

Explainability and Generalisation Properties of Artificial Deep Neural Networks

Ouns El Harzli



Department of Computer Science
University of Oxford
New College

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2025

Supervisor: Prof. Bernardo Cuenca Grau

Abstract

This thesis investigates two foundational questions about modern deep neural networks: (i) how to make their predictions explainable, and (ii) how to predict and understand their generalisation performance. We develop new theory and algorithms on formal methods for interpretability using computational complexity theory and logic; and new kernel, Bayesian and learning-theoretic perspectives on generalisation.

On explainability, we first study two canonical decision problems — Minimum Change Required and Minimum Sufficient Reason — used to compute cardinality-minimal contrastive and abductive explanations for neural network predictions. We prove that these problems remain intractable even for monotonic networks, but we show tractability is recovered under practical regularity assumptions (continuity everywhere and almost-everywhere differentiability of activations), yielding efficient greedy algorithms. Our analysis also uncovers new links to attribution methods via Aumann–Shapley values. We further introduce a logic-based framework that uses the fibring of modal logics to characterize the non-uniform expressiveness of certain classes of graph neural networks and transformer encoders, establishing along the way the exact correspondence between fibring of modal logics and fibring of neural networks. The goal of our fibring formalism is to lay the foundations for the development of a methodology with formal guarantees to extract interpretable logical rules from trained neural networks.

On generalisation, we characterise properties of the Mercer spectrum of the empirical kernel associated with neural networks and leverage this viewpoint to analyse their Bayesian training, deriving an extension to the renormalisation theory of deep neural networks, and providing a new interpretation of the double-descent phenomenon. Using this perspective, we also show that adversarial examples can be, in interpolating regression settings, described as inputs that shift the empirical kernel’s spectrum toward near-zero eigenvalues; helping reconcile strong test performance with adversarial vulnerability. Finally, from the learning-theoretic viewpoint, we give sufficient conditions for algorithmic stability of minimum-norm interpolating deep ReLU networks: the existence of a stable prefix sub-network followed by at least one low-rank layer guarantees stability, linking generalisation to training-induced structures such as low rank and neural collapse.

Together, these contributions make several steps towards understanding how overparameterised networks can generalise on unseen data and how their decisions can be rendered intelligible.

Acknowledgements

First and foremost, I am deeply grateful to my supervisor Prof. Bernardo Cuenca Grau for being an exceptional mentor; for his incredibly helpful advice and patient teaching throughout my DPhil studies. The lessons I have learned from him are ones I will carry for life; his guidance has made me a better researcher and has profoundly influenced my way of thinking.

To my wife, thank you for your unwavering support and for being loving and caring through every stage of this journey. It would not have been possible without you.

I also thank my brother for his friendship and for the many thoughtful, spirited conversations about this thesis.

My heartfelt thanks go to my mother for her love and her constant encouragement.

Finally, I am grateful to all my collaborators and co-authors for the stimulating scientific discussions and exchange of ideas. A special thanks to my friend and closest collaborator, Dr. Yoonsoo Nam, for the countless conversations about our research and about life in general that made this journey both enriching and enjoyable. I am also thankful to Dr. Ilja Kuzborskij for many engaging discussions and for introducing me to the beauty of learning theory. I am also grateful to Prof. Ard Louis and Prof. Ian Horrocks for being great mentors, and for teaching me how to approach research with curiosity, purpose, and enthusiasm, and make the most of this incredible experience.

Contents

1. Introduction	1
1.1. Research Questions	3
1.2. Contributions	3
1.3. Thesis organisation	4
1.4. Publication Correspondences	5
2. Preliminaries	6
2.1. Neural Networks and Training Algorithms	6
2.2. Kernel Theory and Random Matrix Theory	8
2.3. Bayesian Neural Networks and Gaussian Processes	9
 I. Explainability of Neural Networks	 11
3. Overview of Explainability and Interpretability	12
3.1. Rule-based methods	12
3.2. Attribution-based Methods	13
3.3. Perturbation-based Methods	15
3.4. Example-based Methods	17
3.5. Abductive and Contrastive Methods	18
3.6. Mechanistic Interpretability	21
4. Minimal Explanations in Neural Networks	24
4.1. Intractability for Monotonic Networks	26
4.2. Achieving Tractability of MCR and MSR	28
4.2.1. Properties of Monotonic Networks with Admissible Activation	29
4.2.2. Algorithms for Computing Explanations	30
4.3. Discussion and Further Implications	32
4.4. Experiments	33
4.5. Concluding remarks of the chapter	35
5. Fibring: a formalism connecting neural networks and logical theories	36
5.1. Fibring Neural Networks	38
5.2. Exact correspondence between fibring logics and fibring networks	39
5.3. Application to non-uniform expressiveness of GNNs, GATs and Transformer encoders	43
5.3.1. Formal definitions of GNNs and Transformer Encoders.	43
5.3.2. Fibred neural networks can non-uniformly reproduce GNNs, GATs and Transformer encoders	45
5.3.3. GNNs, GATs and Transformer encoders as fragments of fibred logics	47

5.4. Discussion	47
5.5. Concluding remarks of the chapter	49
II. Generalisation of Neural Networks	50
6. Overview of Generalisation Properties	51
6.1. Statistical Learning Theory Approaches	51
6.2. Statistical Physics Approaches	53
6.3. Dynamics of Gradient-Based Training	55
6.4. Bayesian Perspectives	58
7. Mercer Spectrum of Empirical Kernels	61
7.1. A Width-dependent Random Kernel	62
7.2. Limiting Spectral Distributions	65
7.3. The Modified NNGP Kernel	71
7.4. Experiments	73
7.5. Concluding remarks of the chapter	74
8. The Double-Descent Phenomenon	76
8.1. Double Descent from the Perspective of Gaussian Processes Kernels . . .	77
8.2. Experiments	83
8.3. Concluding remarks of the chapter	86
9. Infinite BNNs through Kernel Theory	88
9.1. BNNs as Modified GP Regression	89
9.1.1. Training BNNs with the Modified NNGP Kernel	89
9.1.2. An Extended Renormalisation Theory	92
9.1.3. Application to the Sublinear-Width Regime	93
9.2. Experiments	94
9.3. Concluding remarks of the chapter	95
10. Adversarial Attacks and Near-Zero Eigenvalues	97
10.1. Adversarial Attacks Shift Mercer’s Spectrum Towards Zero	100
10.2. Adversarial Examples Are Exceedingly Unlikely	104
10.3. Experiments	105
10.4. Concluding remarks of the chapter	106
11. Algorithmic Stability and Low-Rank Structure	109
11.1. From sub-network stability to prediction stability	112
11.1.1. Proof of the main technical result	114
11.2. Technical proofs	118
11.2.1. Minimum-norm interpolants and low-rank bias	118
11.2.2. Low-rank layers preserve learned signals	119
11.2.3. Minimum-norm interpolants and margins	120
11.3. Concluding remarks of the chapter	121
12. Conclusion	124
12.1. Summary of Contributions	124

Contents

12.2. Future Research Directions	125
--	-----

1. Introduction

Neural networks are at the heart of the current revolution in artificial intelligence. Over the past decade, they have powered breakthroughs that were once thought to be decades away: superhuman performance in games such as Go (Silver et al. 2016), protein structure prediction at scale (Jumper et al. 2021), automated mathematics and science discovery (Fawzi et al. 2022), state-of-the-art reasoning benchmarks (OpenAI 2023; Bubeck et al. 2023), and human-level image, audio, and video generation (Rombach et al. 2022; Ho et al. 2022; Villegas et al. 2023). Each new generation of neural models has redefined benchmarks and widened the scope of tasks for which artificial intelligence is competitive with human experts.

The progress of deep learning has been closely tied to the development of new architectures. Early successes came with fully-connected feedforward networks (FCNs) (LeCun et al. 1998a), followed by convolutional neural networks (CNNs) for vision (LeCun et al. 1998a; Krizhevsky et al. 2012) and recurrent neural networks (RNNs) for sequence modelling (Hochreiter & Schmidhuber 1997b; Cho et al. 2014). Graph neural networks (GNNs) extended deep learning to non-Euclidean domains (Scarselli et al. 2009; Kipf & Welling 2017), while U-Net architectures became the backbone of dense prediction tasks in vision and medicine (Ronneberger et al. 2015). On the generative side, generative adversarial networks (GANs) (Goodfellow et al. 2014) defined a research frontier for nearly a decade until diffusion-based models (Ho et al. 2020; Dhariwal & Nichol 2021) emerged as the dominant paradigm, paired with U-Net architectures for high-resolution synthesis (Rombach et al. 2022). In language and reasoning, the advent of the Transformer (Vaswani et al. 2017) and the scaling of large language models (LLMs) such as GPT-3 (Brown et al. 2020) and GPT-4 (OpenAI 2023) have transformed natural language processing, code generation, and increasingly, multimodal learning.

Despite their astonishing practical success, the *theoretical foundations* of neural networks remain incomplete. Even the most basic fully-connected feedforward networks resist full mathematical characterisation. Questions of *generalisation*—why these highly over-parameterised and expressive models are able to generalise so well in practice—and *explainability*—how to interpret their predictions and internal representations—remain open problems (Zhang et al. 2016; Nagarajan & Kolter 2019). Addressing these challenges is not only of fundamental scientific interest but also of practical importance for improving architectures, designing safer training protocols, and ensuring robustness when these models are deployed in sensitive, high-stakes applications.

Explainability. Neural networks are notoriously difficult to interpret: their predictions rely on embeddings in high-dimensional and highly non-linear spaces, making them appear as “black boxes”. As performance alone is not sufficient for real-world deployment, criteria such as safety, fairness, and robustness have gained increasing prominence (Kazim & Koshiyama 2021). For large language models, this includes mitigating *hallucinations*, ensuring *alignment*, and defending against misuse via *adversarial prompts* and *jailbreak* attacks (Zou et al. 2023; Hayase et al. 2024; Perez et al. 2022;

1. Introduction

Greshake, Chevalier, et al. 2023). For diffusion models, explainability requires tracing the influence of training data on generated samples and understanding how semantic features emerge during denoising. Improving explainability is thus a prerequisite for safe and trustworthy AI.

Robustness. Neural networks are highly sensitive to adversarial perturbations (Szegedy et al. 2014; Hendrycks & Dietterich 2019). Small, often imperceptible changes to an input can flip predictions, creating vulnerabilities in mission-critical applications (Katz et al. 2017). Foundation models expand this concern: in addition to adversarial perturbations, they can be exploited through *prompt injection*, which manipulate generation behavior or circumvent safety filters. These vulnerabilities underscore that robustness and explainability are deeply linked, and that principled theoretical frameworks are necessary to anticipate and mitigate such risks.

Generalisation. Another defining characteristic of modern neural networks is their extreme overparameterisation. The best-performing models today have far more trainable parameters than training examples—VGG (Simonyan & Zisserman 2014), GPT-3 (Brown et al. 2020), or Stable Diffusion (Rombach et al. 2022) being prime examples. Classical statistical wisdom, grounded in the bias–variance trade-off (Mohri et al. 2012; Vapnik 1995; Shalev-Shwartz & Ben-David 2014), predicts that such models should overfit catastrophically. Yet, in practice, they not only avoid overfitting but often exhibit *benign overfitting* (Bartlett et al. 2020) and even the striking *double descent* phenomenon, whereby generalisation error improves again far into the overparameterised regime (Belkin et al. 2019; Nakkiran et al. 2021). Understanding why these models generalise so well, even when they are universal approximators (Hornik 1991), is one of the central theoretical challenges of modern machine learning.

Optimisation. Finally, the optimisation landscapes of neural networks are highly nonconvex (Li et al. 2018), and training algorithms based on stochastic gradient descent (SGD) (Duchi et al. 2011; Kingma & Ba 2014) introduce stochasticity both in initialization and during training. There is no guarantee of reaching a global optimum, or even the same local optimum across runs. Yet, empirical observations show that most local minima discovered across multiple training instances yield similar generalisation performance (Zhang et al. 2016). Additional dynamics for foundation models, such as instruction tuning and reinforcement learning from human feedback (RLHF), add further complexity, introducing inductive biases and stability considerations that are not yet fully understood (Ouyang et al. 2022).

This thesis being largely a theoretical work (with some experiments to validate the findings), it focuses mainly on the basic fully-connected architecture, for which the theoretical questions are still unsolved, with the exceptions of Chapter 5 which treats GNNs and transformer encoders, more amenable to logical characterisations, and Chapter 10, which considers a slightly more general definitions of neural networks, encompassing CNNs.

In summary, neural networks embody a paradox: they are universal approximators, massively overparameterised, sensitive to perturbations, and trained with stochastic algorithms on highly nonconvex landscapes—yet they consistently achieve remarkable generalisation. This paradox highlights two grand theoretical challenges: *explainability* and *generalisation*. These challenges are not yet solved even for the most basic fully-connected architectures (Zhang et al. 2016; Nagarajan & Kolter 2019; Zhang et

1. Introduction

al. 2021), motivating continued theoretical study of simplified settings. This thesis addresses these intertwined questions: understanding how to systematically *explain* the patterns learned by neural networks, and why these models *generalise* as well as they do. The technical core of both problems is deeply related: adversarial examples and prompt-based attacks challenge both robustness and interpretability, while theoretical explanations of generalisation often reveal the same structures that could underpin interpretability. Thus, progress on either front informs the other, and together they form the central focus of this work.

This thesis treats these two intertwined problems through theoretical lenses: (i) *explainability*—developing formal methods to make model predictions intelligible; and (ii) *generalisation*—using kernel, Bayesian, and learning-theoretic viewpoints to characterise when and why overparameterised models generalise.

1.1. Research Questions

The work is driven by the following research questions (RQs).

- RQ1 Minimal explanations.** Under what conditions can one compute *cardinality-minimal* explanations of neural model decisions—specifically, the *Minimum Change Required* (contrastive) and *Minimum Sufficient Reason* (abductive) problems—for neural architectures? What is the computational complexity of these tasks, and when are efficient algorithms possible?
- RQ2 Reverse-engineering logic from networks.** Can we systematically extract logical theories that reproduce the behaviour of modern neural architectures (e.g., GNNs and transformer encoders), and how can logical *fibring* help to compose such explanations?
- RQ3 Spectral structure and generalisation.** What does the *Mercer spectrum* of an empirical kernel associated with a neural network reveal about its generalisation power? How does this connect to Gaussian-process (GP) and Bayesian views of learning?
- RQ4 Double descent.** Which mechanisms produce the double-descent phenomenon, and how can a kernel/GP perspective explain its shape and regimes?
- RQ5 Adversarial vulnerability.** Can one derive a precise spectral characterisation of adversarial examples—e.g., via near-zero eigenvalues—and what does this imply for robustness vs. performance?
- RQ6 Stability and structure.** Which training-induced structures (e.g., low-rank layers, neural collapse) are sufficient for algorithmic stability, and how do such conditions translate into generalisation guarantees?

1.2. Contributions

This thesis contributes theory and algorithms in formal methods for interpretability and theories of neural network generalisation.

Explainability.

- **Cardinality-minimal explanations for neural predictions.** We study the canonical decision problems *Minimum Change Required* (MCR) and *Minimum Sufficient Reason* (MSR) for neural predictors, establish new computational hardness results, identify practically relevant conditions under which efficient procedures exist, and establish new connections with attribution methods via integrated gradients (El Harzli et al. 2023b).
- **Logic from neural networks via fibring.** We propose a logic-based framework, *fibring*, to combine neural networks, and we use this formalism to derive non-uniform logical expressiveness results for GNNs and Transformer encoders. To that aim, we also establish for the first time the exact formal correspondence between fibring of modal logics and fibring of neural networks. We present arguments towards developing a methodology to reverse-engineer logical theories from trained neural models based on this formalism (El Harzli et al. 2026).

Generalisation.

- **Mercer spectrum of empirical kernels.** We propose the Mercer spectrum of the empirical kernel as a key object to study the generalisation properties of neural networks. We contribute to random matrix theory by deriving novel limiting spectral distributions of certain random kernel matrices that arise in neural networks, and we study the asymptotic behavior of the empirical kernel under relevant limits (El Harzli et al. 2024a; El Harzli & Cuenca Grau 2025).
- **A kernel/GP account of double descent.** We characterize double-descent curves via the discrepancy between width-dependent random kernels and the width-independent *Neural Network Gaussian Process* (NNGP) kernel, deriving analytic predictions for the peak and the different regimes (El Harzli et al. 2024a).
- **Bayesian training via treatment of the spectrum.** We derive a spectral description of the training of Bayesian neural networks (BNNs), which enables an extension of the existing renormalisation theory of deep neural networks to new regimes (El Harzli & Cuenca Grau 2025).
- **Adversarial examples as near-zero eigenvalues.** We show that adversarial examples in interpolating regression settings correspond to spectral mass near zero in the empirical kernel, helping reconcile the apparent contradiction between strong test accuracy and adversarial vulnerability (El Harzli & Cuenca Grau 2024).
- **Algorithmic stability via low-rank structure.** We give sufficient conditions for stability of minimum-norm interpolating deep ReLU networks: the existence of a stable prefix followed by at least one low-rank layer guarantees stability and connects stability to neural-collapse geometry (El Harzli et al. 2024b).

1.3. Thesis organisation

The dissertation is organised in two parts.

Part I: Explainability Chapter 3 surveys explainability and interpretability research,

1. Introduction

emphasizing methods with formal guarantees. Chapter 4 studies the minimal-explanation framework (MCR/MSR), deriving new complexity results, and algorithms, and establishes the connection with attribution methods. Chapter 5 introduces the logic-based fibring framework, establishes the correspondence between fibring of modal logics and fibring of neural networks, derives non-uniform expressiveness results for GNNs/transformers and discusses applications to interpretability and verification.

Part II: Generalisation Chapter 6 reviews learning-theoretic, kernel, Bayesian, and statistical-physics perspectives on generalisation. Chapter 7 studies the Mercer spectrum of empirical kernels in neural networks and its asymptotic behavior, and derives new limiting spectral distributions for the corresponding random kernel matrices. Chapter 8 studies the double descent phenomenon through a kernel/GP lens. Chapter 9 analyses the Bayesian training of neural networks via spectral viewpoint. Chapter 10 characterizes adversarial vulnerability in interpolating regression setting with near-zero eigenvalues. Chapter 11 presents stability results in connection with low-rank structure and neural collapse.

The concluding Chapter 12 summarises the main findings and outlines future work.

1.4. Publication Correspondences

Several chapters are adapted from the following peer-reviewed publications.

- **Chapter 4** — [El Harzli et al. \(2023b\)](#).
- **Chapter 5** — [El Harzli et al. \(2026\)](#).
- **Chapter 7** — [El Harzli et al. \(2024a\)](#) and [El Harzli & Cuenca Grau \(2025\)](#).
- **Chapter 8** — [El Harzli et al. \(2024a\)](#).
- **Chapter 9** - [El Harzli & Cuenca Grau \(2025\)](#)
- **Chapter 10** — [El Harzli & Cuenca Grau \(2024\)](#).
- **Chapter 11** — [El Harzli et al. \(2024b\)](#).

2. Preliminaries

Notation. We use standard notation for vectors $\mathbf{v} \in \mathbb{R}^n$, matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$, and their transposes $\mathbf{v}^T$ and $\mathbf{A}^T$. As usual, $A_{i,j}$ denotes the (i,j) -element of $\mathbf{A}$ and v_i the i -th element of $\mathbf{v}$. We use $\mathbf{a}_i$ to denote the vector in the i -th row of $\mathbf{A}$. If two matrices have the same number of columns or the same number of rows, we can place them side by side to form a larger matrix, called their *concatenation*. The Moore-Penrose pseudo-inverse of a matrix $\mathbf{A}$ is denoted as $\mathbf{A}^\dagger$ (Moore 1920), and the trace of $\mathbf{A}$ is denoted $\text{Tr}(\mathbf{A})$. We denote as $\mathbf{0}_n$ the n -dimensional zero vector. Given $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ and a subset $S \subseteq \{1, \dots, n\}$ of their components, we denote with $\mathbf{x}^{S|\mathbf{x}'}$ the vector obtained from $\mathbf{x}$ by setting each component x_i with $i \in S$ to x'_i . The complement $\bar{\mathbf{x}}$ of a Boolean vector $\mathbf{x} \in \{0, 1\}^n$ is obtained from $\mathbf{x}$ by replacing each 0 with 1 and vice-versa. For L_2 norms we write $\|\cdot\|_2$ or sometimes $\|\cdot\|$ when there is no ambiguity. The Frobenius norm of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with singular values s_j for $1 \leq j \leq \min(m, n)$ is given by $\|\mathbf{A}\|_F = (\sum_j s_j^2)^{\frac{1}{2}}$. The spectral norm of $\mathbf{A}$, that is $\|\mathbf{A}\|_\sigma$ is given by the largest absolute value of its singular values. The stable rank of matrix $\mathbf{A}$ is defined as $\text{S}(\mathbf{A}) = \|\mathbf{A}\|_F^2 / \|\mathbf{A}\|_\sigma^2$. In particular, matrix $\mathbf{A}$ has stable rank 1 if and only if its rank (defined in the usual way) is also one (Tropp 2015); in this case, its singular value with largest absolute value is also the only non-zero singular value. Finally, expectations are noted $\mathbb{E}[\cdot]$ and probabilities $\mathbb{P}(\cdot)$.

2.1. Neural Networks and Training Algorithms

Neural Architectures. A neural architecture with L layers is a tuple

$$\mathbb{A} = \langle \{d_\ell\}_{\ell=0}^L, \{\sigma^\ell\}_{\ell=1}^{L-1} \rangle,$$

where $d_\ell \in \mathbb{N}$ is the *dimension* of layer ℓ . For each $\ell \in \{1, \dots, L-1\}$, $\sigma^\ell : \mathbb{R}^{d_\ell} \rightarrow \mathbb{R}^{d_\ell}$ is a polytime computable *activation map*. In this thesis, we will often assume homogeneous widths $d_1 = \dots = d_{L-1} = d$, and homogeneous activations $\sigma^1 = \dots = \sigma^{L-1} = \sigma$. We will refer to d_0 and d_L as the input and output dimensions respectively. For example, σ^ℓ can be the ReLU activation function applied componentwise. For $1 \leq p \leq q \leq L$, we denote as $\mathbb{A}^{p:q}$ the neural architecture $\langle \{d_\ell\}_{\ell=p-1}^q, \{\sigma^\ell\}_{\ell=p}^{q-1} \rangle$; for simplicity, if $q = L$, we write $\mathbb{A}^{p\uparrow}$ instead of $\mathbb{A}^{p:L}$.

Network Instances, Empirical Feature Maps and Sub-networks. A *fully-connected neural network* (FCN) or *network instance* of $\mathbb{A}$ is a tuple $\mathcal{N} = \langle \{\mathbf{W}^\ell\}_{\ell=1}^L, \{\mathbf{b}^\ell\}_{\ell=1}^L \rangle$. For each layer ℓ , $\mathbf{W}^\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ is a weight matrix and $\mathbf{b}^\ell \in \mathbb{R}^{d_\ell}$ is a bias vector. The application of $\mathcal{N}$ to $\mathbf{x} \in \mathbb{R}^{d_0}$ generates a sequence $\mathbf{h}^1, \dots, \mathbf{h}^L$ of vectors defined as

$$\mathbf{h}^\ell = \mathbf{W}^\ell \cdot \mathbf{x}^{\ell-1} + \mathbf{b}^\ell, \quad (2.1)$$

2. Preliminaries

where $\mathbf{x}^0 = \mathbf{x}$ and $\mathbf{x}^\ell = \sigma^\ell(\mathbf{h}^\ell)$. The result $\mathcal{N}(\mathbf{x})$ of applying $\mathcal{N}$ to $\mathbf{x}$ is the vector $\mathbf{h}^L$; thus, $\mathcal{N}$ realises a function $\mathcal{N} : \mathbb{R}^{d_0} \mapsto \mathbb{R}^{d_L}$. We assume that the last layer has no bias $\mathbf{b}^L = 0$, and in this configuration, the last-layer weights $\mathbf{W}^L$ are referred to as *readout weights*. Let Θ represent all parameters of $\mathcal{N}$ (weight matrices and biases) up to layer $L - 1$ and let θ be the collection of all parameters of $\mathcal{N}$, i.e. $\theta = (\Theta, \mathbf{W}^L)$. We sometimes note $\mathcal{N}(\cdot) = \mathcal{N}(\cdot; \theta)$ to emphasize the dependency in the parameters. The *empirical feature map* $\phi(\Theta, \cdot) : \mathbb{R}^{d_0} \mapsto \mathbb{R}^{d_{L-1}}$ is the nonlinear transformation mapping the input to the activation $\mathbf{x}^{L-1}$. We sometimes note the feature map $\phi_{\mathcal{N}}$ for the feature map of network $\mathcal{N}$. By definition, $\mathcal{N}(\mathbf{x}, \theta) = (\mathbf{W}^L)^T \phi(\Theta, \mathbf{x})$, and to highlight the parameter dependency we sometimes note it as $\mathcal{N}_{\Theta, \mathbf{W}^L}$.

For $1 \leq p \leq q \leq L$, the *sub-network* $\mathcal{N}^{p:q}$ is the instance of $\mathbb{A}^{p:q}$ defined as

$$\langle \{\mathbf{W}^\ell\}_{\ell=p}^q, \{\mathbf{b}^\ell\}_{\ell=p}^q \rangle.$$

For simplicity, if $q = L$, we also note $\mathcal{N}^{p\uparrow}$ instead of $\mathcal{N}^{p:L}$. We will sometimes refer to the Frobenius norm of a neural network $\mathcal{N}$ as that of the matrix obtained by concatenating all weight matrices of $\mathcal{N}$.

