On the Algorithmic Hypothesis Class Complexity of Deep Neural Networks

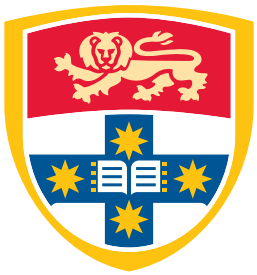
JEREMY GILLEN
SID: 510237022

Supervisor: Dr. Tongliang Liu

This thesis is submitted in partial fulfillment of
the requirements for the degree of
Bachelor of Advanced Studies (Honours)

School of Computer Science
The University of Sydney
Australia

23 November 2021



THE UNIVERSITY OF
SYDNEY

Student Plagiarism: Compliance Statement

I certify that:

I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);

This Work is substantially my own, and to the extent that any part of this Work is not my own I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: Jeremy Gillen

Signature: **Date**:

Abstract

Deep learning algorithms have empirically performed well on many different domains and applications, which is thanks to the large hypothesis class of functions they are capable of representing. However, the flexibility of the hypothesis class makes it difficult to obtain theoretical guarantees on generalisation to data outside the training set. Traditional generalisation bounds are based on the complexity of the entire predefined hypothesis space, but tighter bounds can sometimes be found by analysing a subset of the hypothesis space. We consider the specific case of a two-layer neural network and a previously studied algorithm, and find that the output hypothesis is very likely to lie in a low complexity subset of the predefined hypothesis space, which also contains the target hypothesis. We call the subset an Algorithmic Hypothesis Class (AHC). We prove that the algorithm has a small generalisation bound despite its large predefined hypothesis class, because it has a small AHC. Previous work has shown that shallow classifiers trained with gradient descent can have a small AHC, and this work extends the result to a deep learning setting. We expect that deeper networks and standard training algorithms may also have a low complexity AHC, and this perspective may be useful for obtaining new tighter generalisation bounds.

Acknowledgements

I'd like to thank my supervisor Dr. Tongliang Liu for his effort and enthusiasm supervising this project, and supporting and discussing my ideas when I went on a tangent. I'd also like to thank Adrian Goldwaser for proofreading and pointing out errors in my proofs.

CONTENTS

CHAPTER 1

Introduction

As deep learning becomes more powerful and more useful in many different domains, it becomes more important to understand its generalisation properties. A good understanding of these properties will lead to more reliable and trustworthy algorithms, and and give us a framework that will make it easier to improve neural network based learning algorithms, and better adapt them to new circumstances. Theoretical guarantees are closely linked to intuitive understanding, and vice versa. A good intuitive understanding of generalisation should soon lead to good generalisation guarantees, and a good generalisation guarantee should give us a rigourous understanding of generalisation. Therefore, we seek a both a theoretical and intuitive understanding of neural network generalisation.

Firstly, what is generalisation and why do we want to understand the generalisation of deep learning algorithms? Generalisation is the ability of a learning algorithms to perform well on test data different from the data it was trained on. Usually it is assumed that the test data is sampled from the same distribution as the training data, and using this assumption it is often possible to prove a guarantee that the test error be close to the training error. Usually this **generalisation bound** will be lower if more training data is available, indicating that the learning algorithm will provably be more reliable when given more training data.

Traditional statistical learning theory provides guarantees of generalisation by bounding the complexity of the hypothesis space of an algorithm. The hypothesis space of a learning algorithm is the set of hypotheses that the algorithm searches over when choosing a hypothesis that best fits the training data, and there are many ways to measure the complexity of such a set. **Complexity** in this context is best thought of as the flexibility of the hypothesis set, how well a hypothesis chosen from the set could fit random data. In statistical learning theory, low complexity hypothesis spaces lead to good generalisation bounds, intuitively because hypothesis classes without much flexibility won't be able to overfit to the peculiarities of the training set. This approach doesn't work for overparameterized neural networks,

because if they are capable of fitting random labels (Zhang et al., 2016), then this implies that the complexity of the hypothesis space must be close to maximal (capable of interpolating nearly any data labels). The topic of why overparameterised neural networks tend to generalise well despite such a complex hypothesis class has become a popular topic in the literature in the recent years.

The goal of this research project will be to investigate a new approach to theoretically bounding the generalisation error of neural networks. Current approaches are inadequate, because either the generalisation bounds produced aren't tight enough at realistic data sizes, or the bounds don't reflect empirical observations of generalisation properties, and therefore don't help us to deeply understand neural network generalisation.

Specifically, this project investigates whether the idea of an **Algorithmic Hypothesis Class** (AHC) can be applied to non-linear algorithms like neural networks, and whether generalisation bounds can be derived using this method. An AHC a is a subset of the full hypothesis space with a lower complexity than the full hypothesis space. It is defined such that, conditional on the training data, the learning algorithm is very likely to output a hypothesis that is a member of the AHC. To make the AHC useful for proving generalisation bounds, we also need to define the subset in such a way that it contains the target or expected hypothesis (the hypothesis we would find given infinite data). In order to define such a hypothesis space, assumptions may need to be made about the data distribution, the optimisation procedure, initialisation conditions, and target hypothesis.

If this approach can be scaled to large neural networks, it would give us a simple and intuitive theory of neural network generalisation, which could be easily understood by researchers who don't usually work on generalisation theory. The main benefit of the approach is its similarity to traditional Statistical Learning Theory, in that it still involves measuring the complexity of a hypothesis class, with low complexity hypothesis spaces leading to better generalisation. With a better intuitive and mathmatical model of generalisation, researchers could investigate ways of increasing and decreasing the simplicity bias of neural networks, or adjusting it to better fit different domains. In contrast to the traditional methods, the Algorithmic Hypothesis Class approach is more suited to highly complex algorithms capable of memorising data, since it avoids having to upper bound the complexity of the entire hypothesis class.

There are a couple of reasons to think that generalisation bounds from AHC have some promise for producing good generalisation bounds for overparametersized networks. Neural networks appear to have implicit biases which may cause the output function to tend toward simpler functions (De Palma

et al., 2019). This idea is popular and explored further in the literature review. On top of this, under certain conditions neural networks can be proven to converge to a specific interpolating function between data points (Jacot et al., 2018), which suggests that the generalisation properties of networks under these conditions entirely come from the similarity between the interpolating function and the target function. With appropriate assumptions about the target function which guarantee its similarity to the interpolating function, tight bounds could be proven by showing that the AHC is a small set of functions similar to the interpolating function.

The main question this research will attempt to investigate is which assumptions are necessary in order to create a generalisation bound using the Algorithmic Hypothesis Class method. The approach this research will take will strive to be similar to the approach used in (Liu et al., 2017), in that it will be to seek out the necessary assumptions required to define a suitable subset of the hypothesis space, then prove that the optimisation algorithm output falls into that space with high probability. The final step will be calculating the Rademacher Complexity of the hypothesis space found.

Some common theoretical guarantees are optimization convergence guarantees, function recovery guarantees and generalization guarantees. To clarify terminology, we should first note the differences between each bound. Optimization convergence guarantees provide a rate at which the algorithm will find a local or global optimum. For example, Du et al. (2019) show that sufficiently wide neural networks will provably converge at a linear rate to the global optimum, under certain circumstances. Recovery guarantees are different, and have so far only been proved for small networks under strong assumptions. They work by assuming that there is one specific parameterisation that corresponds to the target function, and provides a guarantee that the network will converge to that parameterisation or a functionally equivalent parameterisation. For example, assuming Gaussian input data, Bakshi et al. (2019) find that two-layer networks can converge to the target parameters in polynomial time. In contrast, generalization guarantees show that the loss on the training set is similar to the loss on the test set, which isn't necessarily the same as converging to the target parameters.

