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 (Neyshabur et al. 2018). 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. In fact, it often happens that, when training two models with $N1$ and $N2$ neurons respectively, $N1 > N2$, training will converge to zero-training loss in both cases, but the larger model will generalise better. 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 (Zagoruyko and Komodakis 2016), 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 (Zhang et al. 2016), 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 favours 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 generalisation, 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 favour 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 "simple" in a way, which is yet to be defined.

While it is curious that large neural networks do not overfit the dataset, even in the presence of massive label noise (Rolnick et al. 2017), it is even more surprising that increasing the number of parameters can decrease the generalization gap (Neyshabur et al. 2018). In other words, given a classification task and a certain model architecture with a fixed number of layers but adjustable layer widths, a certain degree of layer-wise overparameterization might be both sufficient and necessary to achieve good performance after training.

2.1 Self-regularization of overparameterized models

The inability of statistical learning to explain and predict the properties of neural networks is not a new phenomenon. In particular, when it comes to measuring complexity, traditional measures such as the VC dimension or the number of parameters fail to explain why increasing the number of parameters of a model does not necessarily increase the generalization gap. More recently new measures have been introduced that aim to take into account the self-regularizing effect observed during the training of large models. For instance, Neyshabur et al. 2018 measure a network's capicity by looking at the Frobenius norm of the difference of the hidden layer weights with the initialization, and empirically show that this measure decreases with increasing network size. This suggests that the bigger the network, the smaller the work the optimization algorithms needs to do, since the hidden layer weights are closer to their initialization. Furthermore one should keep in mind, when examining a model's capacity, that the notion of overparameterization itself is very vague, as it is intractable to measure a classification problem's complexity, and therefore determine the appropriate number of parameters. Experiments involving teacher-student models avoid this problem and are thus easier to analyze. The teacher is a model with a specific architecture and fixed weights, that given a random input produces an output. This generative process is repeated multiple times, and the input-output pairs are given to a student model as training data. The student is a model with a similar architecture, but a larger number of hidden units, and the goal for the student is to learn the teacher's model. Because the student can express much more complex functions than the teacher function it has to learn, it is said to be over-parameterized. Goldt et al. 2019 found evidence that, with a teacher-student setup, upon overparameterization Stochastic Gradient Descent finds solutions which amount to performing an effective model averaging, thus improving robustness and avoiding overfitting. This property seems to extend to more practical problems than the artificial teacher-student setup, there has been empirical evidence that supports that repeated units emerge when increasing a network's width, as a way to adjust the capacity of the model (Casper et al. 2019). Those repeated units have highly correlated outputs, implying that similar activation regions were learned. Additionally, removable units appear, which can be dropped out of the network without significantly hurting the validation accuracy. This kind of self-regularization enforces capacity constraints, as the function modeled by a network with

either of these units could me modeled by a simpler one.

2.2 Advantageous properties of overparameterized models

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" (Frankle and Carbin 2018), 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 (Brutzkus and Globerson 2019).

The Lottery Ticket Hypothesis

Contemporary experience suggests that overparametrised networks are easier to train, and achieve better generalisation. But such networks can be pruned, which sometimes heavily reduces the parameter-count (Han et al. 2015). One might ask why we do not train instead architectures discovered by pruning. It is commonly believed that these pruned networks are more difficult to train from scratch, and reach lower accuracy than the fine-tuned pruned networks (Li et al. 2016). The "Lottery ticket hypothesis" (Frankle and Carbin 2018) is a theory that provides an explanation as to why overparametrised networks might perform better, and has gained a lot of popularity recently. It states the following:

A randomly-initialized, dense neural network contains a sub-network 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.