Minimum-norm interpolating neural network. A training set $(\mathbf{X}, \mathbf{y})$ consists of n examples $(\mathbf{x}_i, y_i)$ sampled i.i.d. from an unknown distribution p on $\mathbb{R}^{d_0} \times \mathbb{R}$. Furthermore, we denote by $(\mathbf{X}^{(j)}, \mathbf{y}^{(j)})$ the training dataset obtained from $(\mathbf{X}, \mathbf{y})$ by re-sampling j -th example according to p independently and we note $(\mathbf{x}^{(j)}, y^{(j)})$ the new sample. We say that $\mathcal{N}$ interpolates $(\mathbf{X}, \mathbf{y})$ if $\mathcal{N}(\mathbf{x}_i) = y_i$ for each $i \in \{1, \dots, n\}$. Given a neural architecture $\mathbb{A} = \langle L, d, d_0 \rangle$ and a training set $(\mathbf{X}, \mathbf{y})$, we assume that we have access to an algorithm $\mathcal{T}$ which returns the parameters of a minimum-norm interpolating neural network instantiating $\mathbb{A}$. We then denote

$$\mathcal{T}(\mathbf{X}, \mathbf{y}) = \arg \min_{\theta} \left\{ \|\theta\|^2 : \forall i \in \{1, \dots, n\} \quad \mathcal{N}(\mathbf{x}_i; \theta) = y_i \right\}.$$

Finally, we denote the parameters obtained by algorithm $\mathcal{T}$ as $\hat{\theta} = \mathcal{T}(\mathbf{X}, \mathbf{y})$.

Training to interpolation and minimum-norm solutions are common assumptions which represent idealized views of gradient-based training algorithms (in particular with weight decay) (Timor et al. 2023; Galanti et al. 2023). For example, in the limit of infinite training time, gradient flow on ReLU networks converges to a minimum-norm interpolant (Lyu & Li 2019; Ji & Telgarsky 2020).

Algorithmic stability. Algorithm $\mathcal{A}$ is β -uniformly ϵ -stable (Kutin & Niyogi 2002) with respect to a data distribution p if, for each training set $(\mathbf{X}, \mathbf{y})$ sampled from p , the following holds for $i \in \{1, \dots, n\}$, where $\hat{\theta} = \mathcal{A}(\mathbf{X}, \mathbf{y})$ and $\hat{\theta}^{(i)} = \mathcal{A}(\mathbf{X}^{(i)}, \mathbf{y}^{(i)})$:

$$\mathbb{P} \left(\left| \mathcal{N}(\mathbf{x}; \hat{\theta}) - \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)}) \right| \leq \epsilon \right) \geq 1 - \beta,$$

where $\mathbb{P}()$ is taken with respect to the jointly distributed $(\mathbf{X}, \mathbf{y}, \mathbf{x}^{(i)}, y^{(i)}, \mathbf{x}, y)$.

This notion of stability is weaker than the well-known notion of ϵ -uniform stability (Bousquet & Elisseeff 2002): $\sup_{\mathbf{x}, \mathbf{y}, \mathbf{x}^{(i)}, y^{(i)}} |\mathcal{N}(\mathbf{x}; \hat{\theta}) - \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)})| \leq \epsilon$, however both coincide for

2. Preliminaries

$\beta = 0$ almost surely. Often, we will say that the algorithm is stable when for a fixed β , $\epsilon = \mathcal{O}_{n \rightarrow \infty}(n^{-\alpha})$ (standard big O notation) for some $\alpha > 0$.

We will occasionally abuse terminology and speak of a *stable* minimum-norm interpolating neural network $\mathcal{N}(\cdot; \hat{\theta})$, meaning that algorithm $\mathcal{T}$ generating $\hat{\theta}$ is stable.

2.2. Kernel Theory and Random Matrix Theory

Kernels. A (continuous) kernel on $\mathbb{R}^{d_0}$ is a positive semi-definite symmetric (continuous) function $K : \mathbb{R}^{d_0} \times \mathbb{R}^{d_0} \mapsto \mathbb{R}$. By Mercer’s theorem (Minh et al. 2006), given a data distribution which is a Borel measure $\mathbf{x} \sim p(\mathbf{x})$ with compact support on $\mathbb{R}^{d_0}$, there exist unique countable collections of Mercer’s eigenvalues $(\lambda_i)_{i \in \mathbb{N}}$ and eigenfunctions $(\varphi_i)_{i \in \mathbb{N}}$ such that $K(\mathbf{x}, \mathbf{x}') = \sum_i \lambda_i \varphi_i(\mathbf{x}) \varphi_i(\mathbf{x}')$ and (φ_i) are orthonormal w.r.t. the data distribution: $\mathbb{E}_{\mathbf{x} \sim p(\mathbf{x})} (\varphi_i(\mathbf{x}) \varphi_j(\mathbf{x})) = \delta_{i,j}$ for all i, j . By Riesz’s theorem, there exists a Hilbert space $\mathcal{H}$ and a *feature map* $\phi : \mathbb{R}^{d_0} \mapsto \mathcal{H}$ such that $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle_{\mathcal{H}}$. Kernel regression amounts to linear regression in the corresponding Hilbert space: when trained on data $\mathbf{X}, \mathbf{y}$, the prediction on a new point $\mathbf{x}^*$ is given by $\mathbf{k}_{\mathbf{x}^*, \mathbf{X}}^T \mathbf{K}_{\mathbf{X}, \mathbf{X}}^{-1} \mathbf{y}$ where the vector $\mathbf{k}_{\mathbf{x}^*, \mathbf{X}}$ is given by $(\mathbf{k}_{\mathbf{x}^*, \mathbf{X}})_i = K(\mathbf{x}^*, \mathbf{x}_i)$ and the kernel matrix $\mathbf{K}_{\mathbf{X}, \mathbf{X}}$ is given by $(\mathbf{K}_{\mathbf{X}, \mathbf{X}})_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$. Although the kernel’s eigenfunctions exhibit the described structure, the *spectral universality assumption* (SUA) is commonly adopted. The SUA posits that, as the sample size increases, the eigenfunctions can be approximated by independent Gaussian entries: $\varphi_i(\mathbf{x}_j) \sim N(\mu_K, \sigma_K^2)$, where μ_K and σ_K^2 depend on the kernel K and the data distribution $p(\mathbf{x})$, but not on specific instances i and j . The SUA works well in practice (Karoui 2010; Cheng & Singer 2013; Fan & Montanari 2015; Liu et al. 2021; Simon et al. 2023; Lu & Yau 2023; Schröder et al. 2023), and research focuses on identifying conditions under which it holds.

The feature map ϕ naturally connects neural networks to kernels, where inner products of learned representations (empirical feature maps) induce kernels on the input space, an idea further developed in Wilson et al. (2016), which proposes learning the kernel itself by composing a deep neural network feature map with a Gaussian process to obtain flexible, data-adaptive kernels.

Random matrix theory. Random matrix theory (Wigner 1955; Livan et al. 2018) is the study of the spectral distributions of large matrices of random variables. The spectral measure F_n for a given matrix, with eigenvalues λ_i , is given, for $x \in \mathbb{R}$, by $F_n(x) := \frac{1}{n} \sum_{i=1}^n \delta_{\lambda_i}(x)$, where $\delta_{\lambda_i}(x)$ represents the Dirac measure centered at the eigenvalue λ_i . When the matrix is random, the spectral measure becomes a random measure, called the empirical spectral distribution. Our focus lies in studying weak convergences (convergences in distribution) of the spectral measures towards nonrandom measures (Geronimo & Hill 2003). A useful tool to manipulate spectral measures is the Stieltjes transform; for a measure F supported on the real interval I , the Stieltjes transform is given as follows for each $z \in \mathbb{C} - I$: $S_F(z) = \int_I \frac{dF(\lambda)}{\lambda - z}$. There is a one-to-one correspondence between Borel measures and their Stieltjes transforms, as per the inversion formula: $F(x) = \lim_{y \rightarrow 0^+} \frac{1}{\pi} \text{Im}(S_F(x + iy))$ for $x \in I \setminus \{0\}$, where Im denotes the imaginary part of a complex number. Pointwise convergence in their Stieltjes transforms ensures weak convergence of measures. We rely on a famous result in random matrix

2. Preliminaries

theory. Consider $\mathbf{W} \in \mathbb{R}^{d \times n}$, a random matrix with i.i.d. entries drawn from $N(0, \frac{1}{d})$ and Ψ a nonrandom positive semi-definite matrix. Suppose that $\Psi \in \mathbb{R}^{n \times n}$ has a limiting spectral measure ρ as $n \rightarrow \infty$, and let $n, d \rightarrow \infty$ with fixed ratio $\alpha := \frac{n}{d}$, then the random matrix $\Psi^{1/2} \mathbf{W}^T \mathbf{W} \Psi^{1/2}$ has a limiting nonrandom spectral measure $\rho_{MP}^\alpha \boxtimes \rho$. The Marchenko-Pastur map of ρ , denoted $\rho_{MP}^\alpha \boxtimes \rho$, is defined by the Stieltjes transform S solving the Marchenko-Pastur fixed-point equation (Marchenko & Pastur 1967; Fan & Wang 2020):

$$S(z) = \int \frac{1}{x(1 - \alpha - \alpha z S(z)) - z} d\rho(x). \quad (2.2)$$

It also appears in the free probability literature as the free multiplicative convolution between the probability measures ρ_{MP}^α and ρ (Mingo & Speicher 2017). When considering the specific case where $\Psi = \mathbf{I}_n$ (identity matrix of size n), then ρ represents the Dirac measure at 1 and we recover the well-known Marchenko-Pastur distribution, denoted as ρ_{MP}^α . Furthermore, we denote as $\rho_{MP}^\alpha \boxtimes^\ell \rho := \rho_{MP}^\alpha \boxtimes (\dots(\rho_{MP}^\alpha \boxtimes \rho))$ the composition of ℓ successive Marchenko-Pastur maps.

2.3. Bayesian Neural Networks and Gaussian Processes

BNNs. We assume a *prior distribution* over parameters $(\Theta, \mathbf{W}^L)$ with weights sampled i.i.d. from the Gaussian distribution $N(0, \frac{1}{d})$ and biases sampled i.i.d. from $N(0, 1)$; this yields a density $p(\Theta, \mathbf{W}^L)$ that is a product of Gaussian densities. The *posterior distribution* given the training data is given by Bayes' rule:

$$p(\Theta, \mathbf{W}^L | \mathbf{X}, \mathbf{y}) = p(\Theta, \mathbf{W}^L) \frac{p(\mathbf{y} | \mathbf{X}, \Theta, \mathbf{W}^L)}{p(\mathbf{y} | \mathbf{X})},$$

where $p(\mathbf{y} | \mathbf{X}, \Theta, \mathbf{W}^L)$ is the *likelihood* of the data given a set of parameters, and $p(\mathbf{y} | \mathbf{X}) = \int p(\mathbf{y} | \mathbf{X}, \Theta, \mathbf{W}^L) p(\Theta, \mathbf{W}^L) d\Theta d\mathbf{W}^L$ is the *marginal likelihood* (or *Bayesian evidence*). We assume Gaussian likelihoods, i.e. :

$$p(\mathbf{y} | \mathbf{X}, \Theta, \mathbf{W}^L) \sim N(\mathbf{y} | \mathbf{W}^{L^T} \phi(\Theta, \mathbf{X}), \phi(\Theta, \mathbf{X})^T \mathbf{W}^L \mathbf{W}^{L^T} \phi(\Theta, \mathbf{X})),$$

which represents the probability density of sampling $\mathbf{y}$ from a random network $\mathcal{N}_{\Theta, \mathbf{W}^L}$ with Gaussian outputs.

Calculating the posterior distribution, which is the essence of BNN training, is analytically intractable and remains a core challenge (Gal 2016). In practice, the posterior distribution is estimated via variational inference (Blei et al. 2017) or Monte-Carlo simulation methods (Rasmussen 1995).

Given the posterior distribution, the predictor defines a distribution over functions $\mathcal{N}_{\Theta, \mathbf{W}^L}$ with $(\Theta, \mathbf{W}^L) \sim p(\Theta, \mathbf{W}^L | \mathbf{X}, \mathbf{y})$. The mean-squared generalisation error is defined for any new point $(\mathbf{x}^*, y^*)$ as the expectation over the predictor error:

$$\mathbb{E}_{(\Theta, \mathbf{W}^L) \sim p(\Theta, \mathbf{W}^L | \mathbf{X}, \mathbf{y})} \left((y^* - \mathcal{N}_{\Theta, \mathbf{W}^L}(\mathbf{x}^*))^2 \right).$$

Only the mean and variance of the predictor are needed to calculate it.

2. Preliminaries

Gaussian processes and NNGPs. A GP g over a space $\mathbb{R}^{d_0}$ is a random scalar field such that its evaluation at any collection of finitely many points $(g(x_1), \dots, g(x_P))$ follows a multivariate Gaussian distribution. A GP is determined by a mean function $\mu : \mathbb{R}^{d_0} \mapsto \mathbb{R}$, and a covariance function $K : \mathbb{R}^{d_0} \times \mathbb{R}^{d_0} \mapsto \mathbb{R}$, which describe respectively the mean of the Gaussian distribution at each point and the covariance between the Gaussians at any two points. The covariance function of a GP is a kernel (Rasmussen & Williams 2006). We note $g \sim \mathcal{GP}(\mu, K)$. *Gaussian process regression* consists in performing Bayesian inference using a Gaussian process as the prior distribution over functions, usually assuming an *additive noise model* $y = g(\mathbf{x}) + \epsilon$. The prediction distribution of GP regression with prior $\mathcal{GP}(0, K)$ and noise model $\epsilon \sim N(0, \sigma_\epsilon^2)$ trained on the data $\mathbf{X}, \mathbf{y}$ is given, on a new point $\mathbf{x}^*$, by $\mathcal{N}(\mathbf{k}_{\mathbf{x}^*, \mathbf{X}}^T (\mathbf{K}_{\mathbf{X}, \mathbf{X}} + \sigma_\epsilon^2 \mathbf{I}_n)^{-1} \mathbf{y}, K(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}_{\mathbf{x}^*, \mathbf{X}}^T (\mathbf{K}_{\mathbf{X}, \mathbf{X}} + \sigma_\epsilon^2 \mathbf{I}_n)^{-1} \mathbf{k}_{\mathbf{x}^*, \mathbf{X}})$. The mean prediction of GP regression with no noise model coincides with the prediction of kernel regression with the same kernel.

Applying successively the Central Limit Theorem to each layer, the infinite-width limit of (2.1) yields a GP, called the *Neural Network Gaussian Process* (NNGP). If we let the width $d \rightarrow \infty$, the $\mathbf{h}_i^L \sim \mathcal{GP}(\mu^L, K^L)$ are independent and defined inductively by layers as follows for all $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^{d_0}$ and each $\ell \in 1, \dots, L$. First, $\forall \ell \mu^\ell(\mathbf{x}) = 0$ and $K^0(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$. Then, $\mathbf{h}_i^{\ell-1} \sim \mathcal{GP}(\mu^{\ell-1}, K^{\ell-1})$ and the covariance functions $K^\ell(\mathbf{x}, \mathbf{x}')$ are given by $\mathbb{E}_{\mathbf{h}_i^{\ell-1} \sim \mathcal{GP}(\mu^{\ell-1}, K^{\ell-1})} \left(\sigma^\ell(\mathbf{h}_i^{\ell-1}(\mathbf{x})) \sigma^\ell(\mathbf{h}_i^{\ell-1}(\mathbf{x}')) \right)$. The covariance function K^L is the NNGP kernel (Daniely et al. 2016), denoted as $K^L = K_{\text{NNGP}}$. Infinite-width limits involve various subtleties (Matthews et al. 2018), and we follow the approach in Lee et al. (2018) where infinite limits are taken sequentially. In this limit, the number of examples n and the input dimension d_0 remain fixed. Furthermore, we will investigate more comprehensive limits where n, d , and d_0 all tend to infinity simultaneously, first while maintaining constant and bounded ratios $\alpha = \frac{n}{d}$ and $\alpha_0 = \frac{n}{d_0}$ (linear-width regime), then while $n \propto d \cdot d_0$, thus $\alpha \rightarrow \infty$ and $\alpha_0 \rightarrow \infty$ (sublinear-width regime).

Part I.

Explainability of Neural Networks

3. Overview of Explainability and Interpretability

The terms *explainability* and *interpretability* broadly refer to methods that make the behavior or decisions of complex machine learning models understandable to humans. This chapter reviews the main categories of techniques for explaining black-box neural models, drawing connections between different approaches and highlighting recent advances. Historically, some approaches focus on designing intrinsically interpretable models (such as decision trees or linear models) that are transparent by design (Lipton 2018; Doshi-Velez & Kim 2017). However, much of the recent work centers on post-hoc methods that aim to explain trained neural network models without sacrificing their accuracy. We organize the discussion into several major threads: (1) *Rule-based methods*, which derive human-readable rules to characterize model behavior; (2) *Attribution-based methods*, which assign credit or importance to input features; (3) *Perturbation-based methods*, which explore “what-if” changes to inputs; (4) *Example-based methods*, which explain by reference to specific examples from the data; (5) *Abductive and contrastive methods*, which provide minimal sufficient or counterfactual feature sets as explanations; and (6) *Mechanistic interpretability*, an emerging approach that seeks to reverse-engineer the internal mechanisms and representations learned by neural networks. These categories sometimes overlap or complement each other, and together they form a broad toolkit for explainable AI. In the following sections, we survey each category, explain the motivations and techniques involved, and discuss representative work and open challenges.

3.1. Rule-based methods

One line of work seeks to explain model behavior by extracting human-comprehensible rules. The appeal of rule-based explanations is that they can provide precise, symbolic conditions under which certain predictions are made, which are easy for humans to interpret and verify. For example, a rule might state that “if $x_1 > 5$ and x_3 is True, then the model outputs class A,” providing a clear rationale in logical terms.

Early rule extraction methods includes Trepan and related methods that train decision trees on a network’s soft labels (Craven & Shavlik 1996), and variants that transfer a deep learning model into a (soft) decision tree (Frosst & Hinton 2017). A large literature then formalised optimal or certifiably near-optimal rule learning that are applicable to *surrogate models* for neural networks (i.e. models trained to mimic neural network outputs): RuleFit learns sparse linear models over gradient-boosted rules (Friedman & Popescu 2008); Bayesian Rule Lists and Falling Rule Lists construct short probabilistic or monotone lists with Bayesian or combinatorial guarantees (Letham et al. 2015; Wang et al. 2017b); CORELS learns provably optimal (and succinct) rule lists via branch-and-bound with certifiable optimality gaps (Angelino et al. 2017); SLIM learns integer-sparse scoring systems optimised for interpretability and accuracy (Ustun & Rudin 2017); and

3. Overview of Explainability and Interpretability

Shih et al. (2018) proposed a symbolic approach for explaining Bayesian network classifiers by compiling them into ordered decision diagrams.

MaxSAT-based pipelines can compute provably correct explanations for tree ensembles (Ignatiev et al. 2022). At the same time, a number of papers have highlighted limitations of existing post-hoc rule methods—e.g., lack of robustness to distribution shift, insufficient completeness or minimality, or scalability issues—arguing for stronger logical contracts and end-to-end faithfulness metrics (Darwiche 2020; Blanc et al. 2021; Marques-Silva 2022). Recent work has shown that naïve rule-learning pipelines can be unfaithful even when their surrogates appear accurate, and proposed extraction procedures with explicit faithfulness guarantees (Cucala et al. 2022a; Cucala et al. 2022b), that is, applying the model or the rules extracted from the model to a given dataset yields exactly the same outputs, thus ensuring the extracted logical rules closely match the model’s true decision boundary. Note that rule-based methods have been particularly successful for neural models applied to knowledge graphs (Trouillon et al. 2016; Ji et al. 2022; El Harzli et al. 2023a), which is an application domain especially amenable to logical rule extractions. Overall, the field has converged on a set of desiderata—fidelity/faithfulness to the target model, sufficiency/necessity, minimality and compactness (readability). Enabled by solver-backed learning (SAT/MaxSAT/MILP) for optimal rule discovery, these criteria make rule-based explanations one of the few families with verifiable guarantees.

In summary, (logical) rule-based interpretability approaches aim at generating human-readable explanations in the form of decision rules. These approaches achieve interpretability by imposing a discrete, transparent structure on model behavior. A challenge, however, is scalability: extracting exact logical rules for large deep networks can be computationally intractable in general (many such problems are NP-hard or worse). Research therefore explores either approximations – e.g. extracting rules for a simplified surrogate model – or restricted domains where the complexity is manageable (such as monotonic models or bounded-depth networks). Nonetheless, by combining formal guarantees with interpretability, (logical) rule-based methods provide one of the most rigorous forms of explanation, bridging the gap between sub-symbolic learning and symbolic reasoning.

3.2. Attribution-based Methods

Attribution-based interpretability methods attempt to explain a particular prediction by attributing importance scores to the input features. In other words, they ask: which features (pixels, words, or input variables) contributed the most to this output? These methods produce a saliency map or set of feature importances that highlight what parts of the input the model was most sensitive to for the given prediction. A classic example is the gradient heatmap proposed by Simonyan et al. (2014), which uses the gradient of the output with respect to each input dimension as a measure of that input’s influence. These techniques, often called *saliency maps*, effectively show how small changes in each pixel would affect the score of the predicted class, thus indicating which pixels the network “looks at”. Extensions of this idea soon followed: *Guided Backpropagation* and *DeconvNet* applied modifications to the gradient propagation to produce sharper visualisations of important image regions; *Layerwise Relevance Propagation* (LRP) takes

3. Overview of Explainability and Interpretability

a related approach, propagating the prediction backward through the network with a conservation rule to distribute the “relevance” to each input pixel (Bach et al. 2015). These gradient-based attribution methods are fast (requiring only a backward pass) and broadly applicable to differentiable models, making them popular for explaining deep neural networks, particularly in computer vision.

A more principled attribution approach came with *Integrated Gradients* by Sundararajan et al. (2017). Integrated Gradients (IG) address the issue that raw gradients can be noisy or zero for saturated neurons. IG computes the accumulated gradient of the output as the input is smoothly interpolated from a baseline (e.g. a blank image) to the actual input, effectively averaging gradients along that path. The result is an importance score for each feature that satisfies desirable axioms (such as sensitivity and completeness) ensuring the attributions are consistent with changes in the prediction. Around the same time, Shrikumar et al. (2017) proposed *DeepLIFT*, an approach to propagate differences from a reference activation, which also yields feature importance scores that address some shortcomings of plain gradients. Another influential method is *SHAP* (SHapley Additive exPlanations) by Lundberg & Lee (2017), which uses game-theoretic Shapley values (Shapley 1953) to attribute the prediction difference (from a baseline) to features. SHAP provides a unified framework connecting several existing methods (including game-theoretic Shapley values, saliency maps, and DeepLIFT) and offers theoretical guarantees of fairness in how credit is assigned to features. In practice, SHAP estimation for deep networks relies on sampling or linear approximations, but it remains popular due to its solid theoretical foundation and consistent interpretation across models.

Attribution methods have also been developed for specific modalities. *Grad-CAM* (Gradient-weighted Class Activation Mapping) extends attribution to CNN feature maps: it finds important regions in an image by using gradients to weight the activation maps in the last convolutional layer, producing a coarse localisation heatmap for the prediction (Selvaraju et al. 2017). Grad-CAM and its variants enable visual explanations for image and video models by highlighting where the network is focusing its attention for a given class. Similarly, in NLP, gradients or attention weights are often used to attribute importance to words in an input sequence. Attention-based models sometimes use the attention scores as a form of explanation, though it has been debated whether attention weights truly reflect importance. Nonetheless, techniques like Integrated Gradients are model-agnostic and can be applied to text embeddings to find which words most affect a classification.

While feature attributions are intuitive and easy to obtain, they are not without issues. One major concern is their reliability. Adebayo et al. (2018) famously showed that some saliency methods were insensitive to model re-training or randomisation, raising the possibility that they sometimes reflect dataset priors or model structure rather than the true reasoning for a specific prediction. These “sanity checks” demonstrated that certain popular methods (like Guided Backprop) produced similar explanations even for a randomly initialized network, implying that the visuals were largely independent of the model’s actual learned weights. This spurred development of more robust attribution techniques, such as Full-Gradient representations (Srinivas & Fleuret 2019), which enrich attributions in input space with contributions from all layers, as well as stricter evaluation guidelines, including input invariance tests proposed by Kindermans et al. (2019). Another issue is that attributions can be fragmented or noisy for deep

3. Overview of Explainability and Interpretability

networks with highly distributed representations – a single prediction might truly depend on a complex combination of features such that no small subset is “the key.” As a result, attribution maps should be interpreted with caution. Research has proposed smoothing or averaging methods, like *SmoothGrad* (adding noise and averaging saliency maps) to reduce noise and highlight consistent signals (Smilkov et al. 2017). Despite these challenges, attribution-based explanations remain a staple in explainable AI because they are fast, intuitive, and local: they give an immediate sense of “what input caused this output” for each individual case. They are often the first tool applied when trying to debug or audit a neural network’s behavior, as they can flag spurious correlations (e.g. a medical image model focusing on the marker in a corner of X-rays instead of anatomy) or help users gain trust by showing alignment with human-understandable features. Recent work continues to improve attribution methods – for example by combining gradients with perturbation analysis (Xu et al. 2020; Erion et al. 2019; Ancona et al. 2018), or by enforcing desirable properties like continuity and implementation invariance (Wang et al. 2020; Lin et al. 2023)– but the fundamental idea remains assigning credit to input features as an explanation for predictions.

3.3. Perturbation-based Methods

Perturbation-based explanation methods explore the model’s behavior by tweaking input features and observing the effect on the output. Instead of directly computing an importance score (as in gradients), these methods answer questions like: if this part of the input were different, would the prediction change? By studying such counterfactual variations, they identify which portions of the input are crucial for the model’s decision. One simple approach is the occlusion test introduced by Zeiler & Fergus (2014), where one systematically occludes or masks out different regions of an image and notes how the prediction score drops. Regions whose removal significantly decreases the confidence are deemed important. This gave some of the first insights into CNNs by revealing, for example, that certain patches of an image (like an object’s head or distinctive texture) greatly influence the classification. Similarly, in text, one can remove or replace words and see if the model still predicts the same class; important words are those whose removal changes the prediction. Occlusion methods are easy to interpret but can be slow if done one feature at a time, especially for high-dimensional inputs.