In this research, we combine a recovery guarantee for two-layer neural networks (Zhong et al., 2017) with the AHC technique (Liu et al., 2017) to find a generalization bound on two-layer neural networks. The algorithm found by Zhong et al. (2017) trains a two-layer neural network with high dimensional input, and comes with a recovery guarantee on the rate of convergence to the target function. Using this recovery guarantee, we define the AHC as a ball in parameter space around the target parameters, whose

diameter decreases exponentially with the size of the dataset. We prove that this AHC has low complexity, and prove two generalization bounds that follow directly from the small size and low complexity of the AHC.

Chapter 2 will review the literature surrounding the generalisation problem, and the research related to recovery bounds on two layer neural networks. Chapter 3 will introduce some notation and basic theorems in preparation for the mathmatical sections. Chapter 4 will provide a much more in depth review of one paper in particular, whose results are relied upon in the following chapters. Chapter 5 will present our contributions in the form of two theorems and proofs of those theorems. Chapter 5 will discuss the implications of these results and potential avenues of research for further exploring the AHC of deep neural networks.

CHAPTER 2

Literature Review

Learning Theory is a field that tries to understand when and why learning algorithms work. The field seeks to answer questions like "How much data will we need to train this learning algorithm?", "How well can we expect this learning algorithm to perform in the real world?", and "What assumptions are we implicitly making when we use this algorithm?" (Vapnik, 1999). Often, these problems are approached by attempting to prove that a learning algorithm will work well, perhaps given some assumptions about the true source of the data. We measure how well an algorithm works in several ways, but one of the main ones is called **Generalisation Error**, which measures how well the algorithm is expected to perform on a test set of data. Often this is compared to how well the algorithm performs on the training set compared to how it performed on the training set. Using mathematical descriptions of learning algorithms, we can describe precisely what it means to "overfit" and "underfit", analyse the "inductive biases" of algorithms and carefully describe situations where an algorithm will and won't be able to learn effectively. By properly and rigorously understanding the properties and behaviour of learning algorithms, we are better equipped to develop learning algorithms with improved performance and create algorithms to solve new problems.

As neural networks have become more powerful and popular learning algorithms, the community studying the theory of neural networks has grown considerably. Much of the community focuses on studying questions related to the generalisation error of neural networks. This is done both empirically or theoretically. Theoretical studies usually attempt to prove an upper bound on the generalisation error. These upper bounds are naturally very loose, since they must still hold in the worst case scenarios of extreme overfitting. One motivation for finding new bounds and improving existing bounds is that a sufficiently tight generalisation bound will give us a rigorous understanding of neural network generalisation and help to "explain" the unexpectedly good performance of some neural networks. A good bound will help us validate some intuitive ideas about when and why neural networks generalise well, and dismiss

others. The connection between theory and intuitive understanding goes both ways, since a sufficiently rigorous understanding of neural network generalisation should naturally lead to a generalisation bound.

In the following paragraphs I will give an overview of the following topics: the classic learning theory view of generalisation, the recent unresolved puzzles relating to neural network generalisation, and the empirical evidence that might suggest research directions for resolving these puzzles. I will then describe some research relating to the loss landscape of neural networks and how it ties in to generalisation, and a few different lines of research that might lead to a better understanding of generalisation: one of which analyses learning using information theory, and another which connects neural networks to kernel learning. We will then look at an approach that has been used to find tighter bounds on linear algorithms and kernel algorithms by focusing on a subset of the hypothesis class, which may be able to be scaled up to non-linear algorithms and provide another approach to understanding neural network generalisation.

Classic statistical learning theory provides us with a mathematical language to describe learning algorithms, and a general theory that describes why learning algorithms can generalise. A learning algorithm selects a hypothesis $\hat{h}$ from a predefined class of hypotheses $\mathcal{H}$, by finding the hypothesis that minimises some loss function on the training data. Each hypothesis is a function that maps a datapoint x to a label y. Statistical learning theory relates generalisation to the "capacity" of the hypothesis class, which involves a tradeoff between how complex (i.e. flexible) a hypothesis class is and how well the algorithm will generalise. Good generalisation is defined by how close the test error is to the training error. This same tradeoff is often called the "Bias-Variance tradeoff". The complexity of the hypothesis class is often measured using quantities like VC dimension, Rademacher complexity or Covering Number (Shalev-Shwartz and Ben-David, 2014). Upper bounds for these quantities exist for neural networks, but they grow very fast as the size of the neural network grows. With the very large neural networks that are common today, these measures only provide vacuous bounds on the generalisation error, which means that for common dataset sizes the generalisation bound doesn't go below 1 and therefore don't provide any guarantee of generalisation.

A lot of research in the past few years has been motivated by the observation that there are several poorly understood problems related to neural network generalisation. The main puzzle is the one mentioned above, that traditional generalisation bounds don't tell us anything useful about the generalisation of large neural networks. This puzzle became popular after the publication of (Zhang et al., 2016), which showed empirically that large image models were able to fit a randomly labeled dataset. This property, being flexible enough to fit random labels while still generalising well when trained on the correct

labels, completely contradicts what would be predicted by a classic understanding of generalisation. One interesting observation and clue noted in the paper is that training to convergence with random labels tends to be much slower than training to convergence with true labels. Another related puzzle is the phenomenon of "Deep Double Descent" (Belkin et al., 2019; Nakkiran et al., 2019), where the test error is observed to first decrease as the size of the neural network increases, then increase, then decrease again! The first decrease and increase matches the classic bias-variance tradeoff, but that theory doesn't predict the improvement that follows as the size of the model is increased even further. Further contributions point out that both phenomena can be seen in some other circumstances, for example a Gaussian Process model can be flexible enough to fit random labels, but still generalise well. Double descent can also be observed in linear models trained with SGD (Bartlett et al., 2020). Both of these puzzles motivate the search for an improved theory of generalisation.

Several studies attempt to empirically investigate the circumstances where large neural networks generalise. One study (Jiang et al., 2019) investigated 40 complexity measures and studied the correlation between those measures and the empirical test error. The findings suggested norm based measures of complexity didn't work well, but quantities related to the sharpness of the loss landscape around the output hypothesis were well correlated with test error. Another study finds that sensitivity to input perturbation is another measure that correlates well with test error, and suggests that stochastic gradient descent might be increasing the probability of finding low-sensitivity hypotheses (Novak et al., 2018). An alternative perspective suggests that low complexity hypotheses are much more common in weight space, and therefore they are more likely to be found (Valle-Perez et al., 2018; De Palma et al., 2019). This perspective suggests that it is the structure of neural networks that gives them an inductive bias toward simpler functions, and simpler functions are much more likely to generalise well. When they use the term "simple function", these papers mean that the functions have low Kolmogorov complexity, which means that the functions can be implemented by a short program on a Turing machine. It's clear that low complexity functions should generalise better than high complexity functions for theoretical reasons. The justification is intuitively similar the the reason that small hypothesis classes have good generalisation, namely, there are much fewer simple functions than complex functions. Another paper (Kalimeris et al., 2019) suggests that this tendency towards finding simple functions may explain the phenomenon of random labels taking longer to fit, by showing empirically that when training with SGD, simple functions are learned first and gradually become more complex as training progresses. (Shah et al., 2020) also finds empirically that neural networks trained with SGD have a strong bias towards simple functions, but points out that there are some situations where this bias may be disadvantageous.

One investigation (Goldblum et al., 2019) showed that generalisation could empirically be improved by adding a regulariser that encouraged the weight norm to be higher (!), and the same research group finds that many different optimizers, including some that operate on very different principles to gradient descent, all tend to find neural networks that generalise well (Huang et al., 2020). All of these lines of empirical research are very useful for informing theoretical research, and giving us a better idea of which pathways to pursue and which are not promising. The biggest takeaway from these papers is that norm based hypothesis complexity measures, despite their intuitive appeal, are completely insufficient for explaining generalisation.