According to the authors, at initialization, some weights that form a subnetwork "win the lottery", because the combination of their initial value and

the way they are aranged in that particular sub-network makes them particularly "trainable" by the chosen optimization algorithm, compared to other weights that are not in the sub-network. Those weights form what is called a "winning ticket", and the paper provides a way to identify the winning ticket after mutliple iterations of pruning, based on the weight magnitudes, and retraining, to compensate for the loss in accuracy caused by pruning. One of the important results of the paper is that when the weights of the winning ticket, which can be seen as a mask applied to the weights of the network, are reset to their initial value (all other weights set to zero), then the newly obtained network reaches similar if not better accuracy than the original network, up to a certain degree of pruning (up to 96%). Suprisingly, this result does not hold when the weights of the winning ticket are re-sampled, meaning that both the pruned architecture and the values of the unpruned weights are of importance. When randomly re-sampled, the winning tickets learn slower and reach lower accuracy than the re-initialized one.

Note that, as individual weights are pruned to identify the winning ticket, the theory makes use of unstructured pruning, and a winning ticket obtained by this procedure would have to rely on speciliased libraries and hardware to exploit benefits of the weight matrix sparsity. According to Liu et al. 2018, the results do not hold when using structured pruning, and might even be misleading in the case of unstructured pruning. The authors of the paper claim that the optimization algorithm (ADAM) and the small initial learning rate of the original paper lead to inferior accuracy when the winning ticket is randomly initialized, but this can be remedied using SGD and a higher learning rate. Moreover, as claimed in the original paper, the procedure to identify a winning ticket fails for deeper networks. Nonetheless, according to Frankle and Carbin, even if in some cases, up to a certain level of sparsity, highly overparametrised networks can be pruned, reinitialized and retrained successfully, beyond a certain point, when the network is extremely pruned, less severely overparametrised networks only maintain a good validation accuracy if they are well-initialised. Also, winning tickets have been found for deeper networks (Frankle, Dziugaite, et al. 2019) when, instead of resetting the weights of the winning ticket to their initial values, they are rewinded to their former values at iteration k, where k is much smaller than the total number of training iterations. Finally, winning tickets generate so much interest because they might reveal a lot about how to better initialise neural networks, but also help better understand the bias of modern

optimization algorithms towards sparse solutions. Surprisingly, it has been shown that, within the natural images domain, winning ticket initialisations generalise across a variety of datasets, often achieving performance close to that of winning tickets generated on the same dataset (Morcos et al. 2019). This suggests that winning tickets somehow have an inductive bias generic to neural networks, which generalises to multiple configurations.

Alignment of weight vectors during training

Brutzkus and Globerson 2019 define a simple 3-layer neural network for a binary classification task they call the XORD problem (XOR Detection). Its input is a vector with $2d$ entries, which take values in $\{-1, 1\}$, and the vector can be written as a sequence of d pairs. The goal is to determine whether any of the pairs has twice the same value (1 or -1). For this purpose a neural network is trained, with a convolutional, a max-pool and a fully connected layer. Only the weights of the convolutional layer, which are vectors in $\mathbb{R}^2$, are trainable. The authors prove that increasing the size of the convolutional layer results in better feature exploration, and that after convergence the weight vectors associated to the learned convolutions cluster around just a few directions. Furthermore, they show that when the size of the convolutional layer is small (just a few trainable weights), the network can achieve zero-training error on specific datasets, but not generalise well, because it is missing some patterns. This is less likely to happen with a larger convolutional layer because of better exploration of the feature space. According to the authors, this might explain why overparametrised neural networks don't overfit the training dataset and in some cases even generalise better than smaller networks. The former can be explained by weight clustering, which reduces the effective capacity of the network, and the latter by better exploration of the input space. Hence the observed generalisation properties of overparametrised networks do not contradict the findings of statistical learning, since training with gradient descent will facilitate the formation of clusters, thus reducing the network's expressive power. Empirically, the paper looks at a 3-layer neural network with a large convolutional layer, trained on MNIST. More specifically, after training, the vectors associated to the convolutional filters are clustered using K-Means with four clusters. The distribution of angles with respect to the closest cluster center is compared to that of clustered un-trained vectors (at initialization). The findings are in accordance with the XORD theory, as trained vectors tend to

be much closer to the center of the cluster they are part of.