To make perturbation analysis more efficient and precise, local surrogate modeling techniques have been developed. The most notable is *LIME* (Local Interpretable Model-agnostic Explanations) by Ribeiro et al. (2016). LIME generates many perturbed versions of a given input (e.g. turning some super-pixels of an image off, or removing words from a sentence) and observes the model’s outputs. It then fits a simple interpretable model (like a sparse linear model) to these input-output samples in the vicinity of the original input. The coefficients of this local surrogate model serve as an explanation: they indicate which features positively or negatively influence the prediction in that neighborhood. For instance, LIME might produce a small list of weighted words that best explain a text classifier’s decision on a specific document. The key advantage is that LIME makes no assumptions about the global model – it only requires the ability to query the model on perturbed inputs. This makes it applicable to any classifier, even non-differentiable ones or black-box services. However, LIME’s explanations are

3. Overview of Explainability and Interpretability

only locally faithful (near that input); they are not guaranteed to describe the model’s behavior elsewhere or under larger changes.

The influential *SHAP* method, already mentioned, can also be viewed as perturbation-based: it considers all possible subsets of features (perturbing the rest to a reference value) and uses the model outputs to compute Shapley values for feature contribution (Lundberg & Lee 2017). Exact Shapley computation is combinatorially expensive, so practical SHAP methods rely on approximations or built-in linearity for certain model types (like decision trees, where exact Shapley values can be computed efficiently). SHAP connects perturbation analysis with a rigorous game-theoretic framework, ensuring each feature’s credit is the average marginal contribution across all coalitions.

Beyond identifying important features, perturbation approaches are naturally suited for generating *counterfactual explanations*. A counterfactual explanation is a what-if scenario: “What is a small change to the input that would have resulted in a different (often desired) outcome?” For example, for a loan application rejected by a model, a counterfactual might be: “if the applicant’s income were \$5,000 higher, the loan would be approved.”. Wachter et al. (2018) introduced this concept in the context of algorithmic recourse, arguing that such counterfactuals are actionable explanations for individuals. In the image domain, Goyal et al. (2019) demonstrated *Counterfactual Visual Explanations*, where they identify which portion of an image, if altered, would most change the prediction – essentially the visual analog of a counterfactual. Their method, termed “minimal plausible perturbation,” uses a generator to produce a new image where a salient region is modified (e.g. replacing a part of an image with a different context) such that the model’s classification flips to a different category. Similarly, in graph neural networks, perturbation-based techniques have been used to explain predictions by removing or rewiring parts of the graph: for instance, cutting the most influential edges or nodes. Lucic et al. (2022) propose *CF-GNNExplainer*, which finds a small modification to the input graph (like deleting a few edges) that changes the GNN’s output, thus highlighting the subgraph that was crucial for the original prediction. These approaches provide contrastive insight – they not only show what was important, but also how things could be for a different decision to occur.

Perturbation methods often strive to produce sparse and interpretable changes, aligning with human notion of explanation. Fong & Vedaldi (2017) introduced the idea of learning a perturbation mask – essentially training a small network to selectively blur or erase parts of an image to minimize the classifier’s confidence. The resulting mask indicates which pixels must be “turned off” to significantly affect the prediction, thus pointing to the evidence the model relied on. Notably, their method was model-agnostic and optimisation-based, treating explanation as a learned perturbation that covers as few pixels as necessary while remaining meaningful (it causes a big change in output). This approach blurs the line between attribution and perturbation – it effectively finds the most critical features by removing them. Subsequent works have extended this idea using generative models to ensure perturbations stay in the data manifold (so that counterfactual images look realistic, for example).

In summary, perturbation-based methods offer a very intuitive form of explanation by answering “what if?” and “why not?” questions. They tend to be model-agnostic (since one can perturb inputs to any model) and flexible (able to target different desired outcomes or constraints on the explanation). Their drawbacks include computational

3. Overview of Explainability and Interpretability

cost – evaluating many perturbations or solving complex optimisation problems can be slow – and the difficulty of accounting for correlated features (removing one feature at a time may ignore that features act in combinations). Despite these challenges, perturbation methods are fundamental in explainable AI, as they align closely with human reasoning: we often understand a phenomenon by mentally altering conditions and observing outcomes, which is exactly what these methods emulate.

3.4. Example-based Methods

Example-based explanations justify a model’s prediction by referring to specific instances, either from the training data or from some case library, that ground the model’s decision in familiar terms. The underlying intuition is that humans often understand a concept by analogy to known examples (e.g., “this tumor looks similar to these other cases that were malignant”). There are several flavors of example-based interpretability: *prototypes*, *criticisms*, *influences*, and *case-based reasoning*, among others.

Prototypes and case-based reasoning: One approach is to have the model explicitly learn prototype representations during training, as seen in the ProtoPNet model by [Chen et al. \(2019\)](#). In ProtoPNet, the network’s intermediate layers include a set of prototype vectors that are meant to correspond to representative parts or patterns of each class (for instance, a prototype might represent a typical texture of a bird’s wing for the bird class). When making a prediction, the model finds the prototypes most similar to the input’s features and bases its decision on those similarities. The explanation can then be given in terms of “the input was classified as X because it contains a patch that is similar to this prototype patch (from a training image) that is characteristic of class X.” This yields a tangible explanation: the user can inspect the prototype image patch which serves as evidence for the model’s reasoning. The concept of case-based reasoning through prototypes harks back to the idea of nearest neighbors, but instead of raw nearest neighbors, the model learns an explicit set of prototypes (like exemplars) that summarize the dataset. This improves interpretability because the number of prototypes is limited and each is chosen for being representative.

Nearest neighbors and examples from the training set: A simpler form of example-based explanation is to retrieve a few training examples most similar to the input in the feature space or in terms of model behavior. For instance, a support vector machine classifier could be explained by showing a user the support vectors (critical borderline cases). For deep models, one can find the training instances that have the highest *influence* on a given prediction. [Koh & Liang \(2017\)](#) introduced Influence Functions for this purpose. Influence functions, originally from robust statistics, approximate how the model’s parameters would change if a particular training point were upweighted, and thereby estimate how that change would affect the prediction in question. Using this, one can rank training samples by their influence on the test prediction: for example, “this specific training image contributed most to the model classifying the new image as a cat”. This provides an explanation in terms of past cases, which is often very insightful – if the top influential training example seems unrelated or mislabeled, it could indicate a spurious correlation or error that the model has picked up. Influence functions were one of the first methods to attribute responsibility to training data points rather than features, giving a different perspective on “why” a model made a

3. Overview of Explainability and Interpretability

decision (because it had seen certain examples during training that nudged it toward that decision). Subsequent works have built on this by developing more efficient influence approximations (since naive influence calculation can be expensive for deep networks) and by introducing tools like *TracIn*, which tracks the model’s loss on the example through training to gauge influence (Pruthi et al. 2020), and *Representer Point Selection*, which decomposes predictions in kernel machines as a weighted sum of training embeddings (Yeh et al. 2018). All these methods share the goal of connecting a model’s behavior to specific training cases.

Counterexamples and criticisms: In addition to prototypes (typical examples), sometimes it is useful to show atypical examples or “criticisms.” These are instances that the model finds puzzling or that do not fit well with any learned prototype, which can alert users to the model’s blind spots. For instance, one might ask: “which training example, if removed, would most increase the confidence of this prediction?” – that could surface an example that was holding the model back (perhaps an outlier or contradictory example). While this is a bit different from explaining a single prediction, it provides a broader interpretability: understanding the scope and limits of the model’s learned representation by examining which examples are well-represented vs. which are not. (Kim et al. 2016) proposed using *MMD-critic* to pick prototypes and criticisms: prototypes for each class and criticisms that highlight aspects of the data not covered by prototypes.

Example-based explanations are often appreciated by domain experts because they allow direct inspection and comparison. In sensitive applications like healthcare or law, showing analogous past cases (especially if they come with human-understandable annotations) can greatly increase trust in the model’s decision. For example, a diagnostic system might retrieve similar past patients’ records to justify a new diagnosis, which resonates with a doctor’s case-based reasoning. Moreover, example-based methods can sometimes uncover data issues: if the nearest neighbors for a surprising prediction are all from a certain cluster of anomalous data, it might hint that the model latched onto an artefact.

However, there are some caveats. The notion of “similarity” in high-dimensional learned representations might not always align with human similarity – a nearest neighbor in latent space might look quite different to a human if the model’s features are unintuitive. Efforts to ensure that the similarity metric is human-meaningful (perhaps by using disentangled or interpretable embeddings) are ongoing. Another challenge is dataset privacy and portability: showing actual training examples may not be possible due to confidentiality, and it ties the explanation to the availability of those examples. Despite these challenges, example-based methods remain a vital part of the interpretability toolbox. They complement feature attribution by providing concrete context: rather than saying “feature x_5 was important,” an example-based explanation can say “this instance was handled similarly to instance #123 in the training set,” which can often be a more compelling justification.

3.5. Abductive and Contrastive Methods

Abductive and contrastive explanation methods are related to previous methods but are of special interest to this thesis, because they emphasize formal guarantees. Similarly

3. Overview of Explainability and Interpretability

to perturbation methods, they also seek to answer “Why?” and “Why not?” questions about a model’s prediction, but a key requirement is that the resulting explanation must be minimal. Specifically, they identify the smallest subset of input features that either (a) are sufficient to produce the prediction (in the case of abductive explanations, often called “sufficient reasons” or “anchors”), or (b) if changed, would alter the prediction (in the case of contrastive explanations, often corresponding to counterfactual conditions). These methods are rooted in classic notions of explanation in philosophy and logic, where an explanation often consists of the minimal conditions that imply an outcome (for a “Why?” question) or the minimal changes to achieve a different outcome (“Why not?”).

Formally, one can define an abductive explanation for a prediction $f(\mathbf{x}) = y$ as a subset of features S of $\mathbf{x}$ such that if we fix those features to their values in $\mathbf{x}$, the model would still predict y regardless of the other features. In other words, the features in S alone are sufficient to guarantee the same outcome y . An explanation that satisfies this and is minimal (you cannot drop any feature from S without losing sufficiency) is often called a *prime implicant* or *Minimal Sufficient Reason*. For example, for a loan approval model that outputs “approved” for a particular applicant, an abductive explanation might be “income > \$50K and no past defaults” if those conditions alone would always lead to approval for that applicant in the model. A contrastive explanation, on the other hand, is often given as a *counterfactual*: the smallest set of features C such that if those features were changed (often to some alternative values), the model’s prediction would change to a different specified outcome (e.g., loan denied in this case). This is akin to a “minimal sufficient reason for not- y ” or a “minimal tweak”. In the loan example, a contrastive explanation might be “if income were \$30K (instead of \$55K), the application would be denied” – highlighting why it was not denied by pointing out the crucial difference.

Several works have aimed to compute such explanations with formal guarantees. Ignatiev et al. (2019) use a SAT-solver-based approach to compute abductive explanations for individual predictions in classifiers by reducing the problem to a satisfiability query: find a hitting set of input literals that satisfies the classifier’s decision logic. This yields a guarantee of minimality by design (the solver searches for minimal unsatisfiable cores or minimal satisfying assignments depending on formulation). Similarly, Shih et al. (2018), already mentioned in rule-based methods, can be viewed as an abductive method, effectively extracting logical conditions on feature variables that suffice for the classification of Bayesian networks. Their approach enumerates combinations of feature instantiations that lead to the target class with high probability, again ensuring minimality, and producing an explanation like “if (Feature A = val1) AND (Feature B = val2) then outcome = Y, regardless of other inputs.” These explanations can be seen as locally sufficient rules.

On the contrastive side, algorithms for finding counterfactuals can be cast as search problems in the input space. Some use heuristic or gradient-based search to find the closest counterfactual (as discussed earlier under perturbation methods). But when one demands combinatorial minimality (smallest subset of features to tweak), it often requires combinatorial search or optimisation. For Boolean or categorical inputs, one can use techniques like *mixed-integer programming* (MIP) or *branch-and-bound* to find minimal feature flips that change the decision. For instance, an algorithm might search for the smallest number of input bits that need to be flipped from 0 to 1 (or vice

3. Overview of Explainability and Interpretability

versa) such that the classifier’s output changes – this number is sometimes called the Hamming distance of the counterfactual. In continuous domains, one can include an L_0 norm in the optimisation objective to count features changed, though that becomes a non-convex problem.

A practical and user-friendly incarnation of abductive explanations is the notion of *anchors*, proposed by Ribeiro et al. (2018). An “anchor” is essentially a high-precision rule: a condition on a few features that almost always leads to the same prediction. Ribeiro’s algorithm greedily grows a rule by testing it on perturbations of the input and ensures that when that rule holds, the model does not switch class (with high probability). For example, an anchor for a text classification might be that the presence of the words “not recommended” in a review anchors the model’s prediction of a negative sentiment – because any perturbation of the rest of the text still yields negative as long as those words are present. Anchors do not guarantee global sufficiency (they are probabilistic guarantees under a local distribution of perturbations), but they are intuitive and align with the idea of a salient reason you might give: “The model predicted this is spam because it contains the phrase ‘win a free prize’, which is a strong indicator.” This connects abductive explanations to more intuitive linguistic or visual patterns as well.

Contrastive explanations often connect with the psychological viewpoint that humans think in contrasts: we ask “why X rather than Y?” A good example is *contrastive feature highlighting*, where alongside the normal explanation one might highlight what could make the result different. Dhurandhar et al. (2018) explicitly tackled this with the concept of *pertinent positives* and *pertinent negatives*. A pertinent positive is a minimal set of features that must be present for the prediction to hold (similar to abductive sufficient reason), whereas a pertinent negative is a feature (or set of features) that, if it were present (but is currently absent in the input), would change the prediction – effectively pointing out something that the instance lacks which, if it had, would flip the outcome. In image terms, a pertinent positive might be a particular shape in a certain region that the image has and that is crucial for the classification, while a pertinent negative might be a shape that, if it were in the image, would have caused a different classification (i.e. “the model would have called this a cat if it had pointy ears present, but those are absent – hence it’s a dog”). Dhurandhar et al. (2018) use an autoencoder-driven approach to find a close counterfactual (pertinent negative) that shares most of the original characteristics but lacks the predicted class’s features, thus providing a very tangible contrastive explanation.

The strength of abductive and contrastive explanations lies in their minimality and clarity: they do not overload the user with all features’ contributions (as attribution methods might), but rather focus on a small key set. They often translate nicely into human language or simple decision rules. They also have a close connection to robustness – e.g. the size of a minimal counterfactual perturbation is essentially a measure of how robust the prediction is to input changes. A very small counterfactual change means the decision could easily have been different (low robustness), whereas if the counterfactual requires a large or multiple feature change, the decision is more stable. This connection has been formalized in works like Wachter et al. (2018) and others who note that minimal explanation size is inversely related to confidence or margin.

3. Overview of Explainability and Interpretability

On the flip side, computing truly minimal explanations is computationally challenging for general models, as noted. Many approximation algorithms or relaxations are used in practice. There’s also the question of multiple explanations: there could be several distinct minimal feature sets that suffice for a prediction – which one should we present? Some approaches enumerate a diverse set of explanations to give a more complete picture. Moreover, these methods typically assume a fixed model and explain the given input – they don’t necessarily clarify how the model works in general, just how this particular decision can be accounted for by a subset of features. Therefore, they are often combined with global interpretability methods for a fuller understanding. Nonetheless, abductive (“sufficient reason”) and contrastive (“counterfactual condition”) explanations are extremely useful in domains where users ask “what was the critical factor in the decision?” or “what could I change to get a different outcome?”, making these methods highly relevant for actionable interpretability and user recourse.

3.6. Mechanistic Interpretability

While the previous sections focus on *external* explanations of a model’s inputs and outputs, *mechanistic interpretability* (sometimes called “circuits-level” interpretability) dives inside the model to understand the actual computations and representations that give rise to its behavior. The guiding question here is: “How does the neural network implement the function that it has learned? What algorithms or mechanisms has it internally invented?” This approach treats a trained network as an object of study, where the goal is to reverse-engineer its internal workings – analogous to how one might disassemble a compiled program to see how it works. Mechanistic interpretability is particularly concerned with understanding *generalisation*: how the network’s internal structures enable it to solve a range of inputs or tasks beyond the training examples. In fact, [Sharkey et al. \(2025\)](#) identify mechanistic interpretability with the study of the mechanisms underlying neural network generalisation, distinguishing it from surface-level input-output correlations.

One strand of mechanistic interpretability research looks at *feature representations* in hidden layers. Techniques such as *feature visualisation* aim to find an input pattern that maximally activates a particular neuron or a specific circuit in the network ([Olah et al. 2017](#)). By optimizing an input image (or text, etc.) to strongly activate a neuron, researchers can inspect what that neuron has learned to detect – for example, a neuron in a vision model might consistently visualise as “dog faces” or “horizontal lines,” indicating it functions as a dog-face detector or edge detector. [Olah et al. \(2017\)](#) pioneered these approaches in deep vision models, revealing strikingly interpretable neurons (like one that detects cat faces or one that responds to textures) when visualised. Building on this, *network dissection* by [Bau et al. \(2020\)](#) took a more systematic approach: they evaluated each convolutional unit against a large set of human-defined visual concepts (like “door”, “sphere”, “stripe”) to see if activation of that unit corresponds to the presence of that concept in the image. Remarkably, they found that many units had high alignment with specific semantic concepts, effectively acting as concept detectors (e.g., a unit in a scene classifier might reliably activate on text in images, or a certain texture). This gave a *dictionary of internal features* for the network, providing a more human-understandable description of what the hidden layers are looking for.

3. Overview of Explainability and Interpretability

Mechanistic interpretability has advanced further with the notion of *circuits*, which are groups of neurons that interact to perform a subtask. Instead of focusing on single neurons, researchers attempt to map out small networks within the network that correspond to meaningful algorithms. For example, in a language model, a pair of attention heads was discovered that implements an “induction” mechanism: one head finds a pattern like “A ... A” in the context (repetition of a token), and another head uses that to copy following content, effectively implementing a simple algorithm for in-context learning of sequences (Olsson et al. 2022). This discovery provided a concrete example of reverse-engineering – identifying specific weights and their connections that realize a known algorithm (in this case, a kind of search for repeated token and completion) inside a GPT-2 model. By identifying circuits for behaviors like this, we gain confidence that we understand a piece of the model, not just at the input-output level but in terms of internal computation.

A powerful set of tools aiding mechanistic interpretability are methods for *dissecting activations*. *Activation patching*, for instance, tests the role of a neuron or group of neurons by intervening on their activation values and seeing how it affects the output. One can take the activation of a certain layer for input X and replace a subset of it with the activation from a different input Y , then run the network forward to see if the output now behaves more like X or Y . This can pinpoint whether those neurons carry a specific piece of information (if swapping them swaps some aspect of the model’s prediction). Along similar lines, *ablation studies* (zeroing out or removing neurons) examine whether certain components are necessary for certain behaviors: if ablating neuron N consistently drops performance on, say, classifying dogs but not other classes, it’s evidence that N was specialized for detecting something about dogs.

A notable recent development in mechanistic interpretability is the use of *sparse autoencoders* (SAEs) and dictionary learning on internal activations (Anthropic 2023; Anthropic 2024; Shu et al. 2025). The idea here is to find an alternate basis for the network’s activations that is more interpretable – essentially, unsupervisedly “re-describing” the hidden state in terms of a set of sparse features that the researcher can inspect. For example, instead of 512 arbitrary latent features in a transformer layer, one can train a sparse autoencoder to reconstruct those activations but with a constraint that only, say, 20 of the learned basis features are active at a time for any given input. Interestingly, studies have found that these learned sparse features often align with human-meaningful concepts. Huben et al. (2024) show that in a language model, a sparse autoencoder discovered features corresponding to things like “whether the sentence is in quotes” or “the topic is sports”, etc., with single dimensions capturing those high-level attributes across many contexts. Each sparse feature tends to be more interpretable than an entangled combination of neurons. In this sense, sparse autoencoders serve a role similar to attribution, but at the level of hidden representations: they “attribute” portions of the model’s internal state to distinct, intelligible factors. This creates a bridge between *feature attribution* and *mechanistic* views. Traditional attribution methods identify which input features drive a decision; sparse decomposition methods identify which internal features drive the decision, and what those internal features represent in terms of the input. For example, instead of saying “pixel region X was important,” a mechanistic explanation might say “this neuron (or this sparse feature) activated strongly, which corresponds to concept Z in the image, and that led the network to output class Y .” The connection between sparse autoencoders and attribution is that both are attempting

3. Overview of Explainability and Interpretability

to simplify the model’s reasoning: attribution simplifies it in terms of input, while sparse coding simplifies it in terms of internal computation. Indeed, one can consider a sparse code of activations as identifying a small set of *latent features* that serve as a concise explanation for the model’s state on a given input, analogous to how attribution identifies a small set of input features as a concise explanation for the output.

Mechanistic interpretability is still a young and rapidly evolving field, with many open problems (Sharkey et al. 2025). As models grow in size and complexity (billions of parameters), the challenge is how to scale these interpretability techniques. The community is exploring *automated circuit discovery*, where algorithms try to find significant subnetworks or clusters of neurons in giant models. There’s also interest in interpretability tools for transformers, where features might not correspond to single neurons but combinations of attention heads and neurons. Another frontier is using interpretability for *steering and controlling* model behavior (Turner et al. 2023; Meng et al. 2022): if we understand a mechanism, perhaps we can modify or intervene on it (for example, “turn off” a neuron that detects offensive content, or edit a circuit responsible for a specific erroneous behavior). Mechanistic insights could thus enable not just understanding but also improving models. Moreover, mechanistic interpretability has implications for AI safety – by deeply understanding how a model thinks, we might foresee failure modes or emergent undesirable behaviors.

In conclusion, mechanistic interpretability complements other explanation approaches by shifting the focus inward: it tries to make the inscrutable weights and activations of a neural network legible in terms of algorithms and concepts. It does not always produce a simple user-facing explanation for a single prediction; rather, it produces a scientific understanding of the model’s internals that can explain many predictions and general behavior. This approach is akin to neuroscience for neural networks – mapping out what different “neurons” do and how they cooperate. When successful, it can yield extremely satisfying explanations (e.g., “the model translates English to French by first parsing into an interlingual representation X , then mapping that to French – and here are the neurons encoding the grammatical structure”). However, it is labor-intensive and often model-specific. The hope, as articulated by researchers in Sharkey et al. (2025), is that by solving enough of these puzzles, we develop generalisable insights and possibly tools that can automate interpretability at scale. Mechanistic interpretability stands as a promising path toward truly demystifying neural networks, pushing beyond treating them as black-box functions and instead opening the box to see how intelligence emerges from the network’s components.

4. Minimal Explanations in Neural Networks

Background. Abductive and contrastive explanations can be formalised as *explainability queries*—Boolean questions that can be posed to a model and an input feature vector (Barceló et al. 2020b)—, and the computational complexity of the corresponding decision problem for different types of models can be rigorously studied (Wäldchen et al. 2021; Barceló et al. 2020b). In this chapter, we focus on two explainability queries in the context of neural models (Barceló et al. 2020b). The *Minimum Change Required* (MCR) query can be used to compute minimal contrastive explanations: given a model, an input vector $\mathbf{x}$, and $k \in \mathbb{N}$, the query is true if there exists an input vector $\mathbf{y}$ differing from $\mathbf{x}$ in at most k components and which yields a different prediction. Dually, the *Minimum Sufficient Reason* (MSR) query can be used to compute minimal abductive explanations, also called minimal prime-implicants (Shih et al. 2018); given a model, an input vector $\mathbf{x}$, and $k \in \mathbb{N}$, the query is true if there is a subset $\mathbf{y}$ of the components of $\mathbf{x}$ of size at most k that suffices to obtain the current prediction (i.e., such that the prediction remains the same regardless of the values assigned to the components not in $\mathbf{y}$).

Unfortunately, MCR and MSR are NP-complete and Σ_2^P -complete respectively, already for neural models with ReLU activations implementing a Boolean function (Barceló et al. 2020b). Although different techniques have been proposed to cope with intractability (Shi et al. 2020; Ignatiev et al. 2022; Izza & Marques-Silva 2021), these complexity results may still constitute a challenge in practical scenarios.

A way to recover tractability is to focus on neural models implementing functions satisfying additional properties, such as *monotonicity* (Marques-Silva et al. 2021; Cano et al. 2019; Cucala et al. 2022a); although this restricts the model’s expressive power, the monotonicity assumption remains appropriate for a wide range of learning tasks. Furthermore, it was shown that subset-minimal abductive explanations can be computed in polynomial time if the model implements a monotonic real-valued function (Marques-Silva et al. 2021).

Contributions. In this chapter, we study cardinality-minimal abductive and contrastive explanations for monotonic neural architectures. We first show that MCR and MSR remain intractable for FCNs implementing monotonic Boolean functions. Thus, although subset-minimal abductive and contrastive explanations can be computed in polynomial time (Marques-Silva et al. 2021), cardinality-minimal abductive or contrastive explanations cannot be efficiently computed under standard complexity assumptions.

Our hardness proofs, however, rely on neural models equipped with the step activation function. We then focus our attention on monotonic neural networks where the non-linear activations are continuous everywhere and differentiable almost everywhere (as is the case with most practical activations such as ReLU and sigmoid). We show

4. Minimal Explanations in Neural Networks

that, in this setting, both cardinality-minimal contrastive explanations (and hence the MCR query) and abductive explanations (thus the MSR query) can be computed in polynomial time by means of a greedy algorithm. Our tractability results not only apply to models implementing Boolean functions, but also to more general settings involving real-valued functions. To show correctness of our greedy algorithms, we exploit the theoretical properties of the *integrated gradients* method (Sundararajan et al. 2017), thus establishing a connection between the theory of attribution methods developed by the ML community and the theory of abductive and contrastive explanations developed by the KR community. We note, however, that our algorithms do not rely on the application of attribution methods, but only on the ability to apply the model as a black box.

We conducted experiments on two partially monotonic datasets commonly used as benchmarks for designing monotonic and partially-monotonic models (Liu et al. 2020): Blog Feedback Regression (Buza 2014), a regression dataset with 276 features and Loan Defaulter¹, a classification dataset with 28 features. We trained monotonic FCNs to reach acceptable performance and we computed both contrastive and abductive explanations. The experiments showed that contrastive explanations are typically of small cardinality, whereas abductive explanations are typically larger, which is expected for a partially monotonic dataset. In both cases, the explanations could be efficiently computed.