The simplest generalisation bounds are based on intuitive measures of the complexity of neural network functions, like the number of parameters, the norm of the parameters or the regularisation conditions imposed on the network during training. We introduce this more traditional approach to provide a contrast and context for the other approaches described later. The VC dimension, which was the first measure of hypothesis space complexity, can be upper bounded for neural networks and found to be a linear function of both the depth and the number of parameters (Harvey et al., 2017). Generalisation bounds follow directly from bounds on the VC dimension. The next most straightforward approach is to use the norm of the parameters. The Rademacher complexity of neural network function classes (another complexity measure) has been bounded using several different norms of the weight matrices (Neyshabur et al., 2015; Golowich et al., 2019). This type of bound is beginning to be independent of the huge parameter count, but is still related. When we introduce regularisation, the link between parameter count and bound can be weakened such that the error bound is only a logarithmic function of the parameter count (Taheri et al., 2021).

Some research focuses on eliminating possible avenues that we might consider investigating. One line of research argues that the Uniform Convergence PAC framework usually used for describing generalisation bounds isn't capable of producing tight generalisation bounds for neural networks (Nagarajan and Kolter, 2019). Their main argument is based on an empirical observation that parameter norms increase with training data, which implies that bounds that use those norms don't predict the observed behaviour of neural networks. However, a more recent paper (Negrea et al., 2020) suggests that there are ways of analysing generalisation that work around the problems brought up by Nagarajan and Kolter (2019), which still use the framework involving uniform convergence. Both works are focused on the problems caused by the high sensitivity of the output hypothesis to changes in the datapoint location, which make

it difficult to prove generalisation bounds for interpolating classifiers like overparameterised neural networks. Another line of investigation is to work out what sorts of problems can't be learned efficiently. One paper (Shamir, 2016) proves that in order to learn a neural network in polynomial time, assumptions must be made about the class of target functions and the class of possible input distributions. They prove this by providing examples of fairly natural target functions that are very difficult to learn under some data distributions, and data distributions that are difficult to learn under any target function. This result gives some idea of what assumptions will have to be made to create generalisation bounds for neural networks, and something to watch out for: If your hypothesis space contains their counter examples then perhaps a mistake has been made. It also highlights the importance of assumptions and prior knowledge about the learning problem, which is a similar lesson to that of no-free-lunch theorems (Wolpert, 1996).

Analysis of the loss landscape of neural networks provide a rich source of ideas for understanding neural network optimisation and generalisation. The density of diverse functions in small areas of parameter space presents a problem that needs to be overcome when creating generalisation bounds, as it is closely related to the extreme flexibility of neural networks. A line of research investigates this property for wide neural networks (Amari, 2020; Jacot et al., 2018). As mentioned earlier, the shape of the landscape around the output hypothesis has been linked to generalisation, and the properties of the loss landscape are often also used to understand how neural networks converge. One beautiful series of investigations (Huang et al., 2020) found many minima that all perfectly fitted the training data using normal training methods, then found many minima that perfectly fit the training data but generalise poorly, and tried to visualise the loss landscape around both types of minima. They found that the flatness and wideness of minima was strongly correlated with generalisation, and suggested that the reason there is a bias toward finding these minima is their massive volume in comparison to minima that generalise less well. They found that for one dataset, the volume of the basin around minima that generalise well was *at least* 10,000 orders of magnitude larger than the basins that generalise poorly. This provides a convincing intuitive understanding of the inductive bias of neural network training procedures. One potential justification for the connection between the flatness of the loss minima and generalisation is on Minimum Description Length grounds. A neural network that has a wide flat minima can use fewer bits to store the parameterisation, and is therefore simpler than minima that requires high precision to describe (Hochreiter and Schmidhuber, 1997). This is similar to saying that flat minima are robust to perturbation, and also similar to saying that flat minima separate the data with a wide margin, where possible (Huang et al., 2020). There has been some disagreement about the relationship between sharpness and generalisation, with a study (Dinh et al., 2017) pointing out that through reparameterisation we can create equivalent

networks with arbitrarily sharp minima. This indicates that flatness isn't directly equivalent to good generalisation, however it doesn't at all rule out the hypothesis that finding flat minima is an implicit bias in gradient descent, which may be a major part of the explanation for the mystery of neural network generalisation.

One line of research investigates the connection between the batch size in stochastic gradient descent (SGD) and the shape of the minima around the resulting hypothesis (Keskar et al., 2017). Recent work has theoretically characterised this link by showing that the optimisation path taken through the loss landscape under SGD can be described using a modified loss function, which adds a specific regularisation term (Smith et al., 2021). The regularisation term minimises the sum of the norm of the gradient of the loss with respect to each batch. This modified loss is somewhat difficult to interpret, since on large interpolating models it shares the same global minima as the original loss. The paper speculates that the this loss improves generalisation by both penalising sharp regions of the loss, and 'non-uniform' regions where the minibatch gradients don't all point in the same direction.

One approach to resolving the problems related to generalisation in large neural networks is to look for techniques that were useful for bounding simpler algorithms, and try to extend them to the neural network setting. One approach that has been used to bound the generalisation of linear algorithms is presented in (Liu et al., 2017). This approach focuses on one particular portion of the hypothesis space, and tries to bound its complexity. At the same time, it requires some kind of proof that the output hypothesis of the learning algorithm falls into that subset of the hypothesis space with high probability. Together these conditions allow us to use well known theorems bounding generalisation error in terms of hypothesis space complexity (usually Rademacher complexity). A more formal description follows. In the following we will view a randomised learning algorithm as an algorithm that when given a dataset S, outputs a hypothesis $\hat{h}$ sampled from a distribution $Q(h|S)$, which we will call the posterior distribution. We can define the **Algorithmic Hypothesis Class** (AHC) as a set of hypotheses that has mass at least $1 - \delta$ with respect to $Q(h|S)$. This is similar to the definition of the credible region in Bayesian statistics. There are clearly many possible choices of such sets, but we can make the AHC useful for proving generalisation bounds if we can show that such a set exists, includes the optimal hypothesis h^*, and is of limited complexity. Note that this definition depends on the dataset S, however under some circumstances we may be able to show that the complexity of the AHC is limited for any dataset of size n, and our generalisation bound can become data independent. In (Liu et al., 2017), the property of algorithmic stability is used to help define the relevant subset of the hypothesis space, and results in

a generalisation bound for stable linear algorithms. A stable algorithm is one that is robust to resampling one sample of data in the dataset, and is sometimes a property of algorithms learned with gradient descent. This kind of stability may not be present in large neural networks, which provides reason to doubt that the method can be extended. However, one reason to suspect that this approach might extend to large neural networks is that as we have seen in the above paragraphs, there is reason to think that gradient descent selects some minima in the loss landscape with much higher probability than others. If we were able to understand this well enough to define an AHC that contains only those minima that were likely to be selected, then this may allow us to prove very tight generalisation bounds. Another reason to be hopeful is that it has been shown that neural networks often learn very similar representations despite different initialisation points, architectures and data (Kornblith et al., 2019; Olah et al., 2020). This suggests similar computations are being done, which suggests that only a subset of the hypothesis space is being used.

A similar idea to the Algorithmic Hypothesis Class is Local Rademacher complexity (Bartlett et al., 2005), which uses the Rademacher complexity computed on a small subset of the hypothesis space with low empirical risk. It is applied to kernel algorithms in this paper. Both ideas consider the complexity of a subset of the hypothesis class and use it to find tighter generalisation bounds, but the Local Rademacher complexity bounds have yet to be extended to deep learning algorithms. Each approach is different in the way it defines the particular subset to focus on. The Algorithmic Hypothesis Class is (in the original paper) defined as a region around the expected hypothesis, which may include hypotheses that correspond to a high empirical risk.