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?

2.3 Leveraging model simplicity: Compression and Pruning

Structured vs unstructured pruning, suprisingly leads to better generalisation, pruning as a noise signal. Speed-up, memory foot-print. Need overparametrised 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.

3 Approach

This project focuses on classification tasks on natural images datasets, solved by neural networks. It takes a weight-vector perspective when reasoning about the effects of pruning and the dynamics of generalisation. This means that only structured pruning will be performed, at the level of neurons or filters. Those will also be the main entities studied when trying to understand the properties of neural network optimization. Why not look at unstructered pruning / individual weights?

The fact that the lottery ticket hypothesis might not really apply at the level of nodes / filters (Liu et al. 2018) could be due to multiple reasons:

- It is more difficult to identify winning tickets at the level of nodes or filters, i. e. iteratitive pruning based on the ℓ_2-norm fails to identify elements that are part of the winning ticket

- Structured pruning is to restrictive. Unstructured pruning can yield very complicated architectures that have at least one weight in every node or filter, even at high pruning ratios. This does not apply to structured pruning.

- Structured pruning is only efficient at low pruning rates, where the pruned architecture might be able to reach the same training loss when randomly initialised. Thus pruning can be done at the beginning, and there is no need to identify a winning ticket.

It would be interesting, from a theorectical point of view, to find out if, similarly to the well-initialised weights in the standard setting of the lottery ticket hypothesis, "trainable" nodes or filters exist. The hypothesis doesn't directly support this as, at a given layer, every node or filter should have the same expected number of weights which are in the winning ticket. But does that imply that, on average, at a given layer all nodes or filters are equally important and trained? In other words, are weights that are part of a winning ticket uniformly distributed across the nodes or filters of a layer, or do they concentrate on just a few of them?

Nonetheless, larger networks might still generalise better, and when we do not know how complex a classification task is or how big of a network we should use, then training an overparametrised network can be interesting, especially if we are able to quickly identify "trainable" nodes or filters, so that other parameters can be pruned and training made faster.

Assuming that the lottery ticket hypothesis is true, all nodes or filters of a layer in a neural network have the same probability of being part of the "trainable" subnetwork at initialisation, that is having some weights that are in the winning ticket. Also, at every layer, all nodes or filters have the same expected number of weights in the winning ticket.

Marginal distribution? Given that the node contains at least one weight that is part of the winning ticket, how does that affect the probability of other weights in the same node or filter of being part of the winning ticket?

3.1 Formal background

Formally define frame potential, generalisation 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

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

Casper, Stephen et al. (2019). "Removable and/or Repeated Units Emerge in Overparametrized Deep Neural Networks". In: *arXiv preprint arXiv:1912.04783*.

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

Frankle, Jonathan, Gintare Karolina Dziugaite, et al. (2019). "The Lottery Ticket Hypothesis at Scale". In: *arXiv preprint arXiv:1903.01611*.

Goldt, Sebastian et al. (2019). "Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup". In: *arXiv preprint arXiv:1906.08632*.

Han, Song et al. (2015). "Learning both weights and connections for efficient neural network". In: *Advances in neural information processing systems*, pp. 1135–1143.

Li, Hao et al. (2016). "Pruning filters for efficient convnets". In: *arXiv preprint arXiv:1608.08710*.

Liu, Zhuang et al. (2018). "Rethinking the value of network pruning". In: *arXiv preprint arXiv:1810.05270*.

Morcos, Ari S et al. (2019). "One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers". In: *arXiv preprint arXiv:1906.02773*.

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

Rolnick, David et al. (2017). "Deep learning is robust to massive label noise". In: *arXiv preprint arXiv:1705.10694*.

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

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