Chapter-specific assumptions. In this chapter, we assume that a FCN $\mathcal{N}$ has an input domain $\Delta \subseteq \mathbb{R}^{d_0}$, and we assume a final stage of the computation of $\mathcal{N}$, namely we consider a monotonic classification function $\text{cls} : \mathbb{R} \rightarrow \{0, 1\}$ and the result of applying $\mathcal{N}$ to $\mathbf{x} \in \Delta$ is the scalar $\text{cls}(\mathbf{h}^L)$. Thus, the neural network realises a function $\mathcal{N} : \Delta \mapsto \{0, 1\}$. We denote with $\hat{\mathcal{N}}(\mathbf{x}) := \mathbf{h}^L$ the output of the last layer (before classification). The monotonicity of function cls implies that there exists a prediction threshold $t := \inf_{z \in \mathbb{R}} (\text{cls}(z) = 1)$, such that $\hat{\mathcal{N}}(\mathbf{x}) > t$ implies $\mathcal{N}(\mathbf{x}) = 1$ and $\hat{\mathcal{N}}(\mathbf{x}) < t$ implies $\mathcal{N}(\mathbf{x}) = 0$.

When σ^ℓ is the rectified linear unit (ReLU) for each $\ell \in \{1, \dots, L\}$, i.e. $\sigma^\ell(x) = \max(0, x)$, we say that $\mathcal{N}$ is a ReLU FCN. When σ^ℓ is a step function for each $\ell \in \{1, \dots, L\}$, i.e. $\sigma^\ell(x) = 1$ if $x \geq z$ and 0 otherwise for some $z \in \mathbb{R}$, we say that $\mathcal{N}$ is a step-function FCN. A FCN $\mathcal{N}$ with domain Δ is *monotonic* if it satisfies the following property: for all $\mathbf{x}, \mathbf{x}' \in \Delta$, if $x_i \leq x'_i$ for all $i \in \{1, \dots, d_0\}$, then $\hat{\mathcal{N}}(\mathbf{x}) \leq \hat{\mathcal{N}}(\mathbf{x}')$. Monotonicity is syntactically ensured by requiring that the weight matrices in all layers of the network are non-negative and that the activation functions in all layers are monotonic.

Formal definition of attribution methods. Attribution methods (Sundararajan et al. 2017; Sundararajan & Najmi 2020; Shapley 1953) are a family of explanation techniques which, given as input function $f : \mathbb{R}^{d_0} \mapsto \mathbb{R}$, a vector $\mathbf{x} \in \mathbb{R}^{d_0}$ and a baseline vector $\mathbf{x}' \in \mathbb{R}^{d_0}$, assign a numerical score or contribution $C_i^f(\mathbf{x}, \mathbf{x}')$ to each component $i \in \{1, \dots, d_0\}$. Attribution methods fulfil some (or all) of the following axioms for all functions $\mathbb{R}^{d_0} \mapsto \mathbb{R}$ and vectors $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^{d_0}$, components $1 \leq i \leq d_0$ and coefficients $\lambda_1, \lambda_2 \in \mathbb{R}$: (i) *Completeness*: $f(\mathbf{x}) - f(\mathbf{x}') = \sum_{j=1}^{d_0} C_j^f(\mathbf{x}, \mathbf{x}')$; (ii) *Zero-*

¹<https://www.kaggle.com/datasets/wordsforthewise/lending-club>

4. Minimal Explanations in Neural Networks

contribution: $C_i^f(\mathbf{x}, \mathbf{x}') = 0$ whenever $f(\mathbf{y}) = f(y_1, \dots, y_{i-1}, z, y_{i+1}, \dots, y_{d_0})$ for each $\mathbf{y} \in \mathbb{R}^{d_0}$ and each $z \in \mathbb{R}$; (iii) *Symmetry:* $C_i^f(\mathbf{x}, \mathbf{x}') = C_j^f(\mathbf{x}, \mathbf{x}')$ if $x_i = x_j$, $x'_i = x'_j$ and $f(y_1, \dots, y_i, \dots, y_j, \dots, y_{d_0}) = f(y_1, \dots, y_j, \dots, y_i, \dots, y_{d_0})$ for each $\mathbf{y} \in \mathbb{R}^{d_0}$; and (iv) *Linearity:* $C_i^{\lambda_1 f_1 + \lambda_2 f_2}(\mathbf{x}, \mathbf{x}') = \lambda_1 C_i^{f_1}(\mathbf{x}, \mathbf{x}') + \lambda_2 C_i^{f_2}(\mathbf{x}, \mathbf{x}')$.

Completeness ensures that contributions add up to the change in value of the function. Zero-contribution ensures that arguments not influencing the value of the function are assigned 0 as contribution. Symmetry ensures that arguments playing a symmetric role are assigned the same contribution. Finally, linearity ensures that contributions for a function expressed as a linear combination of other functions can be computed as a linear combination of their contributions.

We will exploit the properties of *Integrated Gradients* (or *Aumann-Shapley values*) (Sundararajan et al. 2017; Aumann & Shapley 1974), which is applicable to continuous functions differentiable almost everywhere. The contribution of argument i of function f for input vector $\mathbf{x}$ and baseline $\mathbf{x}'$ is defined as follows:

$$C_i^f(\mathbf{x}, \mathbf{x}') := (x_i - x'_i) \int_0^1 (\nabla f)_i(\mathbf{x}' + \tau(\mathbf{x} - \mathbf{x}')) d\tau. \quad (4.1)$$

Integrated gradients is the only path-based attribution method satisfying all of the aforementioned axioms (Friedman 2004). Furthermore, it is well-suited for functions realised by neural networks, which typically satisfy its continuity and differentiability requirements.

Explainability queries and explanations. An explainability query is a Boolean question, formalised as a decision problem, that we ask about a model and an input vector.

Consider a domain $\Delta \subseteq \mathbb{R}^{d_0}$. Given vector $\mathbf{x} \in \Delta$ and function $f : \Delta \mapsto \{0, 1\}$, a *contrastive explanation* is a subset $S \subseteq \{1, \dots, d_0\}$ such that $f(\mathbf{x}^{S|y}) \neq f(\mathbf{x})$ for some vector $\mathbf{y} \in \Delta$. The *Minimum Change Required (MCR)* query is to decide whether there exists a contrastive explanation for the given f and $\mathbf{x}$ of size at most a given number $1 \leq k \leq d_0$.

Given $\mathbf{e} \in \Delta$ and $f : \Delta \mapsto \{0, 1\}$, an *abductive explanation* is a subset $S \subseteq \{1, \dots, d_0\}$ such that $f(\mathbf{e}^{\bar{S}|z}) = f(\mathbf{e})$ for all $\mathbf{z} \in \Delta$, where $\bar{S} = \{1, \dots, d_0\} \setminus S$. The *Minimum Sufficient Reason (MSR)* query is to decide whether there exists an abductive explanation for the given function f and vector $\mathbf{e}$ of size at most a given number $1 \leq k \leq d_0$.

A contrastive (respectively, abductive) explanation is cardinality-minimal if there exists no contrastive (respectively, abductive) explanation of smaller cardinality for the same function and input vector.

4.1. Intractability for Monotonic Networks

MCR is known to be NP-hard already for FCNs implementing a Boolean function and equipped with ReLU activations only and a threshold-based classification function (Barceló et al. 2020b). In turn, MSR is Σ_2^P -hard for the same setting. A natural way

4. Minimal Explanations in Neural Networks

to recover tractability is to focus on models implementing functions with specific properties, where monotonicity is a natural requirement in many applications.

It was shown in Marques-Silva et al. (2021) that subset-minimal abductive explanations for any ML classifier implementing a monotonic function can be computed in polynomial time. This tractability result is encouraging as well as rather general: it makes no assumptions on the type and structure of the monotonic classifier, or on the domain (e.g., real-valued or Boolean) of the corresponding monotonic function.

The complexity of MCR and MSR in the monotonic setting, however, remains unclear. On the one hand, the algorithms in Marques-Silva et al. (2021) cannot be used to compute cardinality-minimal explanations and thus their tractability results do not imply tractability of MCR and MSR. On the other hand, the neural networks used in the hardness proofs in Barceló et al. (2020b) are non-monotonic.

In this section we close this gap and show that, surprisingly, both MCR and MSR remain intractable in general for monotonic neural networks, already in the Boolean case.

Theorem 4.1. *MCR is NP-complete for monotonic Boolean functions implemented by FCNs.*

Proof. We show hardness by reduction from SET-COVER, which is the problem of checking, given as input m subsets $E_1, \dots, E_m$ of $\{1, \dots, d_0\}$ such that $\bigcup_{i \in \{1, \dots, m\}} E_i = \{1, \dots, d_0\}$ whether there exists $S \subseteq \{1, \dots, m\}$ of size at most K such that $\bigcup_{i \in S} E_i = \{1, \dots, d_0\}$.

We map an instance $E_1, \dots, E_m, K$ of SET-COVER to an instance of MCR for monotonic Boolean functions implemented by FCNs by setting $k = K$, $\mathbf{x} = \mathbf{0}_m$, and $\mathcal{N}$ the 2-layer monotonic step-function FCN defined as given next.

In the first layer, $\mathbf{W}^1$ is a $(d_0 \times m)$ matrix with values $W_{j,i} = 1$ if $j \in E_i$ and 0 otherwise, $\mathbf{b}^1 = \mathbf{0}_{d_0}$ and the activation function σ^1 is a step function $\sigma^1(z) = 1$ if $z > 0$ and 0 otherwise. In the second layer, $\mathbf{W}^2 = \mathbf{1}_{d_0}$, $\mathbf{b}^2 = \mathbf{0}$, σ^2 is the step function $\sigma^2(z) = 1$ if $z \geq d_0$ and 0 otherwise, and $\text{cls}_{\{0,1\}}$ is the identity. One can verify that $\mathcal{N}(\mathbf{x}) = 0$.

Assume there exists $S \subseteq \{1, \dots, m\}$ of cardinality at most $k = K$ and $\mathbf{y} \in \{0, 1\}^m$ satisfying $\mathcal{N}(\mathbf{x}^{S|\mathbf{y}}) \neq \mathcal{N}(\mathbf{x})$. We claim that, by construction of $\mathcal{N}$, S is a solution to the corresponding SET-COVER instance. Given that $\mathcal{N}$ can only take values 0 and 1, we must have $\mathcal{N}(\mathbf{x}^{S|\mathbf{y}}) = 1$, thus $h^2 \geq d_0$. This enforces that $h_j^1 > 0$ for all $j \in \{1, \dots, d_0\}$. By construction, for all $j \in \{1, \dots, d_0\}$, there exists $i \in S$ such that $j \in E_i$, and $y_i > 0$. In particular, this gives us $\bigcup_{i \in S} E_i = \{1, \dots, d_0\}$.

For the converse, let S be a solution to SET-COVER. We claim that S and vector $\mathbf{y} = \bar{\mathbf{0}}_m$ constitute a certificate for the constructed MCR instance. Indeed, $\bigcup_{i \in S} E_i = \{1, \dots, d_0\}$ thus for all $j \in \{1, \dots, d_0\}$, there exists $i \in S$ such that $j \in E_i$. Thus, with input $\mathbf{x}^{S|\bar{\mathbf{0}}_m}$, $h_j^1 > 0$ for all $j \in \{1, \dots, d_0\}$. This yields $h^2 = d_0$ and hence $\mathcal{N}(\mathbf{x}^{S|\mathbf{y}}) = 1 \neq \mathcal{N}(\mathbf{x})$.

Membership in NP follows since $S \subseteq \{1, \dots, m\}$ of cardinality at most k and $\mathbf{y} \in \{0, 1\}^m$ provide a certificate: $(S, \mathbf{y})$ witnesses a solution of MCR for input $\mathcal{N}, \mathbf{x}$ and k if $\mathcal{N}(\mathbf{x}^{S|\mathbf{y}}) \neq \mathcal{N}(\mathbf{x})$, which is polytime verifiable. $\square$

Theorem 4.2. *MSR is NP-complete for monotonic Boolean functions implemented by FCNs.*

4. Minimal Explanations in Neural Networks

Proof. We again show NP-hardness by reduction from SET-COVER. We map an instance $E_1, \dots, E_m, K$ of SET-COVER to an instance of MSR by setting $k = K$, $\mathbf{e} = \bar{\mathbf{0}}_m$, and $\mathcal{N}$ the 2-layer monotonic step-function FCN described in the proof of Theorem 4.1. One can verify that $\mathcal{N}(\mathbf{e}) = 1$.

Assume there exists $S \subseteq \{1, \dots, m\}$ of cardinality at most $k = K$ satisfying $\mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{z}}) = \mathcal{N}(\mathbf{e})$ for all $\mathbf{z} \in \{0, 1\}^m$. We claim that S is a solution to the corresponding SET-COVER instance. In particular, we have $\mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{0}_m}) = 1$, thus $h^2 \geq d_0$. This enforces $h_j^1 > 0$ for all $j \in \{1, \dots, d_0\}$. By construction, for all $j \in \{1, \dots, d_0\}$, there exists $i \in S$ such that $j \in E_i$, which is equivalent to $\bigcup_{i \in S} E_i = \{1, \dots, d_0\}$.

For the converse, let S be a solution to SET-COVER. We claim that S is a certificate for the constructed MSR instance. Indeed, $\bigcup_{i \in S} E_i = \{1, \dots, d_0\}$ thus for all $j \in \{1, \dots, d_0\}$, there exists $i \in S$ such that $j \in E_i$. Thus, with input $\mathbf{e}^{\bar{S}|\mathbf{0}_m}$, $h_j^1 > 0$ for all $j \in \{1, \dots, d_0\}$. This yields $h^2 = d_0$ and hence $\mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{0}_m}) = 1 = \mathcal{N}(\mathbf{e})$. By monotonicity, we thus have $\mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{z}}) \geq \mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{0}_m}) = 1$ for all $\mathbf{z} \in \{0, 1\}^m$.

Membership in NP follows since a set $S \subseteq \{1, \dots, m\}$ of cardinality at most k provides a certificate. Indeed, MSR is true if $\mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{0}_m}) = \mathcal{N}(\mathbf{e}) = 1$, or $\mathcal{N}(\mathbf{e}^{\bar{S}|\mathbf{0}_m}) = \mathcal{N}(\mathbf{e}) = 0$, which is verifiable in polynomial time. $\square$

Note that, although MSR remains intractable, monotonicity does bring its complexity down from the second level of the polynomial hierarchy to NP for Boolean functions.

4.2. Achieving Tractability of MCR and MSR

The hardness proofs in Section 4.1 rely on the use of the step activation function which, in contrast to the activations used in practice such as ReLU, is a discontinuous function.

In this section, we show that cardinality-minimal abductive and contrastive explanations become polytime computable (and hence MSR and MCR become tractable) if we additionally assume that the activations in the FCNs implementing the monotonic function of interest are continuous everywhere and differentiable almost everywhere; these are mild restrictions that are satisfied by most practical activation functions.

Definition 4.3. An activation function is *admissible* if it is continuous everywhere, differentiable almost everywhere, and non-decreasing.

Furthermore, our tractability results can be extended beyond the Boolean setting to real-valued functions over a bounded domain as defined next.

Definition 4.4. A domain $\Delta \subset \mathbb{R}^{d_0}$ is *bounded* if there exist lower and upper bound vectors $\mathbf{l} \in \Delta$ and $\mathbf{u} \in \Delta$ such that, for all $\mathbf{x} \in \Delta$ and each $i \in \{1, \dots, d_0\}$, we have $l_i \leq x_i \leq u_i$.

Note that Boolean domains are bounded by the null vector of the relevant dimension and its complement.

4.2.1. Properties of Monotonic Networks with Admissible Activation

In the remainder of this section, we focus on FCNs where all activations are admissible and where monotonicity is ensured syntactically by requiring that weight matrices in all layers contain only non-negative weights. Let us therefore fix an arbitrary FCN $\mathcal{N}$ over a domain $\Delta \subseteq \mathbb{R}^{d_0}$ of dimension d_0 satisfying these requirements, which we exploit in the formulation of our results; furthermore, assume that Δ is bounded by a lower bound vector $\mathbf{l}$ and upper bound vector $\mathbf{u}$.

The continuity and differentiability requirements of the activation functions ensure that the gradient of $\hat{\mathcal{N}}$ can be computed for each input feature vector $\mathbf{x}$. In turn, as we show next, the monotonicity requirement ensures that each component of the gradient of $\hat{\mathcal{N}}$ (the output of the last layer) at $\mathbf{x}$ can be expressed as a sum where (1) the number of elements in the sum is fixed for $\mathcal{N}$ (i.e., it does not depend on $\mathbf{x}$) and it is the same for all vector components; and (2) each element of the sum consists of a product involving a value that depends on $\mathbf{x}$ but which is *always non-negative*, and two coefficients that do not depend on $\mathbf{x}$. This key property of the gradient allows us to exploit the theoretical properties of the integrated gradients attribution method. In particular, we can show that, by setting a component x_i of the input vector $\mathbf{x}$ to the corresponding component of the lower or upper bound vector, depending on whether the prediction for $\mathbf{x}$ is 1 or 0, we are not altering the relative order of the integrated gradient attributions for the remaining components.

These properties, which are established by the following technical lemma, constitute the basis of our greedy algorithms for answering explainability queries.

Lemma 4.5. *There exists an integer $M \in \mathbb{Z}_{\geq 0}$, positive coefficients $\{A_m\}_{1 \leq m \leq M}$, and $\{B_i\}_{1 \leq i \leq d_0}$, and functions $\{g_m\}_{1 \leq m \leq M}$ from Δ to $\mathbb{R}_{\geq 0}$, such that the following identities are satisfied for each $\mathbf{x}, \mathbf{x}' \in \Delta$ and each $i, j \in \{1, \dots, d_0\}$*

$$(\nabla \hat{\mathcal{N}})_i(\mathbf{x}) = B_i \sum_{m=1}^M A_m g_m(\mathbf{x}) \quad (4.2)$$

and

$$\hat{\mathcal{N}}(\mathbf{x}^{\{i\}|\mathbf{x}'}) - \hat{\mathcal{N}}(\mathbf{x}^{\{j\}|\mathbf{x}'}) = (B_i(x'_i - x_i) - B_j(x'_j - x_j)) \sum_{m=1}^M A_m \int_0^1 g_m(\mathbf{p}^{ij}(\tau)) d\tau, \quad (4.3)$$

where $\mathbf{p}^{ij}(\tau) = \mathbf{x}^{\{j\}|\mathbf{x}'} + \tau(\mathbf{x}^{\{i\}|\mathbf{x}'} - \mathbf{x}^{\{j\}|\mathbf{x}'}).$

Proof. We show (4.2) by induction on the number of layers L in $\hat{\mathcal{N}}$. If $L = 1$, then $\hat{\mathcal{N}} = \langle \mathbf{W}, b, \sigma \rangle$ with $\mathbf{W} \in \mathbb{R}^{d_0}$, $b \in \mathbb{R}$ and $\sigma : \mathbb{R} \mapsto \mathbb{R}$. By the chain rule, $(\nabla \hat{\mathcal{N}})_i(\mathbf{x}) = W_i \cdot (D\sigma)(\mathbf{W} \cdot \mathbf{x} + b)$, with $D\sigma$ the derivative of σ in Euler's notation. This is of the form (4.2) with $M = 1$, $A_1 = 1 \geq 0$, $B_i = W_i$, and $g_1(\mathbf{x}) = (D\sigma)(\mathbf{W} \cdot \mathbf{x} + b)$. Monotonicity of σ ensures $g_1(\mathbf{x}) \geq 0$ for any $\mathbf{x}$.

For the inductive case, assume (4.2) holds for each network with $L - 1$ layers satisfying the same requirements as $\mathcal{N}$. The application of $\hat{\mathcal{N}} = \langle \{\mathbf{W}^\ell\}_{1 \leq \ell \leq L}, \{\mathbf{b}^\ell\}_{1 \leq \ell \leq L}, \{\sigma^\ell\}_{1 \leq \ell \leq L} \rangle$

4. Minimal Explanations in Neural Networks

with L layers to $\mathbf{x}$ is defined as $\sigma^L(h^L(\mathbf{x}))$. By the chain rule and the definition of h^L we obtain the following identity:

$$(\nabla \hat{\mathcal{N}})_i(\mathbf{x}) = (\nabla h^L)_i(\mathbf{x}) \cdot (D\sigma^L)(h^L(\mathbf{x})) = \sum_{j=1}^{d_{L-1}} \mathbf{W}_j^L \cdot (\nabla \hat{\mathcal{N}}^j)_i(\mathbf{x}) \cdot (D\sigma^L)(h^L(\mathbf{x})). \quad (4.4)$$

Here, $\hat{\mathcal{N}}^j$ is given by weight matrices $\{\mathbf{W}^\ell\}_{1 \leq \ell \leq L-2}$ and $\mathbf{W}_j^{L-1}$ (representing the j -th row of $\mathbf{W}^{L-1}$), bias vectors $\{\mathbf{b}^\ell\}_{1 \leq \ell \leq L-2}$ and b_j^{L-1} (representing the j -th element of $\mathbf{b}^{L-1}$), and activations $\{\sigma^\ell\}_{1 \leq \ell \leq L-1}$. We apply the inductive hypothesis to compute the gradient for each $1 \leq j \leq d_L - 1$, which is $(\nabla \hat{\mathcal{N}}^j)_i(\mathbf{x}) = B_i^j \sum_{m_j=1}^{M_j} A_{m_j}^j g_{m_j}^j(\mathbf{x})$. But now, we can replace the value of the gradients in the sum of (4.4) with these values and show the statement of the lemma by instantiating (4.2) with $M = \sum_{j=1}^{d_{L-1}} M_j$, $A_m = W_j^L A_{m_j}^j$, $B_i = B_i^j$ and $g_m(\mathbf{x}) = g_{m_j}^j(\mathbf{x}) \cdot (D\sigma^L)(h^L(\mathbf{x}))$. Again, by induction, $A_m \geq 0$ and $g_m(\mathbf{x}) \geq 0$ for each m and $\mathbf{x}$.

We now show (4.3). Let us consider the attribution for $\hat{\mathcal{N}}$ defined in (4.1). Assume $i \neq j$ (otherwise the equation holds trivially). By replacing the gradient in (4.1) with (4.2), the value of $C_i^{\hat{\mathcal{N}}}(\mathbf{x}, \mathbf{x}')$ is $B_i(x_i - x'_i) \sum_{m=1}^M A_m \int_0^1 g_m(\mathbf{x}' + \tau(\mathbf{x} - \mathbf{x}')) d\tau$. Since integrated gradients satisfy the completeness and zero contribution axioms, we can compute the difference $\hat{\mathcal{N}}(\mathbf{x}^{\{i\}|\mathbf{x}'}) - \hat{\mathcal{N}}(\mathbf{x}^{\{j\}|\mathbf{x}'})$ as the sum of contributions $C_i^{\hat{\mathcal{N}}}(\mathbf{x}^{\{i\}|\mathbf{x}'}, \mathbf{x}^{\{j\}|\mathbf{x}'})$ and $C_j^{\hat{\mathcal{N}}}(\mathbf{x}^{\{i\}|\mathbf{x}'}, \mathbf{x}^{\{j\}|\mathbf{x}'})$ to obtain

$$(x'_i - x_i) \int_0^1 (\nabla \hat{\mathcal{N}})_i(\mathbf{p}^{ij}(\tau)) d\tau - (x'_j - x_j) \int_0^1 (\nabla \hat{\mathcal{N}})_j(\mathbf{p}^{ij}(\tau)) d\tau. \quad (4.5)$$

The gradients $(\nabla \hat{\mathcal{N}})_i(\mathbf{p}^{ij}(\tau))$ and $(\nabla \hat{\mathcal{N}})_j(\mathbf{p}^{ij}(\tau))$ are provided by (4.2); when replaced in (4.5), they yield (4.3). $\square$

4.2.2. Algorithms for Computing Explanations

We are now ready to present our greedy algorithms for computing cardinality-minimal explanations in polynomial time.

Algorithm 1 takes as input a monotonic FCN $\mathcal{N}$ with admissible activation functions over a bounded domain Δ with lower bound $\mathbf{l}$ and upper bound $\mathbf{u}$, and an input feature vector $\mathbf{x} \in \Delta$, and computes a cardinality-minimal contrastive explanation as detailed next.

The algorithm first applies the model $\mathcal{N}$ to the input vector $\mathbf{x}$ and, based on the obtained prediction, chooses to consider the domain's lower bound vector $\mathbf{l}$ (if $\mathcal{N}(\mathbf{x}) = 1$) or the upper bound vector $\mathbf{u}$ (if $\mathcal{N}(\mathbf{x}) = 0$) when searching for vectors that change the model's prediction. The monotonicity requirement will ensure that no other vectors need to be considered. Then, in each iteration of the first loop, the algorithm sets

Algorithm 1 Computing contrastive explanations.

Input: vector $\mathbf{x} \in \Delta$ with Δ bounded by vectors $\mathbf{l}, \mathbf{u}$, and monotonic FCN $\mathcal{N} : \Delta \mapsto \{0, 1\}$ with admissible activation functions.