Another interesting approach focuses on guarantees for specific small neural network architectures. One particular type of guarantee that is strongly related to a generalisation guarantee is a function recovery guarantee. This type of guarantee requires the assumption of a specific target function, and gives a rate at which the parameters of a network converge to those of the target function. Several papers present similar recovery bounds (Zhong et al., 2017; Zhang et al., 2019; Soltanolkotabi et al., 2019), all on two layer neural networks. These types of results may provide useful results for extending the Algorithmic Hypothesis Class framework to two layer neural networks, although results in current research don't extend to the large neural networks common today. The bounds from (Zhong et al., 2017) are worth describing in greater detail, to fully introduce these useful results. Several important assumptions and algorithmic changes are required in order to make this guarantee, which creates a large difference between the setting used for their proof and the real world of neural network use. This does reduce the

usefulness of the setting for understanding real world neural networks, but enough similarities remain that it may be helpful for guiding future research directions. The algorithm introduced in the paper first uses a small portion of the dataset to exactly recover the second layer parameters and get an approximation of the first layer parameters. The portion of the dataset devoted to this initialisation is constant in n, where n is the size of the training sample. Next the algorithm uses gradient descent from the initial estimate to the target estimate. Lower and upper bounds on the convexity of the region around the target parameters are derived, which can be combined with a bound on the distance from the true gradient to the estimated gradient to show that each step of gradient descent moves the parameter estimate at least a constant proportion closer to the target parameters, leading to linear convergence in n. This requires using new samples with every gradient step, similar to minibatch gradient descent, but without ever reusing samples. With this algorithm, it is proved that the estimate of the true parameters will converge with error ϵ toward a paramerisation that corresponds to the target function. The error ϵ is bounded above by $\epsilon = \exp\left(\frac{-\sqrt{n}}{C}\right)$ where C is a constant that depends on the precise target hypothesis. In a similar vein that supports this view of the neural network loss landscape, (Li and Yuan, 2017) analyze the convergence of a different two-layer architecture and find that it has an initial period of non-convex optimisation during training, followed by reaching a convex region and converging to a global minimum.

In the following two paragraphs we will look at a couple of newer approaches, which have made some progress toward creating frameworks for understanding deep learning. (Xu and Raginsky, 2017) and (Bu et al., 2020) build on a line of research that uses an estimate of the mutual information between the dataset and output hypothesis to bound the generalisation error of neural networks. This approach has advantages over most approaches for bounding the complexity of the hypothesis class, since it takes advantage of information about the data distribution and training algorithm to derive a tighter generalization bound. The result is a non-trivial generalisation bound that can be estimated empirically, but in order to get an empirical estimate the of bound, multiple datasets must be used for training the network. While this approach may give us useful generalisation bounds, their usefulness for deeply understanding deep learning may be more limited.

Another recent topic which has sparked many papers is research related to Neural Tangent Kernels (Jacot et al., 2018). This is an approach to understanding neural networks that analyses what happens to neural network initialisation and training as the width of each layer approaches infinity. It turns out that in the infinite width limit, many things become easier to analyse. As width increases, neural network training becomes closer and closer to being an infinitely small linear step from the initialisation. One

of the tools that this approach gives us is the kernel regression perspective on neural networks, which says that a fully trained infinite width network will converge to the minimum norm kernel regression solution, where the norm is measured using the Neural Tangent Kernel (NTK) of the architecture being used. Another advantage of this perspective is that we can represent the neural network hypothesis space using a Reproducing Kernel Hilbert space, which is a function space that has particularly useful properties (in particular, continuity as you move around the function space. E.g. Nearby points in function space always correspond to similar functions). The NTK approach allows techniques originally used for kernel methods to be applied to neural networks, which gives us a whole new window into understanding them. On the other hand, the generalisation properties of interpolating kernel methods are also not fully understood (Belkin et al., 2018). There is some worry that the NTK approximation can't fully explain the generalisation properties observed, because it isn't capable of explaining the benefit of representation learning, and most experiments show that kernel regression using a derived NTK doesn't quite match the empirical generalisation error of the neural network. Recent work has attempted to address these issues, with some success (Li et al., 2019; Roberts et al., 2021).

One final perspective on the problem that is important to include is the Bayesian probability theory approach to understanding generalisation. The Bayesian approach to inference defines a prior probability distribution over the predefined hypothesis class. This distribution indicates how likely each hypothesis is *a priori* i.e. before seeing any of the data. This distribution can encode information such as "the hypothesis is more likely to be a simple function", "it is more likely to be a function with certain symmetries", or simply "expert experience has indicated that certain hypotheses are more likely". This perspective can be helpful for understanding the implicit biases present in non-bayesian algorithms, which do not explicitly follow the rules of probability theory but are still essentially doing the same thing by combining some data with some prior information to produce a hypothesis. Some commentary (Wilson and Izmailov, 2020) shows that this perspective can be useful for understanding neural network generalisation, and that the puzzles of generalisation look less confusing from a Bayesian perspective. The paper shows that the double descent phenomenon goes away when averaging over a posterior distribution of models (which is the standard Bayesian approach), and that complex Bayesian model classes like a Gaussian Process can replicate the overfitting behaviour of overparameterised neural networks, while still generalising well. The generalisation comes from a good prior distribution over functions. It is argued that deep neural networks (especially convolutional networks) implicitly contain a good prior distribution over functions. This implicit bias *allows* the model to fit random labels, but when possible,

chooses the most prefered/simple/realistic function to fit to real data. Viewing neural network generalisation from this perspective may help guide research on generalisation, by reframing the research as characterising the implicit prior knowledge contained in neural networks.

It's difficult to extract general lessons from the dizzying variety of research on neural network generalisation, but some conclusions can be made from the current state of research. Norm and parameter based bounds on generalisation are not enough, and will be replaced by new paradigms. Evidence suggests that some hypotheses are implicitly prioritised over others in neural network training, and this bias is what explains neural network generalisation. Several approaches are looking promising and may lead to a unified theory of generalisation in the coming years, and all of these approaches need to be further developed to fully and rigorously explain the generalisation puzzles that neural networks have presented.

CHAPTER 3

Preliminaries

3.1 Generalisation

A learning algorithm selects a hypothesis from the hypothesis space $\mathcal{H}$. If we have a learning algorithm trained using the dataset S of pairs (x, y) drawn from some distribution $\mathcal{D}$, we are interested in the **generalisation error** of this algorithm:

$$\sup_{h \in \mathcal{H}} L_{\mathcal{D}}(h) - L_S(h) = \sup_{h \in \mathcal{H}} \mathbb{E}_{x,y \sim \mathcal{D}} l(h(x), y) - \sum_{x,y \sim S} l(h(x), y),$$

where l is the loss function, and h is a hypothesis in the hypothesis class $\mathcal{H}$. This can be thought of as the maximum possible difference between the training error and the expected test error, so a tight upper bound on this value will give us useful information about the real world performance of the algorithm. Later we will also consider the generalisation error for a subset B of the full hypothesis class $\mathcal{H}$. We can bound the generalisation error with the following theorem (Shalev-Shwartz and Ben-David, 2014).

THEOREM 1. *We assume $|l(h(x), y)| \leq C$ for any x, y and $h \in \mathcal{H}$. Then for any $h \in \mathcal{H}$, with probability $1 - \delta$:*

$$L_{\mathcal{D}}(h) - L_S(h) \leq 2\mathfrak{R}(l \circ \mathcal{H}) + 4C\sqrt{\frac{2\ln(4/\delta)}{n}}.$$

In Theorem 1 we see that the generalisation error is bounded in terms of the **Rademacher Complexity** ($\mathfrak{R}$) of the loss function composed with the hypothesis class, i.e. $\mathfrak{R}(l \circ H)$.

