametrised network, is it just a subnetwork that is trained, particularly receptive to training and with an inductive bias that makes it "trainable"? Or is there a collapse of the weight vectors around just a few directions, relevant to the classification task at hand? These questions correspond to different explanations of the underlying mechanisms of neural network training, and have been studied in the past. The first one, more formaly known as the "Lottery ticket hypothesis" [2], emphasises the importance of weight initialisation, which leads to the formation of subnetworks that can be trained efficiently. The second one states that overparametrisation works well because of better feature exploration and weight clustering [1].

2.1 The lottery ticket hypothesis

The hypothesis, improvements such as late resetting. Why it matters, how it could be used to design better initialization methods, that lead to faster convergence, and maybe a higher test accuracy. Note that this mainly applies to unstructured pruning, which doesn't come with computational speed-ups.

2.2 Pruning networks

Structured vs unstructured pruning, suprisingly leads to better generalisation, pruning as a noise signal. Speed-up, memory foot-print. Need over-parametrised network for better exploration of hidden layer space (give examples), once the model converges / during training, remove useless components. Suprising results: prune components with highest magnitude, can lead to better test accuracy after re-training the pruned network.

2.3 Alignment of weight vectors during training

Coming back to the second question: XOR-problem, sparse subspace spanned by trained vectors. Question arises: will optimization algorithms such as SGD enforce this aligment, therefore preventing overfitting? Interplay between vector aligment and "trainable" vectors. Goal is to explore and go beyond, understanding this mechanism will help us design better optimization algorithms, pruning methods, architectures, initialization methods, etc. Collapse at last layer?

3 Approach

3.1 Formal background

Formally define frame potential, generalization error, network architecture, optimization algorithms.

3.2 Visualisation

What we expect to observe, how the visualisation works, which parameters can be set.

4 Experiments

5 Discussion

References

[1] Alon Brutzkus and Amir Globerson. "Why do Larger Models Generalize Better? A Theoretical Perspective via the XOR Problem". In: *International Conference on Machine Learning*. 2019, pp. 822–830.

[2] Jonathan Frankle and Michael Carbin. "The lottery ticket hypothesis: Finding sparse, trainable neural networks". In: *arXiv preprint arXiv:1803.03635* (2018).

[3] Behnam Neyshabur et al. "Towards understanding the role of over-parametrization in generalization of neural networks". In: *arXiv preprint arXiv:1805.12076* (2018).

[4] Sergey Zagoruyko and Nikos Komodakis. "Wide residual networks". In: *arXiv preprint arXiv:1605.07146* (2016).

[5] Chiyuan Zhang et al. "Understanding deep learning requires rethinking generalization". In: *arXiv preprint arXiv:1611.03530* (2016).