Output: A cardinality-minimal contrastive explanation S for $\mathcal{N}$ and $\mathbf{x}$

```

1: if  $\mathcal{N}(\mathbf{x}) = 1$  then
2:    $\mathbf{x}' \leftarrow \mathbf{l}$ 
3: else
4:    $\mathbf{x}' \leftarrow \mathbf{u}$ 
5: end if
6: for  $1 \leq j \leq d_0$  do
7:    $c_j \leftarrow \hat{\mathcal{N}}(\mathbf{x}^{\{j\}|\mathbf{x}'})$ 
8: end for
9:  $I \leftarrow$  list of indices obtained from sorting  $\{c_j\}_{1 \leq j \leq d_0}$  in ascending (respectively,
   descending) order with ties broken arbitrarily if  $\mathcal{N}(\mathbf{x}) = 1$  (respectively, if  $\mathcal{N}(\mathbf{x}) = 0$ ).
10:  $S \leftarrow \emptyset$ 
11: for  $1 \leq j \leq d_0$  do
12:    $S \leftarrow S \cup I[j]$ 
13:   if  $\mathcal{N}(\mathbf{x}^{S|\mathbf{x}'}) \neq \mathcal{N}(\mathbf{x})$  then return  $S$ 
14: end for
    
```

each individual input feature to the value of the relevant bound vector (while leaving the remaining components unchanged) and applies the input model to the resulting vector. The values obtained by each of these applications of the model are then sorted in ascending or descending order depending on the value of $\mathcal{N}(\mathbf{x})$. In the second loop, the algorithm successively assigns the components of $\mathbf{x}$ to the chosen bound vector in the order established in the previous step until the prediction changes. The algorithm then returns S consisting of all features that were set to the chosen bound.

Our algorithm is quadratic in the number of input features: both loops require linearly many applications of the FCN, and each application is feasible in linear time in the number of features (Goodfellow et al. 2016). The algorithm’s correctness relies on (4.3) in Lemma 4.5, which ensures that, when set to the chosen bound, each of the features selected by the algorithm in the second loop yields the largest change (amongst all other possible feature choices) in the application of the model, thus getting as close as possible to the prediction threshold. As a result, the output subset S is guaranteed to contain a smallest number of features.

Theorem 4.6. *Algorithm 1 computes a cardinality-minimal contrastive explanation for the input $\mathcal{N}$ and $\mathbf{x}$.*

Proof. By symmetry of the algorithm, we can assume, without loss of generality that $\mathcal{N}(\mathbf{x}) = 1$. By monotonicity of cls, it suffices to show that, for each $j \in \{1, \dots, d_0\}$, the choice of $I[j]$ in the second loop yields the largest change in the evaluation of $\mathcal{N}$ (the output of the last layer). That is, for each $1 \leq j \leq d_0$ and $j \leq k \leq d_0$ we have $\hat{\mathcal{N}}(\mathbf{x}^{S|\mathbf{x}'}) - \hat{\mathcal{N}}(\mathbf{x}^{(S \cup I[j])|\mathbf{x}'}) \geq \hat{\mathcal{N}}(\mathbf{x}^{S|\mathbf{x}'}) - \hat{\mathcal{N}}(\mathbf{x}^{(S \cup I[k])|\mathbf{x}'})$. By construction of list I , the inequality $\hat{\mathcal{N}}(\mathbf{x}^{I[j]|\mathbf{x}'}) - \hat{\mathcal{N}}(\mathbf{x}^{I[k]|\mathbf{x}'}) \leq 0$ holds for each $1 \leq j \leq k \leq d_0$. We apply (4.3) in

4. Minimal Explanations in Neural Networks

Lemma 4.5 together with the fact that $A_m \geq 0$ and $g_m(\mathbf{x}) \geq 0$ for each m and $\mathbf{x}$ (and hence $\int_0^1 g_m(\mathbf{p}^{I[j] I[k]}(\tau)) d\tau \geq 0$) to obtain $(B_{I[j]}(x'_{I[j]} - x_{I[j]}) - B_{I[k]}(x'_{I[k]} - x_{I[k]})) \leq 0$. Since $\{I[j], I[k]\} \subseteq \bar{S}$, we have $(\mathbf{x}^{S|\mathbf{x}'})_{I[j]} = x_{I[j]}$ and $(\mathbf{x}^{S|\mathbf{x}'})_{I[k]} = x_{I[k]}$. By applying (4.3) and the previous inequality, we finally obtain $\hat{\mathcal{N}}(\mathbf{x}^{(S \cup I[j])|\mathbf{x}'}) \leq \hat{\mathcal{N}}(\mathbf{x}^{(S \cup I[k])|\mathbf{x}'})$. This ensures that the output S is a cardinality-minimal contrastive explanation for $\mathcal{N}$ and $\mathbf{x}$, as required. $\square$

Contrastive and abductive explanations are dual to one another. Therefore, as we show next, a minor modification of Algorithm 1 that exchanges the roles of vectors $\mathbf{x}$ and $\mathbf{x}'$ in the second loop can be used to compute cardinality-minimal abductive explanations.

Theorem 4.7. *A modified version of Algorithm 1 where $\mathbf{x}$ and $\mathbf{x}'$ are interchanged in Lines 7, 9 and 13 computes a cardinality-minimal abductive explanation for the given input $\mathcal{N}$ and $\mathbf{x}$.*

Proof. The proof is analogous to that of Theorem 4.6 in showing that the choice of $I[j]$ in the second loop yields the largest change in the evaluation of $\hat{\mathcal{N}}$, thus ensuring that S is a cardinality-minimal subset such that $\mathcal{N}(\mathbf{x}^{S|\mathbf{x}}) \neq \mathcal{N}(\mathbf{x}')$. By the choice of $\mathbf{x}'$, we have ensured that $\mathcal{N}(\mathbf{x}') \neq \mathcal{N}(\mathbf{x})$, which gives us that $\mathcal{N}(\mathbf{x}^{\bar{S}|\mathbf{x}'}) = \mathcal{N}(\mathbf{x})$. Furthermore, by monotonicity, $\mathcal{N}(\mathbf{x}^{\bar{S}|\mathbf{z}}) = \mathcal{N}(\mathbf{x})$ for all $\mathbf{z} \in \Delta$ if and only if $\mathcal{N}(\mathbf{x}^{\bar{S}|\mathbf{l}}) = \mathcal{N}(\mathbf{x}) = 1$ or $\mathcal{N}(\mathbf{x}^{\bar{S}|\mathbf{u}}) = \mathcal{N}(\mathbf{x}) = 0$. This ensures that S is a minimal abductive explanation. $\square$

Note that, although the correctness of our algorithms relies on the properties of integrated gradients, the algorithms themselves do not compute attribution values, and only rely on the ability to apply the input model as a ‘black box’.

4.3. Discussion and Further Implications

The notion of cardinality-minimal contrastive explanation is closely related to existing notions of *robustness* for ML model predictions proposed in the literature. In particular, the minimality requirement ensures that no smaller subset of the features can be used to change the prediction for a given model $\mathcal{N}$ and input feature vector $\mathbf{x}$; thus, the larger the size of the smallest contrastive explanation, the more robust the prediction of $\mathcal{N}$ on $\mathbf{x}$ is. The notion of D-robustness (Shi et al. 2020) is an instance-based robustness measure based precisely on this idea. The D-ROBUST query can be formalised as the complement of the MCR query: given $\mathcal{N}$, $\mathbf{x}$, and k , decide whether the size of a cardinality-minimal contrastive explanation for $\mathcal{N}$ and $\mathbf{x}$ is at least k . It was shown in Shi et al. (2020) that D-ROBUST is coNP-complete for Boolean functions. Thus, Theorem 4.1 in Section 4.1 refines the complexity lower bound in Shi et al. (2020) to monotonic Boolean functions realised by FCNs with step activations. Furthermore, our results in Section 4.2.2 imply tractability of D-ROBUST for monotonic functions implemented by FCNs with admissible activations and bounded domains.

Our results also have interesting implications on the problem of constructing neural networks that exactly replicate a given function (Blum & Rivest 1988; Judd 1988; Jones 1997; Kumar et al. 2019). In particular, our results imply that, unless $P = NP$, there is no polynomial time algorithm that, given as input a monotonic FCN with step

4. Minimal Explanations in Neural Networks

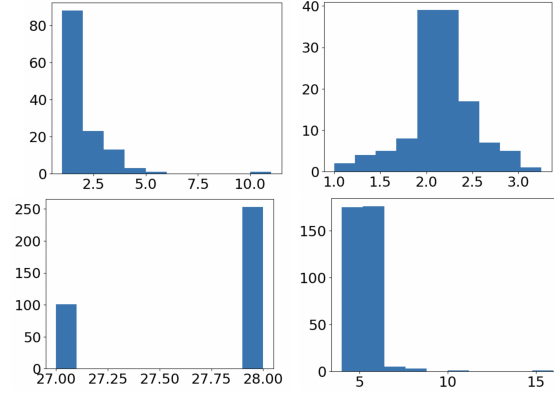


Figure 4.1.: Explanation sizes (top left: contrastive; and bottom left: abductive) and computing times (top right: contrastive and bottom right: abductive) on Loan Defaulter. X-axis are indexed respectively by the cardinality of explanations and the computing time in seconds. Y-axis are indexed by the number of examples.

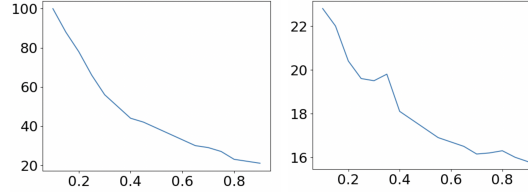


Figure 4.2.: Explanation sizes (left) and computing times (right) on Blog Feedback. X-axis are indexed by the threshold. Y-axis are indexed respectively by the average cardinality of abductive explanations and the average computing time in seconds.

activations, constructs an FCN with admissible activations realising the same function. Indeed, otherwise we could solve in polynomial time the MCR query for monotonic FCNs with step functions by first rewriting the model using admissible activations and then applying our greedy algorithm to the transformed model.

4.4. Experiments

We have implemented our greedy algorithms for computing cardinality-minimal explanations in Section 4.2.2 and assessed their practical suitability on well-known benchmark datasets commonly used to evaluate monotonic models. To the best of our knowledge, our implementation is the only one available for computing cardinality-minimal explanations and hence we could not find a suitable benchmark for comparison. All experiments were conducted using Google Colab with GPU.

Datasets. The *Blog Feedback Regression* dataset (Buza 2014) is a numeric dataset assembled from 37,279 blog pages extracted from 1,200 different sources. The objective is to

4. Minimal Explanations in Neural Networks

predict the number of feedbacks that a blog page will receive in a given time window. The 276 features include the number of links and feedbacks in the past, time and day of publication, discriminative bag of words, etc.

Loan Defaulter is a numeric dataset assembled from LendingClub (a large online loan marketplace). The objective is to predict if the applicant will repay the loan or default. The 28 features include loans executed in the past, amount of the loan and instalments, applicant’s address and zip code, etc.

The features in both datasets are preprocessed and bounded between 0 and 1.

Methodology. We trained monotonic FCN models on both datasets with PyTorch (Paszke et al. 2019) using the mean-squared error loss for the Blog Feedback dataset and the binary cross entropy loss for the Loan Defaulter dataset. We trained the models with Adam (Kingma & Ba 2014) for 10 epochs, setting all negative weights to 0 after each iteration of Adam to ensure monotonicity. We were able to reach a root mean-squared error (RMSE) of 0.175 on the test set for the Blog Feedback regression (an acceptable performance given that the state-of-the-art is at 0.158 (Liu et al. 2020)) and reached an accuracy of 60% on Loan Defaulter (state-of-the-art performance is 65.2%). Since the datasets are only partially-monotonic, we should not expect state-of-the-art performance with a fully-monotonic model; please note, however, that the objective of our experiments is not to improve on the state-of-the-art regression and classification metrics, but rather to show that cardinality-minimal explanations for the trained models can be efficiently computed.

Using the trained monotonic FCNs, we then computed cardinality-minimal abductive and contrastive explanations using the greedy algorithms in Section 4.2.2. Note that, although the models for Blog Feedback are trained for regression, our algorithms can still seamlessly be applied provided that we introduce a threshold. Indeed, given a FCN $\mathcal{N} : \Delta \mapsto \mathbb{R}$, an input vector $\mathbf{x} \in \Delta$ and a numeric threshold t such that $\mathcal{N}(\mathbf{x}) > t$ (respectively, $\leq t$), Algorithm 1 can be used to compute a cardinality-minimal subset S such that $\mathcal{N}(\mathbf{x}^{S|\mathbf{x}'}) \leq t$ (respectively, $> t$) and a cardinality-minimal S such that $\mathcal{N}(\mathbf{x}^{\bar{S}|\mathbf{z}}) > t$ (respectively, $\leq t$) for all $\mathbf{z} \in \Delta$.

Results. For Loan Defaulter, contrastive explanations took 1.5s on average to compute and contained 1.8 out of 28 features on average; in turn, abductive explanations took 5s on average to compute and contained 27.6 features on average. Note that contrastive explanations were much smaller than abductive explanations; this is not unexpected in a partially monotonic dataset since the condition required from abductive explanations requires the prediction to hold *for all possible values* of the features outside the explanation (a very strong condition). Figure 4.1 depicts the cardinalities and computation times for both types of explanations.

For the Blog Feedback Regression dataset, we varied the threshold between the lower bound and the upper bound of the targets and computed the average cardinality and computation time with respect to the threshold for both types of explanations. For contrastive explanations, the cardinality remained constant equal to 1: it was always possible to modify one feature and change the prediction. In the plots of Figure 4.2, we focused on abductive explanations and instances $\mathbf{x}$ such that $\mathcal{N}(\mathbf{x}) \leq t$ and, as expected,

we can see that the sizes of explanations decrease as the threshold increases.

4.5. Concluding remarks of the chapter

In this chapter, we have studied the problem of computing cardinality-minimal contrastive and abductive explanations for the predictions of monotonic neural models. We have strengthened existing intractability results ([Barceló et al. 2020b](#)) to the context of monotonic FCNs and proposed additional requirements to regain tractability.

Our results are of practical interest for the computation of explanations with formal guarantees in the context of monotonic or partially-monotonic tasks. Furthermore, from a theoretical perspective, our results not only strengthen existing intractability results, but to the best of our knowledge they also provide the first polytime algorithms for computing cardinality-minimal explanations in the context of neural models. Finally, our results also establish a novel connection between the theory of attribution methods and the theory of abductive and contrastive explanation methods, and have interesting implications on the computational complexity of other related problems, namely the D-ROBUST query, a decision problem which evaluates instance-based robustness, and the problem of constructing a neural network that replicates a given function.

5. Fibring: a formalism connecting neural networks and logical theories

Background. As logic is inherently more interpretable than sub-symbolic learning systems and amenable to formal verification, there has been a surge of interest in neuro-symbolic AI that integrates neural networks with logical reasoning to improve transparency and trustworthiness (Lamb et al. 2020; Garcez & Lamb 2023).

Fibring of Neural Networks (Garcez & Gabbay 2004) is a theoretical concept from the neurosymbolic AI literature introduced as a way to combine neural network architectures. The idea is to enforce that the parameters of a network are a function of another network, and that the resulting output is re-injected into the original network, computing the output of the combined (fibred) neural network. This theoretical framework was initially inspired by the concept of fibring modal logics (Gabbay 1999), a particular way to combine modal logics. Analogously, in a fibred modal language, the Kripke model and the world in which a formula is evaluated is a function of a possible world in another Kripke model¹. However, the precise correspondence between fibring of neural networks and fibring of modal logics was never formally established.

At the same time, an important research area in neurosymbolic AI aims to establish connections between logic and modern neural architectures, motivated by the prospect of rendering them more interpretable and verifiable. In particular, the capabilities of GNNs and Transformer has sparked a significant research effort to rigorously analyze their *logical expressiveness*—the classes of formal languages they can compute—and their formal verification.

The theoretical research thread of logical expressiveness has emerged to characterize the expressive power of neural models in terms of logic. It bears obvious connections with the interpretability line of research on (logical) rule-based methods but here the goal is not necessarily to provide a user-facing explanation, but to understand what classes of computations or patterns a model can represent, by relating them to formal languages. This kind of correspondence provides interpretability at a high level: it tells us the kinds of logical patterns a model can capture. Understanding logical expressiveness can guide explainability: if a model is known to correspond to a certain logic, one can attempt to systematically extract rules that reproduce the behavior specified by the logic’s syntax and semantics on the model’s typical inputs.

An interesting connection exists between GNNs and Transformer encoders (Bronstein et al. 2021): Transformer encoders can be viewed as GNNs applied to complete graphs with added positional encoding and attention mechanism. This connection, however, has not been exploited in the literature on logical expressivity and formal verification. This gap leaves significant potential for unifying logical characterisations of both

¹The word *model* here refers to structures with assignments of truth-values, differently from the use of the word in neural networks.

5. Fibring: a formalism connecting neural networks and logical theories

architectures.

Existing logical expressiveness results for GNNs and Transformers can be categorised into: *discriminative power* – e.g. is there a logical classifier which can distinguish the same pairs of nodes as a GNN?–; *uniform expressiveness*, – e.g. is there a logical formula whose truth values coincide with the output of a GNN on *any* input graph and node?– and *non-uniform expressiveness* – in which the logical formulas can depend on the input. In this categorisation, uniform expressiveness results stand out as the most powerful kind, providing complete logical characterisations of neural architectures (a single logical formula for each instance of the neural architecture). Initial results established that the discriminative power of standard GNNs is upper-bounded by the Weisfeiler-Leman (WL) test (Barceló et al. 2020a). Subsequently, (Grohe 2023) established non-uniform expressiveness results for GNNs in terms of Boolean circuits and descriptive complexity. More recently, uniform expressiveness results of broad classes of GNNs have been derived (Nunn et al. 2024; Benedikt et al. 2024), using known logics called Presburger logics which involve counting modalities and linear inequalities. Using existing results about the satisfiability of such logics, results about the computational complexity of formal verification problems such as: given an output and a GNN, is there an input to the GNN which yields the output?, follow as corollaries. In particular, (Benedikt et al. 2024) provides a taxonomy of the decidability of these verification problems depending on the types of aggregations and activation functions. Then, (Cuenca Grau et al. 2025) established uniform expressiveness results for bounded GNNs in terms of fragments of first-order logic, showcasing how restrictive, yet practical, assumptions on the class of GNNs significantly simplifies the logical characterisation. On the other hand, Transformer encoders have also been studied through the lens of circuit complexity theory. Unique Hard Attention Transformers (UHATs), have been mapped to fragments of complexity class AC^0 languages and extensions like FO(Mon), a restricted form of first-order logic (Hao et al. 2022). Average Hard Attention Transformers (AHATs) have further demonstrated their ability to capture more complex languages, including those outside AC^0 , and most recently, a (uniform) lower bound has been established (Barcelo et al. 2024) in terms of linear temporal logic: a language called LTL (C, +), which also involves counting modalities.

Contributions. In this chapter, we propose fibring of modal logics as a new formalism to study the logical expressiveness of neural architectures. We start by redefining fibring of neural networks (Garcez & Gabbay 2004) to make its use easier in the context of the above literature. We then formally establish an exact correspondence between fibring neural networks and fibring modal logics. That is, 1) we define a fibred language based on a fibring architecture; 2) we introduce the notion of *compatible* fibred models (with fibred neural networks and their inputs) in which to interpret the formulas; 3) we prove that our fibred logic is a valid fibred logic by demonstrating that the class of compatible fibred models is non-empty and closed under Kripke-model isomorphism; and 4) we construct the formulas from our fibred logic whose truth values coincide with the outputs of fibred neural networks. Subsequently, we prove that fibred neural networks can be used to non-uniformly describe large classes of GNNs, GATs and Transformer encoder architectures. It follows that the corresponding fibred languages provide non-uniform expressiveness results for these network architectures.

As future directions of research, we think that fibring as a formalism might hold

keys into deriving and unifying uniform expressiveness results between GNNs and transformer encoders, and we present the main arguments towards such unification. Finally, we argue that the proposed fibring framework provides a new intuitive way of thinking about GNN and Transformer architectures as successive combinations of underlying logics, and we discuss how this new perspective could be used for interpretability and verification.

5.1. Fibring Neural Networks

In this section, we introduce a definition of fibred neural networks which generalises the original definition of [Garcez & Gabbay \(2004\)](#) to any number and possible combinations of neural networks.

A *fibring architecture* is a directed tree $\mathcal{F}$ whose nodes v are labeled with neural architectures $\mathbb{A}_v$. Each edge (v, v') is labeled with : (1) a layer number ℓ in the parent architecture; and (2) a set of positions in that layer. We impose the additional requirement that for any two edges (v, w) and (v, u) sharing the parent node and labeled with the same layer number, the corresponding position sets in the label of these edges must be disjoint. Finally, we note $\mathcal{F}$ the class of fibring architectures which verify that (i), the architecture at the root is two linear layers with input dimension d_0 and output dimension 1; and (ii) every edge leaving the root node is labeled with the same layer number ℓ .

A *fibred network* $\tilde{\mathcal{N}}$ is a tuple $\langle \mathcal{N}, \mathcal{F}, \tilde{f} \rangle$ where $\mathcal{F}$ is a fibring architecture, $\mathcal{N}$ is a network instance $\mathcal{N}$ of its root, and $\tilde{f}$ is a finite collection of neural *fibring functions* matching $\mathcal{F}$, i.e. one function $\tilde{f}_{(v, v')}$ for each edge (v, v') in $\mathcal{F}$, which specifies how to build an instance of the child architecture and what input to give it; specifically, for an edge (v, v') labelled with layer number ℓ , the fibring function $\tilde{f}_{(v, v')}$ maps vectors of dimension $\leq d_\ell$ to a network instance of architecture $\mathbb{A}_{v'}$ and a valid input vector for that architecture.

To apply a fibred neural network $\tilde{\mathcal{N}}$ to an input $\mathbf{x}$, we start at the root. Whenever we reach a layer that has edges leading to children, we pause, call the corresponding fibring function, and pass part of the current vector into the child network. The child network produces a new vector, which is spliced back into the parent's computation at the specified positions. The overall computation proceeds until the output layer of the root is reached.

Formally, the computation is defined inductively. If $\mathcal{F}$ has only the root node u , then $\tilde{\mathcal{N}}(\mathbf{x}) = \mathcal{N}(\mathbf{x})$. Otherwise, let $u_1, \dots, u_k$ be the children of u , with edges (u, u_i) labeled (ℓ_i, S_i) , let $\mathcal{F}_i$ be the subtree rooted at u_i , and let $\tilde{f}_i$ be the restriction of $\tilde{f}$ to edges in $\mathcal{F}_i$. Assume the children are ordered so that $\ell_1 \leq \ell_2 \leq \dots \leq \ell_k$. For each stage $1 \leq i \leq k$, define a tuple $(\mathbf{x}_i, \mathcal{N}_i, \mathbf{y}_i, \mathbf{h}_i)$ as follows:

$$\begin{aligned} \mathbf{x}_i &= \begin{cases} \mathcal{N}^{1:\ell_1}(\mathbf{x}), & i = 1, \\ \mathcal{N}^{\ell_{i-1}:\ell_i}(\mathbf{y}_{i-1}), & i > 1, \end{cases} \\ (\mathcal{N}_i, \mathbf{y}_i) &= \tilde{f}_{(u, u_i)}(\mathbf{x}_i), \\ \mathbf{h}_i &= \text{the vector } \mathbf{x}_i \text{ with entries in } S_i \text{ replaced by } \langle \mathcal{N}_i, \mathcal{F}_i, \tilde{f}_i \rangle(\mathbf{y}_i). \end{aligned}$$

5. Fibring: a formalism connecting neural networks and logical theories

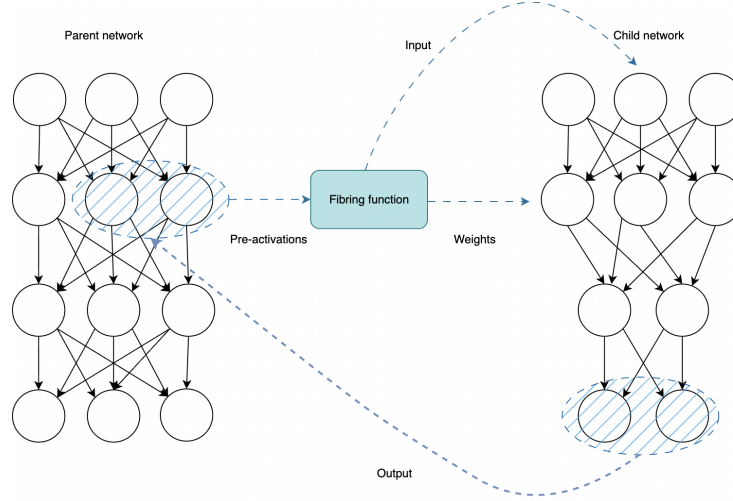


Figure 5.1.: Illustration of fibring between two architectures. The pre-activations at a given layer within the parent network are fed into a fibring function to produce the input and the weights of the child network, the output of the child network is then reinjected at the same pre-activations.

The final output is

$$\widetilde{\mathcal{N}}(\mathbf{x}) = \mathcal{N}^{\ell_k \uparrow}(\mathbf{h}_k).$$

If the output of the root network is a scalar, we can interpret the fibred network as a classifier in the usual way: on input $\mathbf{x}$, it outputs *true* if the final value is strictly greater than 0, and *false* otherwise.

5.2. Exact correspondence between fibring logics and fibring networks

In this section, we introduce a fibred modal language for combining different modal systems and show it captures exactly the behavior of fibred neural networks.

Modal logics and fibring (Gabbay 1999): Fix a finite set of propositions $PROP = \{p_1, \dots, p_{d_0}\}$ and a countable collection of modal operators $\Box_i$ (one for each index i). For each i , logic $\mathcal{L}_i$ has formulas built from propositions, $\top$, Boolean connectives, and the modal operators $\Box_i$. Semantics is the usual Kripke semantics: each $\mathcal{L}_i$ uses a class of Kripke models, and $\Box_i \varphi$ holds at a world in a model when φ holds at all accessible worlds.

To *fibre* these logics, we allow all the modal operators $\Box_i$ in one combined language. That is, formulas are built from the grammar

$$\varphi ::= p \mid \top \mid \varphi_1 \wedge \varphi_2 \mid \neg \varphi \mid \Box_i \varphi \quad (5.1)$$

where p is any proposition and i ranges over the fibred modal logics $\mathcal{L}_i$.