DEFINITION. *Let σ be a list of independent and identically distributed random variables each uniformly drawn from $\{-1, 1\}$. Then the empirical **Rademacher complexity** of a function class $\mathcal{H}$ is:*

$$\mathfrak{R}(\mathcal{H}) = \mathbb{E}_\sigma \sup_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i h(x_i).$$

Rademacher complexity is a measure of the flexibility of a function class. In the case of classification problems, Rademacher complexity can be thought of as a measure of the ability of the hypothesis class to learn randomly labelled data. In the case of regression problems, it could be viewed as the maximum correlation of a hypothesis with a random function whose output is chosen from $\{-1, 1\}$. Hypothesis classes with high Rademacher complexity are closer to interpolating algorithms, and classes with low Rademacher complexity have a stronger inductive bias and are less susceptible to overfitting.

3.2 Setting and Notation

The learning setting assumed in this paper is the problem of regression with a two-layer neural network whose architecture is defined below. The input data is assumed to be normally distributed d-dimensional data $x \sim \mathcal{N}(0, I_d)$, and we assume that the problem is realizable (there exists a network with zero error on any test data). This implies there is zero label noise, since the labels are a deterministic function of the input data. The network structure is defined by

$$y = h(x) = \sum_{i=1}^{k} v_i \phi(w_i^\top x),$$

where x is an input vector, w_i and v_i are parameters, and ϕ is an activation function that satisfies a few common assumptions, detailed in section 4. The variable x and parameter w_i are d-dimensional vectors, v_i is a scalar. The number of hidden units is k, and this is assumed to be less than d. We define $W = [w_1^\top ... w_k^\top]^\top$, the stack of first layer weight vectors, and $V = [v_1 ... v_k]^\top$, the vector of second layer parameters. We call the target parameters V^* and W^* (or v_i^* and w_i^*), so the data distribution $(x, y) \sim \mathcal{D}$ is defined by:

$$x \sim \mathcal{N}(0, I_d)$$

$$y \sim \sum_{i=1}^{k} v_i^* \phi(w_i^{*\top} x).$$

CHAPTER 4

Summary of Recovery Guarantees

This section summarises in detail the results proved by Zhong et al. (2017), which were introduced in the literature review and will be used in Chapter 5 to prove a generalisation bound. Zhong et al. (2017) present an algorithm that guarantees the recovery of the target parameters with the error linearly converging with the size of the training set. The algorithm is based on a proof that the loss function is locally convex near the target parameters, and once in this region, convergence happens very fast.

The algorithm first uses a small portion of the dataset to exactly recover the second layer parameters and get an estimate of the first layer parameters. The portion of the dataset devoted to this initialisation is constant in n, where n is the size of the training sample. Next the algorithm uses gradient descent from the initial estimate to the target estimate. Lower and upper bounds on the convexity of the region around the target parameters are derived, which can be combined with a bound on the distance from the true gradient to the estimated gradient to show that each step of gradient descent moves the parameter estimate at least a constant proportion closer to the target parameters, leading to linear convergence in n. This requires using new samples with every gradient step, similar to minibatch gradient descent, but without ever reusing samples.

A new dataset S is sampled T times, which allows a guarantee that with high probability each of the T updates moved the parameters closer to the target parameters. Then the total dataset size is $T \times |S|$, where $|S|$ is larger than a constant that depends on the size of the network and properties of the target parameters. As $T \times |S|$ increases, T can increase without increasing $|S|$, which allows linear convergence since each update moves the estimate a multiplicative constant closer to the target parameters.

The key step that makes this algorithm easier to analyse is the initialisation step, which exactly recovers the true second layer parameters with high probability. The initialisation step involves separately estimating the signs and lengths of parameters, by estimating a tensor that can be subsequently factorised into these values. The second layer parameters are assumed to be in $\{-1, 1\}$ without loss of generality,

since for any v_i there exists a constant c such that $v_j\phi(w_j^\top x) = \pm\phi(cw_j^\top x)$. This follows from the assumed properties of the activation function ϕ, described in the next paragraph.

The activation function ϕ must satisfy three properties, which are described in detail in the paper (on p 7) (Zhong et al., 2017). The first property bounds the first derivative homogeneously, the second property is a complex criterion that can be interpreted as enforcing non-linearity, and the third property bounds the second derivative.

PROPERTY 1. $0 \leq \phi'(z) \leq L_1|z|^p$ *for some* $L_1 > 0$ *and* $p \geq 0$.

PROPERTY 2. *Let* $\alpha_q(\sigma) = \mathbb{E}_{z\sim\mathcal{N}(0,1)}[\phi'(\sigma z)z^q]$, *and let* $\beta_q(\sigma) = \mathbb{E}_{z\sim\mathcal{N}(0,1)}[(\phi'(\sigma z))^2 z^q]$. *Then for all* $\sigma > 0$,

$$\min\{\beta_0(\sigma) - \alpha_0^2(\sigma) - \alpha_1^2(\sigma), \beta_2(\sigma) - \alpha_1^2 - \alpha_2^2(\sigma), \alpha_0(\sigma)\alpha_2(\sigma) - \alpha_1^2(\sigma)\} > 0.$$

PROPERTY 3. $|\phi''(z)| \leq L_2$ *for any* z, *for some* $L_2 > 0$.

We will add one extra constraint not found in (Zhong et al., 2017), which will be that the activation function is K-Lipschitz continuous

PROPERTY 4. $|\phi(z_1) - \phi(z_2)| \leq K|z_1 - z_2|$ *for any* z_1 *and* z_2.

Many regularly used activation functions fit these properties, including sigmoid, tanh and smooth approximations to ReLU, such as softplus, or ELU.

The following theorem (Theorem 6.1 in (Zhong et al., 2017)) is a convergence guarantee for the above algorithm.

THEOREM 2. *We assume that the sample size* $|S|$ *is greater than* $d\log(1/\epsilon) \cdot$ *poly*$(\log d, t, k, \lambda)$ *and the number of gradient descent iterations* T *is greater than* $\log(1/\epsilon) \cdot$ *poly*$(k, \nu, \lambda, \sigma_1^{2p}/\rho)$. *We also assume that* ϕ *satisfies properties 1,2 and 3 and the data is drawn from* $\mathcal{D}$, *then for any* $t \geq 1$, *with probability at least* $1 - d^{-\Omega(t)}$:

$$\|\hat{W} - W^*\|_F \leq \epsilon\|W^*\|_F, \text{ and } \hat{V} = V^*.$$

Keep in mind that since each step of gradient descent requires a fresh sample, the total number of samples required is $T|S|$. We will call the total sample size required $n = |S|T$. Setting $n = d\log(1/\epsilon)^2 \cdot$

$\text{poly}(\log d, t, k, \lambda, \nu, \sigma_1^{2p}/\rho)$, we can rearrange to obtain the error $\epsilon = \exp\left(\frac{-\sqrt{n}}{\sqrt{d\text{poly}(\log d, t, k, \lambda, \nu, \sigma_1^{2p}/\rho)}}\right)$, which converges to zero very rapidly as n increases.

The as yet undefined parameters above are as follows: t is a parameter that allows a tradeoff between probability of success and sample size, d and k are the input and hidden sizes, and λ is a condition number of the target weight matrix (a function of the singular values). The variable ν is a property of V^*, namely the maximum divided by the minimum element of that vector. The variable σ_1 is the minimum singular value of W^*. The variable ρ is a value that comes from property 2 of ϕ, and p is the degree of homogeneity from property 1.

CHAPTER 5

Generalization Bounds

5.1 Bounding the Rademacher Complexity of the AHC

We will bound the generalization error of the algorithm by first bounding the Rademacher complexity of the Algorithmic Hypothesis Class. First we will prove a lemma upper bounding the value of a term that depends on the Gaussian input data.

LEMMA 1. *If for all $i \in \{1, ..., n\}$ we have x_i drawn from a d-dimension random normal distribution $\mathcal{N}(0, I_d)$, and σ_i sampled from the Rademacher distribution, then we have with probability $1 - \delta$:*

$$\begin{aligned}\sum_{i=1}^{n} \sigma_i \|x_i\| &< \sqrt{n}\sqrt{nd + 2\log(1/\delta) + 2\sqrt{nd\log(1/\delta)}} \\ &< n\sqrt{5d\log(1/\delta)}\end{aligned}$$

PROOF. First we can use the Cauchy-Schwarz inequality to get the bound:

$$\begin{aligned}\sum_{i=1}^{n} \sigma_i \|x_i\| &\leq \sqrt{n}\sqrt{\sum_{i=1}^{n} \|x_i\|^2} \\ &= \sqrt{n}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{d} x_{ij}^2}\end{aligned}$$

Since each x_{ij} is sampled from $\mathcal{N}(0, 1)$, the sum $\sum_{i=1}^{n}\sum_{j=1}^{d} x_{ij}^2$ is drawn from a χ^2 distribution with nd degrees of freedom. We can use a bound implied by Lemma 1 of (Laurent and Massart, 2000), which gives the following bound on the tail probability of χ^2-distributions. If U is sampled from a χ^2 distribution with k degrees of freedom, then we have:

$$\mathbb{P}(U \geq 2\sqrt{k\ln(1/\delta)} + 2\ln(1/\delta) + k) = \delta.$$

Lemma 1 follows by combining this bound with the above inequality. □

Now we are ready to define and bound the AHC of a specific neural network. The neural network function is parameterised by a matrix W of first layer weights and vector V of second layer weights. We can define an Algorithmic Hypothesis Class $B = \{(W, V) \in \mathbb{R}^{d\times k} \times \mathbb{R}^k : V = V^*, \|W - W^*\| \leq \exp(\frac{-\sqrt{n}}{\sqrt{d\text{poly}(\log d,t,k,\lambda,\nu,\sigma_1^{2p}/\rho)}})\|W^*\|\}$. This set is defined such that the algorithm defined in Zhong et. al (2017) will output a hypothesis in this class with probability $1 - d^{-\Omega(t)}$ (as per Theorem 2). See Chapter 4 for the definitions of the parameters of the polynomial.

An algorithmic hypothesis class must be constructed in such a way that its complexity can be bounded. The only additional assumption required on top of the assumptions of Zhong et al. (2017) is that the activation function ϕ is K-Lipschitz continuous, which is only a small additional restriction on top of the first assumption which requires it to be homogeneously bounded.

THEOREM 3. *Assuming ϕ satisfies properties 1,2,3 and 4, and data drawn from $\mathcal{D}$, we can bound the Rademacher complexity of the Algorithmic Hypothesis Class B, with probability $1 - \delta$:*

$$\mathfrak{R}(B) \leq K\sqrt{5d\log(1/\delta)}\|V^*\|\|W^*\|_F \exp\left(\frac{-\sqrt{n}}{\sqrt{d\textit{poly}(\log d, t, k, \lambda, \nu, \sigma_1^{2p}/\rho)}}\right)$$

PROOF.

$$\mathfrak{R}(B) = \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{i=1}^{n} \sigma_i h(x_i) \tag{5.1}$$

$$= \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{j=1}^{k} v_j \phi(w_j^\top x_i) \tag{5.2}$$

$$= \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \left(\sum_{j=1}^{k} v_j \phi(w_j^\top x_i) - \sum_{j=1}^{k} v_j^* \phi(w_j^{*\top} x_i)\right) \tag{5.3}$$

$$= \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{j=1}^{k} v_j^* \left(\phi(w_j^\top x_i) - \phi(w_j^{*\top} x_i)\right) \tag{5.4}$$

$$\leq \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{j=1}^{k} v_j^* K \left|w_j^\top x_i - w_j^{*\top} x_i\right| \tag{5.5}$$

$$= \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{i=1}^{n} \sigma_i \sum_{j=1}^{k} v_j^* K \left|(w_j - w_j^*)^\top x_i\right| \tag{5.6}$$

$$\leq \mathbb{E}_\sigma \sup_{h\in B} \frac{1}{n}\sum_{j=1}^{k} v_j^* K \|w_j - w_j^*\| \sum_{i=1}^{n} \sigma_i \|x_i\| \tag{5.7}$$

$$\leq \mathbb{E}_\sigma \sup_{h\in B} \frac{K}{n} \|V^*\| \, \|W - W^*\|_F \sum_{i=1}^{n} \sigma_i \, \|x_i\| \tag{5.8}$$

$$= \sup_{h\in B} K\|V^*\| \|W - W^*\|_F \sqrt{5d\log(1/\delta)} \tag{5.9}$$

$$= K\sqrt{5d\log(1/\delta)} \|V^*\| \|W^*\|_F \exp\left(\frac{-\sqrt{n}}{\sqrt{d\text{poly}(\log d, t, k, \lambda, \nu, \sigma_1^{2p}/\rho)}}\right) \tag{5.10}$$

Step (5.3) follows because the expectation of any constant multiplied by a Rademacher variable is 0. Step (5.4) uses the assumption that $v_j^* = v_j$. Step (5.5) uses the assumption that ϕ is K-Lipschitz continuous. Step (5.7) and (5.8) use the Cauchy-Schwarz inequality, and the definition of W and V (from Chapter 3). Step (5.9) uses Lemma 1, and step (5.10) uses Theorem 2. □

If we assume that the loss is bounded by a constant C, then it follows that the loss function $l(x) = \frac{1}{2}(\hat{h}(x) - h^*(x))^2$ is C-Lipschitz continuous. Then we can use Talagrand's contraction lemma to bound the Rademacher complexity of the composition of the loss and hypothesis functions:

$$\mathfrak{R}(l \circ B) \leq C\mathfrak{R}(B)$$

We can now combine Theorem 1 and Theorem 3 to obtain a generalization bound. We use a union bound on the probability of both statements being false into $\delta = \delta_{\text{theorem1}} + d^{-\Omega(t)}$.

THEOREM 4. *Assuming the loss is bounded: $|l(h,x)| \leq C$ for any x and $h \in B$, then we have with probability $1-\delta$:*

$$L_{\mathcal{D}}(h) - L_S(h) \leq 2CK\sqrt{5d\ln(1/\delta)}\|V^*\|\|W^*\|_F \exp\left(\frac{-\sqrt{n}}{\sqrt{d\textit{poly}(\log d, \ln(1/\delta), k, \lambda, \nu, \sigma_1^{2p}/\rho)}}\right) + 4C\sqrt{\frac{2\ln(4/\delta)}{n}}.$$

Theorems 3 shows how the Rademacher complexity of the Algorithmic Hypothesis class can be related to the diameter of the class, which allows us to bound the complexity tightly despite the non-linear nature of the hypothesis class. Theorem 4 immediately follows using standard techniques, allowing us to bound the generalisation error.

5.2 A direct bound on the Excess Risk

The **Excess Risk** is a measure of how bad the output hypothesis is compared to the best possible hypothesis in the hypothesis class. Excess Risk is closely linked to the generalization error, since the excess risk can be bounded by twice the worst case generalization error, with the following well known lemma:

LEMMA 2. *For $\hat{h} \in B$ and $h^* \in B$:*

$$L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*) \leq 2\sup_{h\in B}|L_{\mathcal{D}}(h) - L_S(h)|.$$

A modification of Theorem 3 can be used to directly bound the Expected Excess Risk on the data distribution. The direct approach is more limited in its applicability, since it takes advantage of some more of the specific assumptions made in the learning setting. As we expect, we are able to find a faster bound in n, since the generalisation bound above only converges at $O(1/\sqrt{n})$. This bound takes advantage of the small diameter of the AHC directly, and avoids computing the Rademacher complexity.