A *fibred model* chooses, for each i , one Kripke model for $\mathcal{L}_i$, and also gives a way to *jump* from worlds of other models into the model for i . Intuitively, evaluating $\Box_j \varphi$ at a world

5. Fibring: a formalism connecting neural networks and logical theories

in model i either uses the native accessibility for the model i if $j = i$, or first jumps into model j and then evaluates there. (We assume the model classes are disjoint so each world's home model is unambiguous and we also assume all component Kripke models finite). Formally, a fibred model $\mathcal{M}$ consists of one Kripke model $\mathbf{m}_i$ for each logic $\mathcal{L}_i$, together with a family of logical *fibring functions* f_i . Each f_i maps each world in another model $\mathbf{m}_j$ ($j \neq i$) to a world in $\mathbf{m}_i$. Intuitively, f_i tells us how to jump from another component into the i th one.

Satisfaction in a fibred model $\mathcal{M}$ is defined as follows. If w is a world in $\mathbf{m}_i$, then:

- $\mathcal{M}, w \models p$ iff w is in the valuation of p in $\mathbf{m}_i$.
- $\mathcal{M}, w \models \top$ always.
- $\mathcal{M}, w \models \varphi_1 \wedge \varphi_2$ iff $\mathcal{M}, w \models \varphi_1$ and $\mathcal{M}, w \models \varphi_2$.
- $\mathcal{M}, w \models \neg\varphi$ iff not $\mathcal{M}, w \models \varphi$.
- $\mathcal{M}, w \models \Box_j \varphi$ iff (i) $j = i$ and for all w' with (w, w') in the accessibility relation of $\mathbf{m}_i$, we have $\mathcal{M}, w' \models \varphi$; or (ii) $j \neq i$ and $\mathcal{M}, f_j(w) \models \Box_j \varphi$.

A fibred logic for a fibring architecture: Let $\mathcal{F}$ be a fibring architecture. For each node v in $\mathcal{F}$ and each layer number ℓ of $\mathbb{A}_v$, associate a distinct modal operator $\Box_{v,\ell}$ interpreted over the class of all finite Kripke models; let $\mathcal{L}$ be the resulting fibred logic. Based on $\mathcal{L}$ we will define another fibred logic $\mathcal{L}_{\mathcal{F}}$ based on the notion of "compatible" models.

Definition 5.1. Let $\mathbf{m}$ be a Kripke model and $\mathcal{N}$ a network instance. A map π that assigns to each world of $\mathbf{m}$ an input vector for $\mathcal{N}$ is $(\mathbf{m}, \mathcal{N})$ -admissible if it is injective and, for all worlds w, w' ,

$$w \text{ is related to } w' \text{ in } \mathbf{m} \iff \mathcal{N}(\pi(w)) = \mathcal{N}(\pi(w')).$$

Definition 5.2. Let $\mathcal{M}$ be a fibred model of $\mathcal{L}$ and let us denote with $\mathbf{m}_{v,\ell}$ the component Kripke model in $\mathcal{M}$ for node v in $\mathcal{F}$ and layer number ℓ in $\mathbb{A}_v$. Let $\tilde{\mathcal{N}} = \langle \mathcal{N}, \mathcal{F}, \tilde{f} \rangle$, let $\mathbf{x} \in \{0, 1\}^{d_0}$, and let u be the root of $\mathcal{F}$. We say that $\mathcal{M}$ is *compatible* with $(\tilde{\mathcal{N}}, \mathbf{x})$ if we can assign:

- to each node v in $\mathcal{F}$, a network instance $\mathcal{N}_v$ of $\mathbb{A}_v$, and
- to each pair (v, ℓ) of a node in $\mathcal{F}$ and ℓ a layer number in $\mathbb{A}_v$, a bijection $\pi_{v,\ell}$ from a finite set of Kripke worlds to a finite set of vectors

such that the following compatibility conditions hold:

- (C0) $\mathcal{N}_u = \mathcal{N}$ and $\pi_{u,1}$ maps each world w in $\mathbf{m}_{u,1}$ to the vector in $\{0, 1\}^{d_0}$ whose i -th bit is 1 exactly when proposition p_i is true at w .
- (C1) $\pi_{v,\ell}$ is $(\mathbf{m}_{v,\ell}, \mathcal{N}_v^{\ell\uparrow})$ -admissible.
- (C2) If v has children $v_1, \dots, v_k$, ordered by the layer numbers $\ell_1, \dots, \ell_k$ labelling edges (v, v_i) , and let w be the world reached via the composition of logical fibring functions from $(\pi_{u,1})^{-1}(\mathbf{x})$ to Kripke model $\mathbf{m}_{v,1}$. Running $\langle \mathcal{N}_v, \mathcal{F}_v, \tilde{f}_v \rangle$ on $\pi_{v,1}(w)$ produces tuples $(\mathbf{x}_i, \mathcal{N}_i, \mathbf{y}_i, \mathbf{h}_i)$ for $1 \leq i \leq k$. Then:

5. Fibring: a formalism connecting neural networks and logical theories

1. $\mathcal{N}_{v_i} = \mathcal{N}_i$ and $\mathbf{y}_i = \pi_{v_i,1}(f_{v_i,1}(w))$.
2. $\mathbf{h}_i = \pi_{v,\ell_i}(f_{v,\ell_i}(w))$.
3. the set of propositions true at world $\pi_{v,\ell_k}^{-1}(\mathbf{h}_k)$ in model $\mathbf{m}_{v,\ell_k}$ agrees with the sets of propositions true at $\pi_{v_i,1}^{-1}(\mathbf{y}_i)$ in model $\mathbf{m}_{v_i,1}$ for $1 \leq i \leq k$.

For a fixed pair (v, ℓ) of a node v in $\mathcal{F}$ and a layer number ℓ in $\mathbb{A}_v$, let

$$\text{Comp}_{\mathcal{F}}(v, \ell) := \left\{ \mathbf{m} \mid \begin{array}{l} \exists \widetilde{\mathcal{N}}, \mathbf{x}, \mathcal{M} \text{ a fibred model of } \mathcal{L} \text{ compatible with } (\widetilde{\mathcal{N}}, \mathbf{x}) \\ \text{such that the } (v, \ell)\text{-component of } \mathcal{M} \text{ is } \mathbf{m} \end{array} \right\}.$$

In words, $\text{Comp}_{\mathcal{F}}(v, \ell)$ is the *projection* of all compatible fibred models onto their (v, ℓ) component.

Proposition 5.3. *The class $\text{Comp}_{\mathcal{F}}(v, \ell)$ is non-empty and closed under Kripke-model isomorphism.*

Proof. Non-empty. The class $\text{Comp}_{\mathcal{F}}(v, \ell)$ contains at least the Kripke models of fibred models obtained by explicitly running the computation of fibred networks $\widetilde{\mathcal{N}}$ on vectors $\mathbf{x}$.

Specifically, for each $\widetilde{\mathcal{N}} = \langle \mathcal{N}, \mathcal{F}, \tilde{f} \rangle$ and $\mathbf{x}$, we can construct a compatible fibred model as follows.

We start from the root model: the domain of the root model contains a distinct element $w_{\mathbf{z}}$ for each vector $\mathbf{z} \in \{0, 1\}^{d_0}$, two worlds $w_{\mathbf{z}}, w_{\mathbf{z}'}$ are related iff $\mathcal{N}(\mathbf{z}) = \mathcal{N}(\mathbf{z}')$, and the valuation is given by: p_j holds at world $w_{\mathbf{z}}$ in the root model iff $z_j = 1$.

We run the fibred neural network $\widetilde{\mathcal{N}}$ on all vectors $\mathbf{z} \in \{0, 1\}^{d_0}$ (recursively generating sequences of tuples $(\mathbf{x}_i, \mathcal{N}_i, \mathbf{y}_i, \mathbf{h}_i)$) and we collect at each (v, ℓ) all vectors attained by the computations, that is, for each (v, ℓ_i) we collect vectors $\mathbf{h}_i$ and for each $(v_i, 1)$, we collect vectors $\mathbf{y}_i$. Furthermore, we run the fibred neural network $\widetilde{\mathcal{N}}$ on the specific vector $\mathbf{x}$ and we collect at each v_i , the network instance $\mathcal{N}_i$ attained by the computation.

We then define Kripke models as follows: the domain of $\mathbf{m}_{v,\ell}$ contains a distinct element $w_{\mathbf{v}}$ for each vector $\mathbf{v}$ collected at (v, ℓ) , two worlds $w_{\mathbf{v}}, w_{\mathbf{v}'}$ are related iff $\mathcal{N}_v^{\ell\uparrow}(\mathbf{v}) = \mathcal{N}_v^{\ell\uparrow}(\mathbf{v}')$ where $\mathcal{N}_v$ is the network instance collected at v .

The valuations are defined recursively starting from the root model: a proposition p holds at world $w_{\mathbf{h}}$ in model $\mathbf{m}_{v,\ell_i}$ iff there is a world $w_{\mathbf{v}}$ in $\mathbf{m}_{v,1}$ such that $\mathbf{h} = \mathbf{h}_i$ is attained by the computation of $\langle \mathcal{N}_v, \mathcal{F}_v, \tilde{f}_v \rangle(\mathbf{v})$ and p holds at $w_{\mathbf{v}}$ in model $\mathbf{m}_{v,1}$; and a proposition p holds at world $w_{\mathbf{y}}$ in model $\mathbf{m}_{v_i,1}$ iff there is a world $w_{\mathbf{v}}$ in $\mathbf{m}_{v,1}$ such that $\mathbf{y} = \mathbf{y}_i$ is attained by the computation of $\langle \mathcal{N}_v, \mathcal{F}_v, \tilde{f}_v \rangle(\mathbf{v})$ and p holds at $w_{\mathbf{h}_k}$ in model $\mathbf{m}_{v,\ell_k}$.

Finally, we require that the (logical) fibring function from $\mathbf{m}_{v,1}$ to $\mathbf{m}_{v,\ell_i}$ verifies for each world $w_{\mathbf{v}}$ in $\mathbf{m}_{v,1}$, $f_{v,\ell_i}(w_{\mathbf{v}}) = w_{\mathbf{h}_i}$ where $\mathbf{h}_i$ is the vector attained by the computation of $\langle \mathcal{N}_v, \mathcal{F}_v, \tilde{f}_v \rangle(\mathbf{v})$; and the (logical) fibring function from $\mathbf{m}_{v,1}$ to $\mathbf{m}_{v_i,1}$ verifies, for each world $w_{\mathbf{v}}$ in $\mathbf{m}_{v,1}$, $f_{v_i,1}(w_{\mathbf{v}}) = w_{\mathbf{y}_i}$ where $\mathbf{y}_i$ is the vector attained by the computation of $\langle \mathcal{N}_v, \mathcal{F}_v, \tilde{f}_v \rangle(\mathbf{v})$.

5. Fibring: a formalism connecting neural networks and logical theories

The above proof construction covers all compatibility conditions and ensures that the resulting fibred model is compatible with $(\tilde{\mathcal{N}}, \mathbf{x})$ by taking the bijections $\pi_{v,\ell} : w_v \mapsto \mathbf{v}$.

Closed under Kripke-model isomorphism. Let $\mathbf{m} \in \text{Comp}_{\mathcal{F}}(v, \ell)$ come from some compatible $\mathcal{M}$, and let $\pi : \mathbf{m} \cong \mathbf{m}'$ be an isomorphism. Form a new fibred model $\mathcal{M}'$ by replacing the (v, ℓ) -component of $\mathcal{M}$ with $\mathbf{m}'$ and replace the bijection $\pi_{v,\ell}$ by $\pi_{v,\ell} \circ \pi^{-1}$. This preserves (C0)–(C2): edges, valuations, and fibring jumps are transported through π without changing any observable behavior. Hence $\mathcal{M}'$ is still compatible with the same $(\tilde{\mathcal{N}}, \mathbf{x})$, so $\mathbf{m}' \in \text{Comp}_{\mathcal{F}}(v, \ell)$. $\square$

This shows that the subclass of compatible models is a valid fibred logic, since the projections of compatible models represent a non-empty class of Kripke models closed under isomorphism (hence a reasonable class of Kripke models). We refer to this fibred logic as $\mathcal{L}_{\mathcal{F}}$.

Equivalence between fibring neural networks and a fragment of fibred logic.

Definition 5.4. Let $\mathcal{F}$ be a fibring architecture and let φ be a propositional formula. The formula $\psi(\varphi, \mathcal{F}) \in \mathcal{L}_{\mathcal{F}}$ is recursively defined as follows:

- If $\mathcal{F}$ has only a single node, then $\psi(\varphi, \mathcal{F}) = \varphi$.
- Otherwise, let $v_1, \dots, v_k$ be the children of the root, with corresponding subtrees $\mathcal{F}_1, \dots, \mathcal{F}_k$. For each i , let l_i be the largest layer label on an edge leaving v_i (or $l_i = 1$ if none). Then set

$$\psi(\varphi, \mathcal{F}) = \Box_{v_1, l_1} \psi(\varphi, \mathcal{F}_1) \wedge \dots \wedge \Box_{v_k, l_k} \psi(\varphi, \mathcal{F}_k).$$

Theorem 5.5. Let $\mathcal{F} \in \mathcal{F}$ be a fibring architecture rooted at u . For every network instance $\mathcal{N}$ of the root architecture $\mathbb{A}_u$, there is a propositional formula φ such that, for every input $\mathbf{x} \in \{0, 1\}^{d_0}$, for every collection of fibring functions $\tilde{f}$ matching $\mathcal{F}$, and every fibred model $\mathcal{M}$ compatible with $\langle \mathcal{N}, \mathcal{F}, \tilde{f} \rangle$ and $\mathbf{x}$, we have

$$\mathcal{M}, (\pi_{u,1})^{-1}(\mathbf{x}) \models \psi(\varphi, \mathcal{F}) \quad \text{iff} \quad \langle \mathcal{N}, \mathcal{F}, \tilde{f} \rangle \text{ classifies } \mathbf{x} \text{ as true}.$$

Proof idea. In a compatible fibred model, the accessibility relations between worlds mirror the behavior of network instances on their inputs (condition C1), and the fibring functions on worlds mirror the way a parent network delegates part of its computation to children (condition C2). Thus evaluating the fibred network on $\mathbf{x}$ corresponds exactly to checking truth of the formula $\psi(\varphi, \mathcal{F})$ at the matching world, since both follow the same recursive structure given in Definition 5.4.

Proof. Fix a network instance $\mathcal{N}$ of the root architecture. Define the *characteristic formula* of $\mathcal{N}$ by

$$\varphi := \bigvee_{\mathbf{h} \in \{0,1\}^{d_0}: \mathcal{N}(\mathbf{h}) > 0} \left(\bigwedge_{h_k=1} p_k \wedge \bigwedge_{h_k=0} \neg p_k \right).$$

Fix $\mathcal{F} \in \mathcal{F}$, $\mathbf{x} \in \{0, 1\}^{d_0}$ and $\mathcal{M}$ compatible with $\langle \mathcal{N}, \mathcal{F} \rangle$ and $\mathbf{x}$. We prove by induction on the depth of $\mathcal{F}$ that, for the world $w = (\pi_{u,1})^{-1}(\mathbf{x})$ at the root,

$$\mathcal{M}, w \models \psi(\varphi, \mathcal{F}) \iff \langle \mathcal{N}, \mathcal{F}, \tilde{f} \rangle \text{ classifies } \mathbf{x} \text{ as true}.$$

5. Fibring: a formalism connecting neural networks and logical theories

Base case (leaf). If $\mathcal{F}$ has only the root, then $\psi(\varphi, \mathcal{F}) = \varphi$ and by (C0), at the root model $\mathbf{m}_{u,1}$, φ satisfies

$$\mathbf{m}_{u,1}, \pi_{u,1}^{-1}(\mathbf{h}) \models \varphi \iff \mathcal{N}(\mathbf{h}) > 0 \quad (\mathbf{h} \in \{0,1\}^{d_0}).$$

Induction step. Suppose the root has children $u_1, \dots, u_k$ with labels (ℓ_i, S_i) , subtrees $\mathcal{F}_i$ and $\tilde{f}_i$ restrictions of f to $\mathcal{F}_i$. Run the root until layer ℓ_1 , obtain $\mathbf{x}_1$; fibre to u_1 by $(\mathcal{N}_1, \mathbf{y}_1) = \tilde{f}_{(u, u_1)}(\mathbf{x}_1)$; run the child fibred network $\langle \mathcal{N}_1, \mathcal{F}_1, \tilde{f}_1 \rangle$ on $\mathbf{y}_1$; splice its output back to get $\mathbf{h}_1$; and continue similarly for $i = 2, \dots, k$. By (C1)–(C2) and the semantics of $\square_{v,\ell}$, evaluating $\psi(\varphi, \mathcal{F}) = \bigwedge_i \square_{v_i, \ell_i} \psi(\varphi, \mathcal{F}_i)$ at w amounts to evaluating each $\psi(\varphi, \mathcal{F}_i)$ at the world reached by the corresponding fibring jump. By the induction hypothesis, each of those subformulas is true exactly when the corresponding child computation returns the vector that is spliced into the parent. Hence $\mathcal{M}, w \models \psi(\varphi, \mathcal{F})$ iff the overall fibred computation yields a final root output > 0 , i.e., iff $\langle \mathcal{N}, \mathcal{F}, f \rangle$ classifies $\mathbf{x}$ as true. $\square$

5.3. Application to non-uniform expressiveness of GNNs, GATs and Transformer encoders

5.3.1. Formal definitions of GNNs and Transformer Encoders.

Graph Neural Networks. A Graph Neural Network (GNN) for neural architecture $\mathbb{A}$ is a tuple $\mathcal{G} = \langle \{\mathbf{A}^\ell\}_{\ell=1}^L, \{\mathbf{B}^\ell\}_{\ell=1}^L, \{\mathbf{b}^\ell\}_{\ell=1}^L \rangle$. For each layer ℓ , matrices $\mathbf{A}^\ell \in \mathbb{Q}^{d_\ell \times d_{\ell-1}}$ and $\mathbf{B}^\ell \in \mathbb{Q}^{d_\ell \times d_{\ell-1}}$ are weight matrices and vector $\mathbf{b}^\ell \in \mathbb{Q}^{d_\ell}$ is a bias vector. A Graph Attention Network (GAT) $\mathcal{G}'$ for $\mathbb{A}$ contains all the elements of a GNN plus an additional collection $\{\mathbf{a}^\ell\}_{\ell=1}^L$ of attention vectors, where $\mathbf{a}^\ell \in \mathbb{Q}^{2 \cdot d_\ell}$. Both GNNs and GATs apply to undirected graphs $G = (V, E)$ where a node vector $\mathbf{x}_u \in \{0,1\}^{d_0}$ is associated to each node $u \in V$ and where the neighbourhood of node u is defined as $N(u) = \{v \in V : \{u, v\} \in E\}$. Their application to graph G with node vectors $\mathbf{x} := (\mathbf{x}_u)_{u \in V}$ yields a sequence $(\mathbf{x}_u^1)_{u \in V}, \dots, (\mathbf{x}_u^L)_{u \in V}$ of node vectors and the final result is a graph with the same nodes and edges as G but with updated node vectors $\{\mathbf{x}_u^L\}_{u \in V}$. When there is no ambiguity, we will refer to $\mathbf{x}_u^L$ as $\mathcal{G}(G, \mathbf{x}, u)$, the result at node u of applying GNN (or GAT) $\mathcal{G}$ to graph G with node features $\mathbf{x}$.

The application of GNN $\mathcal{G}$ to $G, \mathbf{x}$ is defined, for each $u \in V$, as $\mathbf{x}_u^\ell = \sigma^\ell(\mathbf{h}_u^\ell)$, where $\mathbf{x}_u^0 = \mathbf{x}_u$ and

$$\mathbf{h}_u^\ell = \mathbf{B}^\ell \cdot \mathbf{x}_u^{\ell-1} + \sum_{v \in N(u)} \mathbf{A}^\ell \cdot \mathbf{x}_v^{\ell-1} + \mathbf{b}^\ell \quad (5.2)$$

The application of GAT $\mathcal{G}'$ to G is defined analogously by replacing expression (5.2) with

$$\mathbf{h}_u^\ell = \alpha_{uu}^\ell \cdot \mathbf{B}^\ell \cdot \mathbf{x}_u^{\ell-1} + \sum_{v \in N(u)} \alpha_{uv}^\ell \cdot \mathbf{A}^\ell \cdot \mathbf{x}_v^{\ell-1} + \mathbf{b}^\ell \quad (5.3)$$

where the *attention coefficient* α_{uv}^ℓ is the component associated to node v in the following vector:

$$\text{hardmax}\{\mathbf{a}^{\ell T} \cdot (\mathbf{A}^\ell \cdot \mathbf{x}_w^{\ell-1} || \mathbf{B}^\ell \cdot \mathbf{x}_u^{\ell-1})\}_{w \in N(u) \cup \{u\}}. \quad (5.4)$$

5. Fibring: a formalism connecting neural networks and logical theories

The hardmax function applied to a vector sets all occurrences of its largest value to the inverse of the number of its occurrences in the vector and all remaining components to 0. The expression within the hardmax function is a vector with a component for each node $w \in N(u) \cup \{u\}$ and where each component is computed as the dot product of two vectors.

Transformer Encoders with Hard Attention. A Transformer encoder $\mathcal{T}$ for architecture $\mathbb{A}$ is defined similarly to a GAT as $\mathcal{T} = \langle \{\mathbf{A}^\ell\}_{\ell=1}^L, \{\mathbf{B}^\ell\}_{\ell=1}^L, \{\mathbf{b}^\ell\}_{\ell=1}^L, \{\mathbf{a}^\ell\}_{\ell=1}^L \rangle$ but is applicable only to specific types of input. While GATs are applied to arbitrary graphs, Transformer encoders are restricted to complete graphs, where each node is connected to every other node (Bronstein et al. 2021). Additionally, the input graph for $\mathcal{T}$ is constructed from an ordered sequence of tokens, with each token assigned a vector representation, which is then combined with a positional encoding vector to construct a token feature. More precisely, for a complete graph with s nodes associated to a token sequence S of length s , each node is uniquely associated to a token ξ at position $t \in \{0, \dots, s-1\}$ in S and each token is associated to a token feature $\mathbf{x}_\xi$ which is obtained as the sum $\mathbf{x}_\xi = \text{vec}(\xi) + \text{pos}(t, s)$, where vec is a vector representation function mapping tokens to vectors in $\{0, 1\}^{d_0}$ and pos is a positional encoding function mapping pairs of positions and sequence length to vectors in $\mathbb{Q}^{d_0}$. The application of $\mathcal{T}$ to S and token features $\mathbf{x} := (\mathbf{x}_\xi)_{\xi \in S}$ is then obtained by applying expressions (5.3) and (5.4) to the complete graph with node features $\{\mathbf{x}_\xi\}_{\xi \in S}$. We will also note $\mathcal{T}(S, \mathbf{x}, \xi)$, the result at token ξ of applying transformer encoder $\mathcal{T}$ to sequence S with token features $(\mathbf{x}_\xi)_{\xi \in S}$, i.e. $\mathcal{T}(S, \mathbf{x}, \xi)$ is the application of transformer encoder $\mathcal{T}$ to $(S, \mathbf{x}, \xi)$.

Section-specific assumptions. In this section, we restrict ourselves to GNNs and GATs with local sum aggregation, rational coefficients, and applied to Boolean vectors. Our technical results also assume that each σ^ℓ is the truncated ReLU function mapping each component x_i of $\mathbf{x}$ to $\min(1, \max(0, x_i))$. We also consider GATs and Transformers with hard attention only. These restrictive assumptions are the same as the assumptions taken in Benedikt et al. (2024) and Barcelo et al. (2024) to render GNNs and Transformers amenable to logical characterisation, and tractable (for example, hard attention is the discrete version of soft attention). To avoid repetitions, the technical results in this section will be stated for GATs, but they easily extend to GNNs and Transformer encoders, by considering that a GNN is a GAT without attention, and a Transformer encoder is a GAT on complete graphs with positional encodings.

If the output dimension of the GNN or GAT (respectively, Transformer encoder) is 1, we can interpret them as *node* (respectively, *token*) *classifiers*: on input $(G, \mathbf{x})$ (respectively, $(S, \mathbf{x})$), for each node u (respectively, for each token ξ), it outputs *true* if $\mathbf{x}_u^L > 0$ (respectively, $\mathbf{x}_\xi^L > 0$) and *false* otherwise. By abuse of notation, we sometimes note $\mathcal{G}(G, \mathbf{x}, u) = \text{true}$ or *false* (respectively, $\mathcal{T}(S, \mathbf{x}, \xi) = \text{true}$ or *false*).

5.3.2. Fibred neural networks can non-uniformly reproduce GNNs, GATs and Transformer encoders

The following result establishes that fibred neural networks provide a non-uniform description of GATs, i.e. given a GAT $\mathcal{G}$, for each input G, u, x there is a different fibred neural network whose computation coincides with that of $\mathcal{G}$. Interestingly, for a given input (G, u) , the fibring architecture is the same, and only the fibring functions vary for different node features x . This property is key to derive the (non-uniform) logical expressiveness result in the next section.

Theorem 5.6. *For every tuple $\tau = \langle \mathcal{G}, G, u \rangle$, where $\mathcal{G}$ is a GAT with input/output dimensions $d_0/1$, $G = (V, E)$ a graph, and $u \in V$, there exist a fibring architecture $\mathcal{F}^\tau$, a network instance $\mathcal{N}^\tau$ of the root architecture and a family $\mathcal{G}_\tau = \{\tilde{f}_x^\tau\}_x$ of collections of fibring functions matching $\mathcal{F}^\tau$, indexed by possible node features $x = (x_v)_{v \in V}$, such that, for all x , the fibred neural network $\langle \mathcal{N}^\tau, \mathcal{F}^\tau, \tilde{f}_x^\tau \rangle$ applied to x_u computes $\mathcal{G}(G, x, u)$.*

Proof idea. For each tuple $\tau = \langle \mathcal{G}, G, u \rangle$, the tree structure of the corresponding fibring architecture is given by the unravelling tree of node u in G at the depth of $\mathcal{G}$. Indeed, a GAT computes recursively w.r.t. the depth of its unravelling tree, as do fibred neural networks w.r.t. the depth of their fibring architecture. At each node, the fibring function returns the node features of the neighbouring nodes and assigns the weights of the relevant layer of $\mathcal{G}$. In the computation of the fibred neural network, the step of "replacing the entries in S_i of the vector x_i by $\langle \mathcal{N}_i, \mathcal{F}_i, \tilde{f}_i \rangle(y_i)$ " is used to concatenate the outputs of the computations of the previous layer. The concatenation of outputs at the previous layer is then aggregated (or the attention layer is computed), using the relevant weights (or attention vectors) assigned by the fibring function upstream of the concatenation.