THEOREM 5. *With probability $1-\delta$:*

$$L_{\mathcal{D}}(\hat{h}) - L_{\mathcal{D}}(h^*) \leq \frac{K^2 d}{\delta}\|V^*\|^2\|W^*\|_F \exp\left(\frac{-\sqrt{n}}{\sqrt{d\textit{poly}(\log d, \log(1/\delta), k, \lambda, \nu, \sigma_1^{2p}/\rho)}}\right).$$

This bound has much larger constants compared to Theorem 4, but a better rate of convergence as n increases.

PROOF. With probability $1 - d^{-\Omega(t)}$, we have:

$$\mathbb{E}_{x\sim\mathcal{D}}[l(\hat{h}(x)) - l(h^*(x)] = \mathbb{E}_{x\sim\mathcal{D}}[(\sum_{j=1}^{k} v_j\phi(w_j^\top x) - \sum_{j=1}^{k} v_j^*\phi(w_j^{*\top}x))^2] \tag{5.1}$$

$$\leq \mathbb{E}_{x\sim\mathcal{D}}[(\sum_{j=1}^{k} v_j^* K \left|(w_j - w_j^*)^\top x\right|)^2] \tag{5.2}$$

$$\leq \mathbb{E}_{x\sim\mathcal{D}}[(K\|V^*\| \sum_{j=1}^{k} \left|(w_j - w_j^*)^\top x\right|)^2]$$

$$\leq \mathbb{E}_{x\sim\mathcal{D}}[(K\|V^*\|\|x\| \sum_{j=1}^{k} \|w_j - w_j^*\|)^2]$$

$$\leq \mathbb{E}_{x\sim\mathcal{D}}[(K\|V^*\|\|x\|)^2(\sum_{j=1}^{k} \|w_j - w_j^*\|^2)]$$

$$= \mathbb{E}_{x\sim\mathcal{D}}[(K\|V^*\|\|x\|)^2\|W - W^*\|_F]$$

$$= (K\|V^*\|)^2\|W - W^*\|_F\mathbb{E}_{x\sim\mathcal{D}}[\|x\|^2]$$

$$= K^2\|V^*\|^2\|W - W^*\|_F\mathbb{E}_{x\sim\mathcal{D}}[\sum_{i=1}^{d} x_i^2]$$

$$= K^2\|V^*\|^2\|W - W^*\|_F \sum_{i=1}^{d} \mathbb{E}_{x\sim\mathcal{N}(0,1)}[x^2]$$

$$= dK^2\|V^*\|^2\|W - W^*\|_F$$

$$\leq dK^2\|V^*\|^2\|W^*\|_F \exp\left(\frac{-\sqrt{n}}{\sqrt{d\text{poly}(\log d, t, k, \lambda, \nu, \sigma_1^{2p}/\rho)}}\right)$$

Step (5.2) follows because the expected loss of the target hypothesis is 0. Other steps use similar techniques to Theorem 3.

Using Markov's inequality and a union bound, we can use this to get a bound on the Excess Risk. By setting $\delta = d^{-\Omega(t)} + \delta_{markov}$, where δ_{markov} is the probability that Markov's inequality holds, we arrive at Theorem 5. □

Theorem 5 shows that there is also an alternative approach, if we want to bound the excess risk instead of the generalisation error. This gives us a better asymptotic bound on the quality of the output hypothesis, as a direct result of the small Algorithmic Hypothesis Class. This bound has a similar rate of convergence to the recovery guarantee that it was derived from.

CHAPTER 6

Discussion

The full hypothesis class of most neural networks is too flexible to be useful for proving tight generalisation bounds using traditional complexity based techniques. By analysing the Algorithmic Hypothesis Class of a learning algorithm we obtained a generalisation bound that takes into account the data distribution and optimisation algorithm, rather than just the hypothesis class. This work shows that the complexity of the AHC can be bounded in the case of two-layer networks, in the regime where the data is high dimensional with hidden layer smaller than the dimension of the data. The technique can be understood as using the prior information we gained by making assumptions about the learning algorithm and data distribution to prove tighter generalization bounds.

The specific algorithm used by Zhong et al. (2019) converge to the target parameters very fast, which allows the resulting generalisation bound to also converge very rapidly. It is expected that the AHC gets smaller as the number of the training data grows, and this is observed in this case. The most important reason for the fast convergence is the local convexity of the loss near the target hypothesis. While this may be seen as a strength of the result, it also highlights the difference between the learning setting studied and real-world neural networks. Under normal circumstances, neural network training doesn't have the advantage of starting with a good approximation of the true parameters, and doesn't throw out each batch of data after each step of stochastic gradient descent. For these reasons, the theorems found in this research are not applicable under practical circumstances. However, theoretical results on neural networks have so far always required quite strong and unrealistic assumptions. Research using such assumptions is still worthwhile for the purpose of exploring research directions until assumptions can be found that are realistic enough to be useful in practice.

Generalisation bounds can often be categorised into two types, depending on whether the rate of convergence depends on the target hypothesis or not. Bounds that don't depend on the form of the target function are called Uniform convergence bounds, the converse being nonuniform convergence bounds

(Nagarajan and Kolter, 2019). The bounds we have found may be considered similar to non-uniform convergence bounds, since the convergence rate depends on the term $\|V^*\|\|W^*\|_F$, and the spectral values of the target weight matrix. If we assume *a priori* that the parameters of the target function have bounded norm, then we would have a bound with uniform convergence. One difference to note is that uniform and nonuniform convergence bounds are usually defined as being independent of the data distribution, whereas our bounds depend on the assumption that the data distribution is Gaussian.

The natural next question is whether this property extends to a more general class of neural networks. Some empirical evidence suggests that the AHC of deeper networks is much smaller than the full predefined hypothesis space. For example, different networks have been found to learn similar representations of data despite very different architectures (Olah et al., 2020; Li et al., 2016). Under the reasonable assumption that these representations would be maintained in the expected hypothesis, this observation suggests that the AHC is small in representation space, since a large AHC implies that many different hypotheses might be chosen for a given dataset, so it is unlikely they are similar.

Recent work (Jacot et al., 2018; Lee et al., 2018) showing that trained deep networks can be viewed as being approximately sampled from a Gaussian Process also suggests a small AHC, since the variance of the Gaussian Process is likely to be small if n is large, even when the network is massively overparameterised. This would imply that the set of likely output hypotheses is small, however additional assumptions would have to be made on the form of the target function in order to show that a high probability set of hypotheses must contain the target function. This line of research is an example of a set of assumptions (infinite width networks) that may be realistic enough that they end up being usefully applied to better understand neural networks used in practice. Future work is planned to investigate this.

Overall this research has made incremental progress in extending a new approach to generalisation analysis to neural networks. While the results are limited by strong assumptions, they confirm that it is at least in principle possible to define and use the AHC of a neural network, and the research direction of extending the method further is promising.

Bibliography

Shunichi Amari. 2020. Any Target Function Exists in a Neighborhood of Any Sufficiently Wide Random Network: A Geometrical Perspective. *Neural Computation*, 32(8):1431–1447.

Ainesh Bakshi, Rajesh Jayaram, and David P. Woodruff. 2019. Learning Two Layer Rectified Neural Networks in Polynomial Time. In *Conference on Learning Theory*, pages 195–268. PMLR. ISSN: 2640-3498.

Peter L. Bartlett, Olivier Bousquet, and Shahar Mendelson. 2005. Local Rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537. ArXiv: math/0508275.