Proof. Let $\tau = \langle \mathcal{G}, G, u \rangle$ where $\mathcal{G} = \langle \{\mathbf{A}^\ell\}_{\ell=1}^L, \{\mathbf{B}^\ell\}_{\ell=1}^L, \{\mathbf{b}^\ell\}_{\ell=1}^L, \{\mathbf{a}^\ell\}_{\ell=1}^L \rangle$ is a GAT instance, $G = (V, E)$ is a graph with maximum degree m and u is a node in G .

Define $\mathcal{N}^\tau$ as the two-layer linear network whose first layer is the concatenation of $m + 1$ identity matrices (with no bias) and the second layer is the concatenation of the matrix $\mathbf{B}^L$, and m times the matrix $\mathbf{A}^L$ (with bias $\mathbf{b}^L$).

Definition 5.7. Let $G = (V, E)$ be a graph. A *lazy walk* in G is a finite sequence of nodes $(u_0, \dots, u_k)$ such that for each $i \in \{1, \dots, k\}$ $u_{i-1} = u_i$ or $(u_{i-1}, u_i) \in E$. A lazy walk differs from a *path* in that one can stay on the same node multiple times.

Definition 5.8. Let $G = (V, E)$ be a graph, $u \in V$, and $L \in \mathbb{N}$. The (lazy) unravelling of node u in G at depth L is the graph that is the tree having:

- a root u
- a node $(u, u_1, \dots, u_\ell)$ for each lazy walk $(u, u_1, \dots, u_\ell)$ in G with $1 \leq \ell \leq L$, and
- an edge between $(u, u_1, \dots, u_{\ell-1})$ and $(u, u_1, \dots, u_\ell)$ when $u_{\ell-1} = u_\ell$ or $(u_{\ell-1}, u_\ell) \in E$ (assuming that u_0 is u).

The fibring architecture $\mathcal{F}^\tau$ is constructed from the lazy unravelling of u in G at depth L to which we add (in order to encode the attention mechanism) some nodes and

5. Fibring: a formalism connecting neural networks and logical theories

edges in the following way: for each $\ell \in \{1 \dots L\}$ (assuming that u_0 is u), each node $(u, u_1, \dots, u_{\ell-1})$ has an extra child node v_ℓ , and we require that each node v_ℓ has no child node. Note that by this construction, each node $(u, u_1, \dots, u_\ell)$ has $|\mathbf{N}(u_\ell)| + 2$ children nodes (namely the nodes $(u, u_1, \dots, u_\ell, w)$ for $w \in \{u_\ell\} \cup \mathbf{N}(u_\ell)$, and the node $v_{\ell+1}$).

Let x be node features for G .

The label of the edge between $(u, u_1, \dots, u_{\ell-1})$ and $(u, u_1, \dots, u_\ell)$ is defined by layer number 1 and the set of positions designed so the final output vector is the concatenation of the children computations. Its fibring function is as follows: on input the node feature $\mathbf{x}_{u_\ell}^0$, it returns the concatenation of vectors $\mathbf{x}_w^0$ for $w \in \{u_\ell\} \cup \mathbf{N}(u_\ell)$ and a network with the relevant weights of layer $L - \ell$ in $\mathcal{G}$ designed so that, in the computation of the fibred network, the network $\mathcal{N}_{(u, u_1, \dots, u_\ell)}$ applied to the concatenation of vectors $\mathbf{h}_w^{L-\ell+1}$ for $w \in \{u_\ell\} \cup \mathbf{N}(u_\ell)$ returns the concatenation of $\mathbf{A}^{L-\ell} \cdot \sigma(\mathbf{h}_w^{L-\ell+1})$ (and $\mathbf{B}^{L-\ell} \cdot \sigma(\mathbf{h}_w^{L-\ell+1})$ when $w = u_\ell$).

The label of the edge between $(u, u_1, \dots, u_{\ell-1})$ and v_ℓ is defined by layer number 2 and the set of positions is all positions of layer 2. Its fibring function takes as input a vector $\mathbf{y}$ and returns the same vector $\mathbf{y}$ (*self-fibring*) and a network using the relevant attention vector $\mathbf{a}^{L-\ell}$ (and the bias) of layer $L - \ell$ designed so that when $\mathbf{y}$ is the concatenation of vectors $\mathbf{y}_w^{L-\ell}$ for $w \in \{u_\ell\} \cup \mathbf{N}(u_\ell)$, then $\mathcal{N}_{v_\ell}(\mathbf{y})$ is the sum of vectors $\alpha_{u_\ell w}^{L-\ell} \cdot \mathbf{y}_w^{L-\ell}$ (plus the bias $\mathbf{b}^{L-\ell}$).

The neural architectures labelling nodes have the dimensions and nonlinearities required to realise the networks described above (in particular, the neural architectures labelling nodes v_ℓ uses the hardmax nonlinearity), and it is easy to see that the set of neural architectures labelling nodes, layer numbers, and sets of positions labelling edges are fully determined by τ (in particular, they do not depend on x).

As a result, since v_ℓ is a leaf node, it follows that, in the computation of the fibred network, the output vector at node $(u, u_1, \dots, u_{\ell-1})$ is given by $\mathcal{N}_{v_\ell}(\mathbf{y})$ where $\mathbf{y}$ is the concatenation of the children computations, i.e. from nodes $(u, u_1, \dots, u_{\ell-1}, u_\ell)$.

We prove by (descending) induction on the depth that the fibred network $\tilde{\mathcal{N}} = \langle \mathcal{N}^\tau, \mathcal{F}^\tau, \tilde{f}_x^\tau \rangle$ verifies $\tilde{\mathcal{N}}(\mathbf{x}_u^0) = \mathbf{h}_u^L$.

Base ($L - 1$): By construction, each child computation is given by the concatenation of $\mathbf{A}^1 \cdot \mathbf{x}_w^0$ (and $\mathbf{B}^1 \cdot \mathbf{x}_w^0$) for $w \in \{u_{L-1}\} \cup \mathbf{N}(u_{L-1})$, where u_{L-1} is a neighbour reached at depth $L - 1$ starting from the root u . Applying $\mathcal{N}_{v_L}$ to the concatenation of these children computations yields the sum of vectors $\alpha_{u_{L-1}w}^1 \cdot \mathbf{A}^1 \cdot \mathbf{x}_w^0$ (plus the bias $\mathbf{b}^1$), which is $\mathbf{h}_{u_{L-1}}^1$.

Induction step ($\ell \rightarrow \ell - 1$): By inductive hypothesis, the output vector at each node $(u, u_1, \dots, u_\ell)$ is $\mathbf{h}_{u_\ell}^{L-\ell}$ where u_ℓ is a neighbour reached at depth ℓ starting from the root u . Computing the output at node $(u, u_1, \dots, u_{\ell-1})$: by construction, each child computation is thus given by the concatenation of $\mathbf{A}^{L-\ell+1} \cdot \sigma(\mathbf{h}_w^{L-\ell})$ (and $\mathbf{B}^{L-\ell+1} \cdot \sigma(\mathbf{h}_w^{L-\ell})$) for $w \in \{u_{\ell-1}\} \cup \mathbf{N}(u_{\ell-1})$. Applying $\mathcal{N}_{v_\ell}$ to the concatenation of these children computations yields the sum of vectors $\alpha_{u_{\ell-1}w}^{L-\ell+1} \cdot \mathbf{A}^{L-\ell+1} \cdot \sigma(\mathbf{h}_w^{L-\ell})$ (plus the bias $\mathbf{b}^{L-\ell+1}$), which is $\mathbf{h}_{u_{\ell-1}}^{L-\ell+1}$.

□

The same result holds for GNNs and Transformer encoders simply by removing the

5. Fibring: a formalism connecting neural networks and logical theories

nodes and edges in the fibring architecture $\mathcal{F}^\tau$ that encode the GAT attention mechanism and from the fact that a Transformer encoder is a GAT applied to a complete graph with positional encoding incorporated into token features $(\mathbf{x}_\xi^0)_{\xi \in S}$.

5.3.3. GNNs, GATs and Transformer encoders as fragments of fibred logics

The following result establishes a non-uniform expressiveness result for GATs, characterised by a countably infinite family of formulas in the fragment of the fibred logic of interest.

Theorem 5.9. *Let $\tau = \langle \mathcal{G}, G, u \rangle$, $\mathcal{N}^\tau$, $\mathcal{F}^\tau$ and $\mathcal{G}_\tau = \{\tilde{f}_x^\tau\}_x$ be as in Theorem 5.6. Denote u^τ the root of $\mathcal{F}^\tau$. There exists a formula $\tilde{\varphi}^\tau$ in $\mathcal{L}_{\mathcal{F}^\tau}$ such that for each $\mathbf{x} = (\mathbf{x}_u)_{u \in V}$, for each fibred model $\mathcal{M}$ compatible with $\langle \mathcal{N}^\tau, \mathcal{F}^\tau, \tilde{f}_x^\tau \rangle$ and $\mathbf{x}_u$:*

$$\mathcal{M}, (\pi_{u^\tau, 1})^{-1}(\mathbf{x}_u) \models \tilde{\varphi}^\tau \quad \text{iff} \quad \mathcal{G}(G, \mathbf{x}, u) = \text{true}$$

Proof idea. It suffices to consider the fibred neural networks which reproduce the computations of $\mathcal{G}$ and to invoke the correspondence between fibred neural networks and our fibred logic. Since the formulas in our fibred logic only depend on the network at the root, and the fibring architecture, it follows that the formula does not depend on the node features.

Proof. Let $\tau = \langle \mathcal{G}, G, u \rangle$ be a tuple with $\mathcal{G}$ a GAT with input/output dimensions $n/1$, G a graph and u a node in G . For each $\mathbf{x}$ node features for G , consider a fibred neural network $\tilde{\mathcal{N}}_x^\tau = \langle \mathcal{N}^\tau, \mathcal{F}^\tau, \tilde{f}_x^\tau \rangle$ obtained from Theorem 5.6.

By Theorem 5.5 applied to $\mathcal{N}^\tau$ and $\mathcal{F}^\tau$, there exists a propositional formula φ^τ such that for each fibred model $\mathcal{M}$ compatible with $\tilde{\mathcal{N}}_x^\tau$ and $\mathbf{x}_u$, $\mathcal{M}, (\pi_{u^\tau, 1})^{-1}(\mathbf{x}_u) \models \psi(\varphi^\tau, \mathcal{F}^\tau)$ iff $\tilde{\mathcal{N}}_x^\tau(\mathbf{x}_u) > 0$. Furthermore by Theorem 5.6 $\tilde{\mathcal{N}}_x^\tau(\mathbf{x}_u) > 0$ iff $\mathcal{G}(G, \mathbf{x}, u) = \text{true}$, which gives us the result. □

The same result also holds for GNNs and Transformer encoders by replacing the fibring architecture $\mathcal{F}^\tau$ obtained from Theorem 5.6. In particular, to incorporate positional encodings for Transformers, it suffices to replace $\pi_{u, 1}$ by the bijection $\pi_{\xi, 1}$ from worlds in $\mathbf{m}_{\xi, 1}$ to $\{\text{pos}(t, s), 1 + \text{pos}(t, s)\}^{d_0}$ where t is the position of token ξ and s is the length of the sequence.

5.4. Discussion

For a given instance $\mathcal{G}$ of a neural architecture (a GNN, a GAT or a Transformer encoder), our non-uniform expressiveness results provide a countable family of formulas that characterize the instance: $(\tilde{\varphi}^\tau)_\tau$ for all tuples $\tau = \langle \mathcal{G}, G, u \rangle$ with the same $\mathcal{G}$. We note that the result of [Cuenca Grau et al. \(2025\)](#) on bounded GNNs also implies a non-uniform expressiveness result in terms of first-order logic formulas: to construct a

5. Fibring: a formalism connecting neural networks and logical theories

countable family of formulas that characterize a GNN $\mathcal{G}$, we can, for example, enumerate for $k \in \mathbb{N}$ the first-order logic formula of the bounded GNN obtained by applying $\mathcal{G}$ to all graphs with degree k .

Closing the gap with uniform expressiveness requires "collapsing", for each instance $\mathcal{G}$, the corresponding family into a single formula in another logic. With the uniform expressiveness result by [Benedikt et al. \(2024\)](#), we already know that, for GNNs, the family must collapse to a Presburger formula. The problem remains open for GATs and Transformer encoders; in particular for Transformer encoders, only a lower bound of the resulting logic is known ([Barcelo et al. 2024](#)).

We now present ideas towards the unification of uniform expressiveness results using fibring. Consider the countable family of fibred neural network $(\langle \mathcal{N}^\tau, \mathcal{F}^\tau, \tilde{f}_x^\tau \rangle)_{\tau, x}$ characterizing a GNN, given in the proof of Theorem 5.6. These fibred neural networks follow a common recursive structure, and by applying them to all possible input vectors $\mathbf{x} \in \{0, 1\}^n$, we can collect at each component (Kripke models of compatible fibred models) the vectors attained by the computations of the fibred neural networks, following the procedure in the proof of Proposition 5.3. Note that, given the correspondence between the structure of the fibring architectures and the layers of the GNN (as seen in the proof of Theorem 5.6), the set of vectors collected by this procedure is closely related to the ℓ -spectrum of the GNN, defined in [Benedikt et al. \(2024\)](#) as the set of all vectors attainable by the GNN at the ℓ -th layer when applied to all possible inputs. In particular, the finiteness of the ℓ -spectrum is a key argument used in the derivation of their uniform expressiveness results. An idea would thus be to identify commonalities between the fibred formulas for different inputs associated to the same GNN, in order to map them to the single Presburger formula. Many technical details must be made precise to close this gap, and we leave this research to future work. Repeating the exercise to find commonalities in the fibred formulas for GATs and Transformer encoders constitutes, in this way, a new approach towards deriving uniform expressiveness results. We argue that the fibring formalism thus provides a unified lens, although still non-uniform at this point, on the logical expressiveness of different network architectures, and might even hold the promise of offering a common methodology to uncover uniform expressiveness results in the future.

The intuition of fibred logics applied to neural architectures, viewing neural networks as combining underlying logics, is appealing and similar in spirit to the current trend in interpretability research, which tries to break down neural network computations into several modular reasoning steps, e.g. ([Ameisen et al. 2025](#); [He et al. 2025](#)). We believe that scrutinizing the commonalities between fibred formulas for typical inputs, thereby reverse-engineering the fibred logical theories learnt by neural networks, constitutes an exciting new take on the extraction of interpretable logical rules capturing neural network's most relevant computations. Therefore, we posit that our novel approach to uniform logical expressiveness via fibring shares common ground with interpretability research, and we hope this thesis will serve as a foundation in this direction. Furthermore, achieving uniform expressiveness results using the fibring formalism could enable new results on formal verification by leveraging existing results on the complexity of fibred logics, e.g. ([Wu et al. 2015](#)).

5.5. Concluding remarks of the chapter

In this chapter, we have formally established the exact correspondence between fibred neural networks and fibred modal logics for the first time. This formalism has allowed us to derive non-uniform logical expressiveness results for several modern neural architectures. A gap remains unresolved to bridge non-uniform and uniform expressiveness results, and we hope our work will inspire further research in this direction. We think fibring as a formalism has the potential to enable the unification of expressiveness results for various network architectures, which were initially derived using different methodologies, and has applications in interpretability and verification.

Part II.

Generalisation of Neural Networks

6. Overview of Generalisation Properties

Modern deep neural networks exhibit remarkable generalisation performance despite being vastly overparameterised, a fact that initially appears to contradict classical statistical wisdom. In traditional statistical learning theory, epitomised by the bias–variance trade-off, one expects models with complexity far exceeding the number of training samples to overfit and perform poorly on new data (Vapnik 1995; Mohri et al. 2012; Shalev-Shwartz & Ben-David 2014). Yet, in practice, large networks often achieve low test error even in regimes where the number of parameters greatly exceeds the sample size (Simonyan & Zisserman 2014). This puzzle has spurred a range of theoretical approaches to explain how neural networks generalise so well. Broadly, these approaches can be grouped into (i) classical statistical learning theory analyses, (ii) statistical physics inspired analyses, (iii) studies of the dynamics of gradient-based training algorithms, and (iv) Bayesian perspectives on learning. In this chapter, we review each of these perspectives, highlighting their insights and the connections and distinctions between them.

6.1. Statistical Learning Theory Approaches

Classical *statistical learning theory* (SLT) provides general bounds on generalisation error via concepts such as VC dimension and uniform convergence (Vapnik 1995; Mohri et al. 2012; Shalev-Shwartz & Ben-David 2014). The contradiction between classical bias-variance tradeoff and deep neural networks was highlighted by Zhang et al. (2016), who showed that deep networks can easily fit randomly permuted labels (memorising data with zero training error), yet the same networks generalise well when trained on real labels. Traditional capacity measures (like VC dimension or Rademacher complexity) are typically too coarse to explain this phenomenon, since they often deem these large networks as universally capable of fitting any data (and thus predict vacuous worst-case generalisation) (Nagarajan & Kolter 2019). This realization has prompted a re-examination of classical statistical learning theory in the context of overparameterised models.

One important observation in modern deep learning is the presence of the *double-descent* risk curve (Belkin et al. 2019; Nakkiran et al. 2021). As model capacity increases, the test error often follows a U-shaped curve (as classical theory predicts) up to the point of *interpolation* (when the model can fit the training data perfectly), but then, beyond this point, the test error may decrease again with further increase in capacity. In particular, for a fixed training set size n , increasing the number of parameters N can lead to a peak in error around $N \approx n$, followed by a second descent as $N \gg n$ so that the most overparameterised models yield the *lowest* generalisation error (Belkin et al. 2019). Classical SLT bounds are not tight in these regimes. Understanding why “more is different” in this context has been a central question.

6. Overview of Generalisation Properties

Contemporary SLT research has pursued several directions to tighten generalisation analyses for neural networks. One line of work refines complexity measures by incorporating properties of the *learned function* rather than raw parameter counts. For instance, margin-based generalisation bounds extend the theory of large margins (well-developed for support vector machines) to deep networks: if a deep network classifies training points with a large margin (confidence), one can bound its test error in terms of network norms and the margin, largely independent of the number of parameters (Bartlett et al. 2017). Neyshabur et al. (2015) and subsequent works provided norm-based capacity controls (e.g. bounds based on the product of spectral norms of weight matrices or path norms), which scale with some notion of network “size” that is more refined than parameter count. These bounds remain formally valid even in the $N \gg n$ regime and have been used to argue that networks trained by gradient descent often find solutions of surprisingly low complexity (low norms, high margins), thereby explaining their good performance (Jiang et al. 2019). However, a common issue is that while such bounds qualitatively align with observed trends (for example, decreasing with increasing margin), their numerical values for realistic networks are still often vacuous (Zhang et al. 2016). This suggests current theory still does not precisely capture the effective capacity of deep networks.

Another approach focuses on the *stability* of the learning algorithm. An algorithm is *uniformly stable* if its output does not change much when a single training sample is removed or resampled (Bousquet & Elisseeff 2002). Stability directly yields generalisation guarantees: roughly, if a model is not overly sensitive to any one data point, its training and test performance will be close. Stochastic Gradient Descent (SGD), the workhorse of deep learning, has been studied under this lens. In convex settings, SGD was shown to be uniformly stable for a bounded number of steps, leading to generalisation bounds proportional to the step size and inversely to the dataset size (Hardt et al. 2016). Yet, applying these results to deep networks (which are non-convex and often trained for many more steps than the sample size) is challenging. Indeed, naive extensions of stability bounds to long-running or non-convex training yield vacuous results when $t \gg n$ (Hardt et al. 2016; Kuzborskij & Lampert 2018). Recent advances, however, demonstrate that introducing additional structure can restore useful stability guarantees. For example, Richards & Kuzborskij (2021) prove non-vacuous bounds for training a *shallow* overparameterised network by showing on-average stability without relying on convexity or Lipschitz assumptions (which are often violated in deep networks). Recent research looks for conditions under which even overparameterised deep networks can be stable, suggesting a new view of overparameterisation: extra parameters need not hurt if the network can internally organise into parts that effectively behave like a simpler model. Still, a general characterisation of stability for deep learning remains an open research frontier; current findings often assume special architectural conditions or rely on empirical observations of implicit low-rank structure in trained networks (Richards & Kuzborskij 2021; Wang & Ma 2023).

Beyond uniform convergence and stability, information-theoretic and Bayesian-inspired methods have enriched SLT. The *PAC-Bayesian* framework (McAllester 1999) provides generalisation bounds for stochastic predictors in terms of a trade-off between training performance and the KL-divergence of the posterior distribution of weights from some prior. PAC-Bayes bounds are especially appealing for understanding deep networks because one can choose a prior centered on “simple” models and consider a posterior

that places mass around the learned weights (plus some noise). By optimising such bounds, Dziugaite & Roy (2017) were able to compute non-vacuous generalisation guarantees for small deep networks, a notable breakthrough at the time. PAC-Bayes bounds have also been linked to flat minima: if the learned weight vector lies in a flat region of the loss landscape, then a Gibbs distribution around it (a “posterior” concentrated in that flat basin) will have low KL divergence to a broad prior, yielding a tight bound (Jiang et al. 2019; Foret et al. 2021). This connection formalises the intuition that *flat minima generalise better* (Hochreiter & Schmidhuber 1997a). However, measuring flatness in a coordinate-invariant way is tricky (Dinh et al. 2017), so PAC-Bayes provides a more principled handle—through the notion of KL divergence—on why flatness correlates with generalisation. Indeed, recent work by Jiang et al. (2019) found that PAC-Bayes bounds based on flatness were among the most predictive measures of a model’s actual generalisation gap across many tasks, compared to dozens of other complexity metrics. In summary, statistical learning theory approaches have evolved to address the deep learning regime by (i) developing capacity measures sensitive to implicit regularisation (norms, margins, flatness), and (ii) considering the interplay of the *algorithm* with the hypothesis class (stability and PAC-Bayes views). While these efforts have substantially improved our qualitative understanding, there remains a gap between the sufficient conditions for generalisation that theory can certify and the actual conditions under which modern networks generalise in practice.

6.2. Statistical Physics Approaches

Complementing the worst-case orientation of SLT, *statistical physics* approaches aim to understand generalisation in neural networks under typical-case assumptions and in various asymptotic limits (such as number of neurons N , width d , training samples n , or input dimension d_0 tending to infinity in fixed ratios). The statistical mechanics of learning was pioneered in the late 1980s for simple connectionist models like the perceptron (Gardner 1988) and two-layer neural networks (Seung et al. 1992). These works introduced the *teacher–student* paradigm: one assumes data are generated by a “teacher” network and studies the performance of a “student” network learning from examples. Using techniques like the replica method and annealed approximations, researchers calculated the expected generalisation error of the student as a function of parameters such as the number of training examples per model parameter. A classic result is the Gardner limit for perceptrons: a perceptron (linear separator) can perfectly classify $\approx 2d$ random patterns in d dimensions before it starts to overfit, and beyond this capacity the generalisation error undergoes a phase transition (Gardner 1988). These analyses described learning curves and phase transitions that qualitatively resembled phenomena later observed in deep learning, such as sharp changes in error when the model becomes just expressive enough to fit the data.

In recent years, statistical physics methods have been instrumental in explaining the double-descent phenomenon and benign overfitting in more complex models. By making simplifying assumptions (e.g. data are high-dimensional Gaussian, networks are heavily overparameterised but random or semi-random in certain ways), one can derive *exact asymptotic formulas* for the generalisation error. A seminal example is ridgeless linear regression on random features or random data: Hastie et al. (2019) computed

6. Overview of Generalisation Properties

the limiting risk of the minimum-norm interpolant in a linear model and demonstrated the presence of a peak in test error at the interpolation threshold followed by a decrease as model complexity grows further. In a similar vein, [Advani et al. \(2020\)](#) and [Mei & Montanari \(2019\)](#) analyzed two-layer neural networks (and related kernel machines) in high dimensions and obtained analytical expressions for the generalisation error as a function of $\alpha = n/d$ (data to model capacity ratio) and other problem parameters. These calculations reproduce the double-descent curve and identify mechanisms for the second descent: in the overparameterised regime ($\alpha \rightarrow 0$), the effective “noise” in the learned function is averaged out by the many redundant parameters, so the generalisation error can decrease even as the model becomes more complex. In other words, overparameterisation can be *benign* or even beneficial ([Bartlett et al. 2020](#)) when the surplus parameters are used in ways that do not increase functional complexity (for instance, by converging to a minimum-norm solution that is simple in function space).

A key tool in bridging neural networks and statistical physics is the correspondence between infinitely wide neural networks and kernel methods. As seen in the preliminaries, in the limit of infinite width (with appropriate variance scaling), a random network’s output converges to a Gaussian Process ([Neal 1996](#)). The covariance function of this GP is the so-called *Neural Network Gaussian Process* (NNGP) kernel associated with the architecture and initialization. Likewise, as one trains a wide network by gradient descent, its evolution can be described by the *Neural Tangent Kernel* (NTK), which remains nearly constant during training for very wide networks ([Jacot et al. 2018](#)). These results imply that sufficiently wide networks essentially perform kernel regression with a fixed kernel (determined by the network architecture at initialization) ([Lee et al. 2018](#); [Matthews et al. 2018](#)). While this “infinite-width” limit is an oversimplification (since real networks are finite and can learn features), it provides a solvable reference point. Notably, infinite-width networks do not exhibit traditional double descent with respect to width — because their performance is equivalent to a fixed kernel method, it typically improves monotonically with training data. However, finite-width networks deviate from this kernel behaviour, and it is precisely these deviations (feature learning, non-Gaussian fluctuations) that statistical physics methods can handle via perturbation analysis or mean-field theory. A fruitful concept from statistical physics is viewing learning through the prism of a *phase diagram*. [Spigler et al. \(2019\)](#) showed that one can delineate an under-parameterised phase (where the model cannot perfectly fit the training data) and an overparameterised phase (where many perfect fits exist) separated by a sharp “jamming transition”. At the jamming point, which corresponds to the interpolation threshold, the training error goes to zero and the loss landscape changes topology (a proliferation of global minima appears). This transition has characteristics akin to physical phase transitions or jamming in granular materials, and near this critical point the network’s performance is highly unstable (small changes in data or initialization can lead to large changes in the fitted function). In the overparameterised phase, however, all minima achieve zero training error and one finds empirically that these minima often yield comparable and low test errors (once the model is sufficiently overparameterised) ([Spigler et al. 2019](#); [Geiger et al. 2020](#)). The statistical mechanics framework explains this by noting that beyond the critical point, the volume of solution space is so large that SGD can find solutions that not only fit the data but also satisfy additional “random” constraints which effectively act like regularisation. In other words, the implicit bias of the algorithm (which we discuss more in the next section) combined with the easy accessibility of many minima in the high-dimensional weight landscape

leads to a selection of solutions with good generalisation almost by default. This aligns with the notion of *self-averaging*: in extremely overparameterised systems, the learned function tends to be an average over many microscopic weight configurations, which smooths out irregular behaviour.