Peter L. Bartlett, Philip M. Long, Gábor Lugosi, and Alexander Tsigler. 2020. Benign Overfitting in Linear Regression. *arXiv:1906.11300 [cs, math, stat]*. ArXiv: 1906.11300.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine learning practice and the bias-variance trade-off. *arXiv:1812.11118 [cs, stat]*. ArXiv: 1812.11118.

Mikhail Belkin, Siyuan Ma, and Soumik Mandal. 2018. To understand deep learning we need to understand kernel learning. *arXiv:1802.01396 [cs, stat]*. ArXiv: 1802.01396.

Yuheng Bu, Shaofeng Zou, and Venugopal V. Veeravalli. 2020. Tightening Mutual Information Based Bounds on Generalization Error. *IEEE Journal on Selected Areas in Information Theory*, 1(1):121–130. ArXiv: 1901.04609.

Giacomo De Palma, Bobak Toussi Kiani, and Seth Lloyd. 2019. Random deep neural networks are biased towards simple functions. *arXiv:1812.10156 [cond-mat, physics:math-ph, physics:quant-ph, stat]*. ArXiv: 1812.10156.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. 2017. Sharp Minima Can Generalize For Deep Nets. In *International Conference on Machine Learning*, pages 1019–1028. PMLR. ISSN: 2640-3498.

Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. 2019. Gradient Descent Provably Optimizes Over-parameterized Neural Networks. *arXiv:1810.02054 [cs, math, stat]*. ArXiv: 1810.02054.

Micah Goldblum, Jonas Geiping, Avi Schwarzschild, Michael Moeller, and Tom Goldstein. 2019. Truth or backpropaganda? An empirical investigation of deep learning theory.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. 2019. Size-Independent Sample Complexity of Neural Networks. *arXiv:1712.06541 [cs, stat]*. ArXiv: 1712.06541.

Nick Harvey, Christopher Liaw, and Abbas Mehrabian. 2017. Nearly-tight VC-dimension bounds for piecewise linear neural networks. In *Proceedings of the 2017 Conference on Learning Theory*, pages 1064–1068. PMLR. ISSN: 2640-3498.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Flat Minima. *Neural Computation*, 9(1):1–42.

W. Ronny Huang, Zeyad Emam, Micah Goldblum, Liam Fowl, Justin K. Terry, Furong Huang, and Tom Goldstein. 2020. Understanding Generalization Through Visualizations. pages 87–97. PMLR. ISSN: 2640-3498.

Arthur Jacot, Franck Gabriel, and Clément Hongler. 2018. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. *NeurIPS*.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. 2019. Fantastic Generalization Measures and Where to Find Them. *arXiv:1912.02178 [cs, stat]*. ArXiv: 1912.02178.

Dimitris Kalimeris, Gal Kaplun, Preetum Nakkiran, Benjamin Edelman, Tristan Yang, Boaz Barak, and Haofeng Zhang. 2019. SGD on Neural Networks Learns Functions of Increasing Complexity. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. ON LARGE-BATCH TRAINING FOR DEEP LEARNING: GENERALIZATION GAP AND SHARP MINIMA. page 16.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of Neural Network Representations Revisited. page 11.

B. Laurent and P. Massart. 2000. Adaptive estimation of a quadratic functional by model selection. *The Annals of Statistics*, 28(5):1302–1338. Publisher: Institute of Mathematical Statistics.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2018. Deep Neural Networks as Gaussian Processes.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. 2016. Convergent Learning: Do different neural networks learn the same representations? *arXiv:1511.07543 [cs]*. ArXiv: 1511.07543.

Yuanzhi Li and Yang Yuan. 2017. Convergence Analysis of Two-layer Neural Networks with ReLU Activation. *arXiv:1705.09886 [cs]*. ArXiv: 1705.09886.

Zhiyuan Li, Ruosong Wang, Dingli Yu, Simon S. Du, Wei Hu, Ruslan Salakhutdinov, and Sanjeev Arora. 2019. Enhanced Convolutional Neural Tangent Kernels. *arXiv:1911.00809 [cs, stat]*. ArXiv: 1911.00809.

Tongliang Liu, Gábor Lugosi, Gergely Neu, and Dacheng Tao. 2017. Algorithmic stability and hypothesis complexity. *arXiv:1702.08712 [cs, stat]*. ArXiv: 1702.08712.

Vaishnavh Nagarajan and J. Zico Kolter. 2019. Uniform convergence may be unable to explain generalization in deep learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 1042, pages 11615–11626. Curran Associates Inc., Red Hook, NY, USA.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2019. Deep Double Descent: Where Bigger Models and More Data Hurt. *arXiv:1912.02292 [cs, stat]*. ArXiv: 1912.02292.

Jeffrey Negrea, Gintare Karolina Dziugaite, and Daniel Roy. 2020. In Defense of Uniform Convergence: Generalization via Derandomization with an Application to Interpolating Predictors. In *Proceedings of the 37th International Conference on Machine Learning*, pages 7263–7272. PMLR. ISSN: 2640-3498.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. 2015. Norm-Based Capacity Control in Neural Networks. *arXiv:1503.00036 [cs, stat]*. ArXiv: 1503.00036.

Roman Novak, Yasaman Bahri, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2018. Sensitivity and Generalization in Neural Networks: an Empirical Study.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom In: An Introduction to Circuits. *Distill*, 5(3):e00024.001.

Daniel A. Roberts, Sho Yaida, and Boris Hanin. 2021. The Principles of Deep Learning Theory. *arXiv:2106.10165 [hep-th, stat]*. ArXiv: 2106.10165.

Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. 2020. The Pitfalls of Simplicity Bias in Neural Networks. *arXiv:2006.07710 [cs, stat]*. ArXiv: 2006.07710.

Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, Cambridge.

Ohad Shamir. 2016. Distribution-Specific Hardness of Learning Neural Networks. *Journal of Machine Learning Research*, 19.

Samuel L. Smith, Benoit Dherin, David G. T. Barrett, and Soham De. 2021. On the Origin of Implicit Regularization in Stochastic Gradient Descent. *arXiv:2101.12176 [cs, stat]*. ArXiv: 2101.12176.

Mahdi Soltanolkotabi, Adel Javanmard, and Jason D. Lee. 2019. Theoretical Insights Into the Optimization Landscape of Over-Parameterized Shallow Neural Networks. *IEEE Transactions on Information Theory*, 65(2):742–769. Conference Name: IEEE Transactions on Information Theory.

Mahsa Taheri, Fang Xie, and Johannes Lederer. 2021. Statistical guarantees for regularized neural networks. *Neural Networks*, 142:148–161.

Guillermo Valle-Perez, Chico Q. Camargo, and Ard A. Louis. 2018. Deep learning generalizes because the parameter-function map is biased towards simple functions.

V.N. Vapnik. 1999. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988–999. Conference Name: IEEE Transactions on Neural Networks.

Andrew G Wilson and Pavel Izmailov. 2020. Bayesian Deep Learning and a Probabilistic Perspective of Generalization. In *Advances in Neural Information Processing Systems*, volume 33, pages 4697–4708. Curran Associates, Inc.

David H. Wolpert. 1996. The Lack of A Priori Distinctions Between Learning Algorithms. *Neural Computation*, 8(7):1341–1390. Conference Name: Neural Computation.

Aolin Xu and M. Raginsky. 2017. Information-theoretic analysis of generalization capability of learning algorithms. In *NIPS*.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization.

Xiao Zhang, Yaodong Yu, Lingxiao Wang, and Quanquan Gu. 2019. Learning One-hidden-layer ReLU Networks via Gradient Descent. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1524–1534. PMLR. ISSN: 2640-3498.

Kai Zhong, Zhao Song, Prateek Jain, Peter L. Bartlett, and Inderjit S. Dhillon. 2017. Recovery Guarantees for One-hidden-layer Neural Networks. *arXiv:1706.03175 [cs, stat]*. ArXiv: 1706.03175.