It is important to emphasise the distinction in philosophy between SLT and statistical physics approaches. SLT strives for rigorous guarantees that hold for *any* data distribution (worst-case or distribution-free bounds), which often leads to loose but universally valid results. In contrast, statistical physics assumes a specific data model or random scenario and derives results that are exact in asymptotic settings (average-case analysis). The latter can capture phenomena like benign overfitting with great precision and offer clear intuitive explanations (e.g. “jamming transition” at the interpolation threshold), but their predictions may not hold universally if the assumptions are violated. Nevertheless, the insights obtained from statistical physics have informed the development of refined SLT hypotheses. For instance, the idea that overparameterisation might not harm (and can help) because of an implicit norm minimisation is supported both by replica calculations (Advani et al. 2020) and by rigorous results in linear models (Bartlett et al. 2020). This convergence of evidence from different methodologies gives confidence in the emerging picture of generalisation: overparameterisation opens up a regime where interpolation is possible, and provided the learning algorithm finds a “simple” interpolating solution (e.g. minimum-norm), generalisation can be excellent. In summary, statistical physics approaches have been invaluable for charting the qualitative phase diagram of learning and elucidating mechanisms (spectral properties, phase transitions, etc.) behind the surprising generalisation performance of large neural networks.

6.3. Dynamics of Gradient-Based Training

The generalisation ability of neural networks is profoundly influenced by the dynamics of the training algorithm, particularly *gradient-based* methods. Unlike classical learning theory, which often treats the hypothesis space and final minimiser abstractly, here we consider how the path taken by the training process acts as an implicit regulariser that steers the model towards simpler solutions. Empirically, it has been observed that standard training procedures (stochastic gradient descent with relatively small batches, moderate to large learning rates, and prolonged training past zero training error) tend to produce models that generalise well, whereas other training regimes (like very large batch sizes or certain adaptive optimisers) can lead to worse generalisation even if training error is equally low (Keskar et al. 2017). This indicates that *how* we minimise the loss matters as much as *how well* we minimise it. Several properties of SGD have been proposed as explanations for favourable generalisation behaviors.

A prime example of training dynamics influencing generalisation is the phenomenon of *implicit bias* towards certain solutions. In an influential line of work, Soudry et al. (2018) and Lyu & Li (2019) showed that for deep homogeneous networks (including linear models and deep ReLU networks) on separable data, gradient descent converges in direction to the max-margin classifier. That is, even though many weight vectors can achieve zero training error, GD picks one that maximises the classification margin on the training set, which is a known proxy for good generalisation by Vapnik’s

6. Overview of Generalisation Properties

theory. This occurs without any explicit regularisation in the loss — it is purely a consequence of the dynamics (e.g. GD with infinitesimal step size asymptotically follows the path of steepest descent which, for exponential losses, biases towards large-margin solutions). Similarly, for linear regression, gradient descent is known to converge to the minimum ℓ_2 -norm solution when the solution is underdetermined. These results explain why overparameterised networks often generalise: among the infinite interpolating solutions, the training algorithm tends to find one that is in some sense the “simplest” (minimum norm or maximum margin), which is exactly the solution that classical theory would favor if it had been enforced explicitly (Neyshabur et al. 2015). Implicit bias thus bridges a gap between the function space viewpoint and the parameter space: overparameterisation gives many degrees of freedom, but gradient-based training navigates this freedom by incrementally fitting the data in a way that minimises unnecessary complexity.

Another critical aspect is the role of *stochasticity* in training. For instance, the tendency of SGD to escape saddle points thanks to its random walk property has been established (Criscitiello & Boumal 2019) and could explain how solutions that generalise well are selected over all others. Furthermore, SGD, which samples random mini-batches of data to compute noisy gradient estimates, has a regularising effect compared to full-batch (deterministic) gradient descent. Intuitively, the noise in SGD can kick the parameters out of sharp narrow minima of the training loss and allow exploration of broader valleys. Keskar et al. (2017) provided evidence for this by showing that very large batch training (which has low noise) often converges to solutions that are “sharp” (the loss increases quickly in some direction around the minimum) and these solutions tend to generalise worse than the flatter minima found by small-batch SGD. Jastrzebski et al. (2017) and others further quantified this by relating the ratio of learning rate to batch size to an effective temperature: higher “temperature” (e.g. larger learning rate or smaller batch) leads SGD to prefer wider minima. In fact, Smith & Le (2018) argued from a Bayesian perspective that the noise in SGD encourages the weights to settle in regions of high posterior probability (high marginal likelihood), which correspond to flatter minima with simpler functions. This aligns with the PAC-Bayesian picture mentioned earlier. Empirically, one finds that techniques like *batch normalization* and cyclical learning rate schedules, which inject additional noise or instability, can further improve generalisation, presumably by helping SGD escape sharp basins and continually seek flatter regions.

The dynamic view also sheds light on puzzling behaviour like the continued improvement of test performance *after* the training error has reached zero. In classical theory, once a model interpolates the data, any further training would just be fitting noise and one might expect overfitting. Yet deep networks often undergo a period of “*implicit self-regularisation*” after interpolation: the training loss keeps decreasing (or stays at zero) and certain norm or margin metrics improve, which correlates with slowly improving test accuracy (Nakkiran et al. 2021). This is sometimes referred to as the model entering a “*grokking phase*” (Nam et al. 2025), during which a structure known as *neural collapse* can emerge in classification tasks (Papayan et al. 2020). Neural collapse is the tendency for the last-layer features of training examples of the same class to collapse to their class mean (and those means to become maximally separated). Although this is a phenomenon of the model’s internal representations rather than a direct measure of complexity, it illustrates that gradient descent continues to organise and refine the

6. Overview of Generalisation Properties

solution in ways that improve generalisation (by making the classes more separable in feature space) even without any change in training error. Traditional theory does not account for this because it usually considers any interpolating solution as equally good (all have zero training error), but the dynamics clearly select among interpolating solutions based on a host of implicit criteria (margins, symmetry, etc.). It is worth noting however that these rich dynamics do not always correlate with generalisation performance, as highlighted in (Nam et al. 2025).

Researchers have also noticed the prevalence of an “*edge of stability*” regime during training: as training progresses (especially with large learning rates), the model often operates at a point where the Hessian’s largest eigenvalue hovers around the value $2/\eta$ (where η is the learning rate), meaning the training is dynamically balanced at the border of stability (Cohen et al. 2021). In this regime, the loss may not decrease monotonically, but the model nonetheless keeps finding directions in weight space that lower the test loss. Lewkowycz et al. (2020) studied this phenomenon (sometimes called the “catapult mechanism”) and showed that starting with a large learning rate can rapidly catapult the model into a flat minimum that generalises well, after which reducing the learning rate (or implicitly, as the curvature evolves) fine-tunes the solution. This two-phase behaviour – an initial chaotic exploration followed by a stable convergence – seems to boost generalisation, echoing again the theme that traversing flat regions is beneficial. Such findings are inspiring new theoretical questions: How can a large learning rate find a region of low test error even when it initially overshoots minima? How do the non-linear dynamics of training avoid getting “stuck” in poor minima? While a full theory is still lacking, small-scale models and simplified analyses (e.g. for quadratic models) have replicated edge-of-stability behaviour and suggested that oscillations induced by too-large learning rates can serve as a form of implicit regularisation, preventing premature convergence to sharp minima (Cohen et al. 2021).

In summary, the dynamics of gradient-based training provide a narrative for generalisation that is complementary to capacity-based arguments: rather than constraining the function class *a priori*, the learning algorithm itself navigates towards functions of lower complexity. overparameterisation gives the algorithm many degrees of freedom, and indeed naive optimisation could find extremely complex (bad) solutions, but standard practices (SGD with noise, sufficiently large but not overly small learning rates, etc.) seem to inherently favor solutions with properties like large margins, low norms, flatness, and symmetric clustering of features. This implicit regularisation is now a major focus in understanding why deep learning works. It also has practical implications: if we modify the training dynamics (for instance, using adaptive optimisers that alter the effective step sizes per parameter), we might inadvertently change the implicit bias and thus affect generalisation in unpredictable ways. Empirically, this is seen in the fact that certain adaptive optimisers (like Adam) can sometimes generalise worse than SGD on vision tasks, presumably because they converge to different minima. The ongoing challenge is to precisely characterise the connection between training protocols and the functions they tend to produce. Current research in this area blends empirical findings with theoretical models, and while a complete theory is not yet in hand, a consensus is emerging that “SGD is not just a minimiser, it’s a regulariser.” The dynamical systems view of training, including phenomena like implicit bias and edge-of-stability, is thus an essential piece of the puzzle in explaining neural network generalisation.

6.4. Bayesian Perspectives

The favourable generalisation properties of neural networks in the overparametrised regime may also be explained using Bayesian methods and other kernel machines in a way that is unrelated to SGD training; indeed, the good performance of NNGP regression (Lee et al. 2018; Lee et al. 2020) and the successes of Bayesian neural networks (Gal 2016) provide compelling evidence in this direction. A related argument is that the parameter-function map is exponentially biased towards Kolmogorov simple functions (Dingle et al. 2018; Valle-Pérez et al. 2019; Mingard et al. 2019); since the data on which neural networks are trained has structure, the inductive bias implemented by the neural network prior leads to good generalisation. Furthermore, empirical evidence shows that SGD converges to functions with a probability that is remarkably close to the Bayesian posterior probability that a neural network expresses upon random sampling of parameters (Mingard et al. 2020), a phenomenon attributed to large differences in the sizes of the basins of attraction (Schaper & Louis 2014), suggesting that SGD is not so special after all. These ideas are still being debated (Ghorbani et al. 2020; Wilson & Izmailov 2020; Belkin 2021).

Bayesian perspective on neural networks thus offers yet another angle on generalisation, one that naturally embodies Ockham’s razor and uncertainty estimation. In Bayesian inference, instead of finding a single best parameter set, we maintain a *distribution* over parameters (the posterior) proportional to how well they explain the data times how plausible they were *a priori*. Predictions are made by averaging (integrating) over this posterior. The Bayesian framework automatically penalises complicated models due to the prior: unless the data strongly warrants a complex explanation, the posterior will assign higher probability to simpler models. This means that a Bayesian neural network (BNN) does not typically overfit even in extreme overparameterisation, as long as the prior is chosen sensibly. In fact, from a Bayesian viewpoint, the notion of “generalisation” is connected to *uncertainty*: a well-trained Bayesian model knows what it does not know, i.e. it makes cautious predictions (with large uncertainty) on inputs that are far from the training distribution, rather than confidently outputting arbitrary labels as a deterministic overfit model might.

While exact Bayesian inference for deep networks is computationally intractable, conceptually it is useful to consider the limiting case of an infinitely wide network with a Gaussian prior on weights. As noted earlier, such a network’s prior over functions is a Gaussian Process (GP) (Neal 1996). Performing Bayesian prediction in this limit is equivalent to doing kernel ridge regression with the NNGP kernel. This yields a clear benchmark: an infinite Bayesian net will generalise at least as well as any kernel method using that kernel, and in practice often better than a finite net that has not been trained in a Bayesian manner (since the infinite BNN effectively enforces a form of complexity control via the prior). However, intriguingly, in some cases *finite* neural networks can generalise better than their infinite-width (GP) counterparts. This is because finite networks can *learn features* from data, whereas an infinite network’s kernel is fixed at initialization. Recent theoretical work has explored this phenomenon. Zavatone-Veth et al. (2022), for example, compared deep Bayesian linear networks (where some layers are learned weights and others are random) to pure random feature models, and showed how learning the features (even in a Bayesian setting) can significantly improve generalisation especially when the amount of data is limited. The finite network

6. Overview of Generalisation Properties

can align its weights with the data distribution (a form of representation learning), effectively creating a data-dependent kernel that is better tuned to the task than the initial random kernel. This represents a *departure from the GP regime* and is an area where Bayesian analysis of neural networks intersects with statistical physics (using techniques like the replica method to handle the averages over weight distributions). For instance, [Li & Sompolinsky \(2021\)](#) developed a renormalisation group approach for deep linear networks to quantify how finite width and depth impact the posterior over functions, finding that finite-width networks have “broader” function priors that can fit data more easily (hence potentially better generalisation when appropriately Bayesian-trained).

In practice, exact Bayesian neural network training is approximated by methods like variational inference or Monte Carlo methods (e.g. Hamiltonian Monte Carlo or stochastic gradient Langevin dynamics). One influential approximation is *Monte Carlo dropout* ([Gal 2016](#)), which interprets dropout regularisation as a variational approximation to a BNN posterior. This approach gives improved uncertainty estimates and often slightly better generalisation than standard training, confirming that a Bayesian flavour (even approximate) helps avoid overfitting. Another approach is *deep ensembles*, which, while not strictly Bayesian, empirically behave like a Bayesian model averaging to some extent. Deep ensembles (training multiple models from different random initialisations and averaging their predictions) often yield not only lower errors but also well-calibrated probabilities, hinting that the ensemble is finding a diverse set of solutions whose average is a simpler function than any single solution. This again mirrors a Bayesian intuition: averaging over many plausible predictors tends to cancel out the idiosyncrasies of each, yielding a smoother, more broadly applicable predictor. Indeed, [Wilson & Izmailov \(2020\)](#) argue that techniques like deep ensembles or SGD with noise implicitly perform a form of Bayesian model averaging, which is one reason for their strong performance.

One concrete way the Bayesian view helps explain generalisation is via the concept of *marginal likelihood* or *Bayesian evidence*. The evidence is the probability of the data under the model, obtained by integrating over parameters. It naturally encodes a trade-off between model fit and model complexity (Occam’s factor) ([MacKay 1992](#)). Models (or parameter settings) that achieve high marginal likelihood must not only fit the data well but also come from a not-too-small volume of the parameter space (so as not to be overly fine-tuned). Empirically, it has been shown that within a given neural network architecture, the marginal likelihood can sometimes predict which settings (e.g. level of noise regularisation, or which epoch to stop training) yield better generalisation ([Valle-Pérez et al. 2019](#)). It is under this prism that [Valle-Pérez et al. \(2019\)](#) found that simpler functions (in terms of description length) are exponentially more likely under random weights, and this bias correlates with the functions that neural networks learn in practice. In other words, even absent explicit regularisation, the architecture itself plus random initialisation induces a prior over functions that favours simplicity, and SGD tends to follow the high-probability modes of this implicit prior. This perspective suggests that part of the generalisation mystery is resolved by recognizing that neural network architectures are not neutral function approximators—they are biased towards a certain class of functions (often smooth, hierarchical feature extractors, etc.), and this acts like a prior that complements the Bayesian story.

Finally, PAC-Bayesian theory provides a formal bridge between Bayesian integrals

6. Overview of Generalisation Properties

and generalisation bounds. Many PAC-Bayes bounds can be viewed as controlling the generalisation error of a Gibbs classifier (stochastic classifier drawn from a learned posterior) in terms of the KL divergence to a prior. If one chooses the prior as a simple distribution (say centered at the origin or at an earlier epoch’s weights) and the posterior as a distribution concentrated around the final trained weights (plus injected Gaussian noise), then a tight PAC-Bayes bound essentially says: “if the flatness/entropy of the minimum is high (i.e. the model could wander in a neighborhood without losing much accuracy), then generalisation will be good” (Dziugaite & Roy 2017; Jiang et al. 2019). This resonates strongly with Bayesian intuition that a flat minimum corresponds to a large evidence (many parameter settings give almost as good explanations, hence the explanation is robust and not finely tuned). In fact, recent work like Foret et al. (2021) (which introduced Sharpness-Aware Minimization, SAM) explicitly try to optimise a notion of flatness and can be interpreted through PAC-Bayes lenses as well. SAM modifies the training objective to find parameters that lie in a wide flat basin of the loss, leading to improved generalisation empirically; theoretically, this connects to tighter PAC-Bayes bounds as the KL term becomes smaller for a flat minimum (one can choose a narrower posterior around it).

In summary, the Bayesian perspectives emphasize *integrating* over model uncertainties and the induced preference for simpler models. Where classical SLT might worry that a network has many parameters, a Bayesian would point out that most of those parameter settings are ruled out by either the prior or the posterior normalization, and the model average will favor a simpler explanation of the data. overparameterisation is not an issue per se; it only means there is a vast space of functions, but the Bayesian procedure (and even SGD to some extent) will concentrate on a small, structured subset of those functions. By casting generalisation in terms of probability and inference, we gain access to tools like marginal likelihood and PAC-Bayes bounds that naturally encapsulate the trade-off between complexity and fit. The challenge in deep learning is that exact Bayesian computation is hard, but approximations and theoretical insights have shown that even a rough Bayesian thinking goes a long way: from explaining why dropout works (Gal 2016), to why flat minima are good (Jiang et al. 2019), to how feature learning can beat the GP limit (Aitchison 2020). Ultimately, a fully unified theory of generalisation in deep learning may require synthesizing all these perspectives—classical, physicist, dynamical, and Bayesian—since they each capture part of the truth. For now, this literature review shows that we have a much richer understanding in 2025 than we did a decade ago: deep networks generalise well not by magic or by defying the laws of statistics, but because of implicit regularities arising from high-dimensional geometry, algorithmic biases, and principled Bayesian averaging that were not apparent under the old paradigms.

7. Mercer Spectrum of Empirical Kernels

In this chapter, we propose the *Mercer spectrum of the empirical kernel* as a key mathematical object to study the generalisation in neural networks. The key insight is to notice that any finite neural network, for which the weights are considered fixed, for example after random initialisation or even after training, readily defines a kernel. This kernel therefore allows us to study properties of finite neural networks, and their asymptotic behaviours, using the machinery of kernel theory (Wand & Jones 1995; Györfi et al. 2002). This viewpoint is key to the contributions in chapters 8, 9 and 10. In this chapter, we will first give a precise definition of what we refer to as the empirical kernel, since it appears in the literature under various forms, and then, we develop tools to study its Mercer spectrum.

Finite neural networks define kernels. In this paragraph, we give our formal definition of the empirical kernel of a finite neural network. Consider the empirical feature map $\phi_{\mathcal{N}} : \mathbb{R}^{d_0} \mapsto \mathbb{R}^d$ of a finite (continuous) neural network $\mathcal{N}$, the function $K_{\mathcal{N}} : (\mathbf{x}, \mathbf{x}') \mapsto \frac{1}{d} [\phi_{\mathcal{N}}(\mathbf{x})]^T \phi_{\mathcal{N}}(\mathbf{x}')$ defined over $\mathbb{R}^{d_0} \times \mathbb{R}^{d_0} \mapsto \mathbb{R}$ is (continuous) positive semi-definite, so by definition it is a kernel, which therefore depends on parameters Θ of $\mathcal{N}$, and on the dimensions d, d_0 . In the following, we will sometimes note K_{Θ}^{d, d_0} for $K_{\mathcal{N}}$ to emphasize the dependency. We refer to this kernel as the *empirical kernel* of neural network $\mathcal{N}$. We note that the empirical kernel considered here differs substantially from the random features perspective developed in e.g. Rahimi & Recht (2007); Rahimi & Recht (2008); Rahimi & Recht (2009). Random features approximates a target kernel by mapping inputs through a finite-dimensional randomised feature map and taking inner products. In the random features setting, the feature map is explicitly constructed from random bases drawn independently of the data, and the associated kernel converges, as the number of features increases, to a predefined kernel. In contrast, the empirical feature map $\phi_{\mathcal{N}}$ of a neural network is structured and parameter-dependent, arising from the composition of learned (or randomly initialised) neural transformations. As a result, the empirical kernel $K_{\mathcal{N}}$ encompasses data-dependent and learned kernels, which do not approximate a fixed kernel a priori but instead reflects the representation induced by a specific finite network instance. This distinction is central to our analysis, as it allows the Mercer spectrum of $K_{\mathcal{N}}$ to capture both architectural biases and the effect of training. Importantly, note that the kernel studied in this chapter should also not be confused with the NNGP kernel (although related to it via the infinite-width limit): it is a parameterized kernel which is different for each finite neural network instance. In this particular case, the Hilbert space of Riesz's representation theorem is known $\mathcal{H} = \mathbb{R}^d$. Therefore, Mercer's theorem is applicable to $K_{\mathcal{N}}$ and any data distribution p which is a Borel measure with compact support on $\mathbb{R}^{d_0}$. The resulting Mercer eigenvalues and eigenfunctions thus depend on $K_{\mathcal{N}}$ and p . This perspective will be used extensively in chapters 8, 9 and 10.

Technical contributions. In the following sections, we first characterise a counterpart of the NNGP kernel with finite width. Specifically, we identify the random kernel underpinning the empirical covariance matrix by assuming that the width of the last layer is finite. Then, we study the limiting spectral distribution of the corresponding empirical covariance matrix of features. We derive a non-trivial relationship between the spectral distributions of the empirical covariance random matrix and the actual NNGP kernel random matrix.

Finally, starting from the empirical kernel, we showcase the emergence of a *modified NNGP kernel* from BNNs in the linear-width and sublinear-width limits, and using the previous results, we also study the Mercer’s spectrum of the modified NNGP kernel.

7.1. A Width-dependent Random Kernel

In this subsection, we assume that the training set is sampled i.i.d. from $\mathbf{x} \sim \mathbb{P}_{d_0}$, and that the family $(\mathbb{P}_{d_0})_{d_0 \in \mathbb{N}}$ of probability distributions over $\mathbb{R}^{d_0}$ verifies that $\mathbb{E}_{\mathbf{x} \sim \mathbb{P}_{d_0}}(\mathbf{x}^T \mathbf{x})$ is bounded as $d_0 \rightarrow \infty$.

Consider $g \sim \mathcal{GP}(0, K_{\text{NNGP}})$ obtained with the infinite width limit of (2.1). Finding a dependence with the width d is not straightforward in the case of NNGPs because, at this point, the network width has already been taken to infinity. Our idea is therefore to study the behavior of a counterpart of the Gaussian process g *before* the width is taken to infinity.

Following (Lee et al. 2018), we consider the output $h^{L,d}(\mathbf{x}_i) = \sum_{k=1}^d \mathbf{W}_k^L \sigma(\mathbf{h}_k^{L-1}(\mathbf{x}_i))$ of a random network (2.1) with $L \geq 2$, $d_1, \dots, d_{L-2} = \infty$, $d_{L-1} = d$ and $d_L = 1$, i.e. where all widths have been taken to infinity with the exception of the last one.

Proposition 7.1. *The covariance matrix of the evaluations of $h^{L,d}$, conditional on the pre-activations, satisfies, for all pairs of training data points $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^{d_0}$ (rows of $\mathbf{X}$), that:*

$$\begin{aligned} \left(\mathbf{K}_{\mathbf{X}, \sigma, \mathbf{h}^{L-1}}^d \right)_{i,j} &= \mathbb{E}_{\mathbf{W}^L} \left(h^{L,d}(\mathbf{x}_i) h^{L,d}(\mathbf{x}_j) \mid \{\mathbf{h}_k^{L-1}\}_{k=1}^d \right) \\ &= \frac{1}{d} \sum_{k=1}^d \sigma(\mathbf{h}_k^{L-1}(\mathbf{x}_i)) \sigma(\mathbf{h}_k^{L-1}(\mathbf{x}_j)) \end{aligned}$$

where the expectation is thus taken over the last-layer weights $\mathbf{W}^L$, and it is an unbiased estimator of $K_{\text{NNGP}}(\mathbf{x}_i, \mathbf{x}_j)$ with variance $\text{Var} \left(\left(\mathbf{K}_{\mathbf{X}, \sigma, \mathbf{h}^{L-1}}^d \right)_{i,j} \right) = \mathcal{O}_{d \rightarrow \infty}(\frac{1}{d})$.

Proof. The proof is immediate using that the fact the $\mathbf{h}_i^{L-1}$ are independent, $\mathbf{h}_i^{L-1} \sim \mathcal{GP}(0, K^{\sigma, L-1})$, and that $\frac{1}{d} \sum_{i=1}^d A_i$ is an unbiased estimator of $\mathbb{E}(A)$ with variance $\frac{\text{Var}(A)}{d}$ for any collection of random variables A_i, A such that $A_i \sim A$ sampled i.i.d. . To calculate the variance, in order to show that it is $\mathcal{O}_{d \rightarrow \infty}(1)$ (standard big O notation), one can use the same trick as in (Poole et al. 2016), where they computed an iterative map of the covariances in the limit of infinite width:

$$K^{\sigma, l}(\mathbf{x}, \mathbf{x}') = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sigma(u_1^{\sigma, l-1}(\mathbf{x}, \mathbf{x}')) \sigma(u_2^{\sigma, l-1}(\mathbf{x}, \mathbf{x}')) \mathcal{D}z_1 \mathcal{D}z_2 \quad (7.1)$$

7. Mercer Spectrum of Empirical Kernels

and

$$K^{\sigma,0}(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle \quad (7.2)$$

where $\mathcal{D}z$ is the standard Gaussian measure, and

$$u_1^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') = \sqrt{K^{\sigma,l-1}(\mathbf{x}, \mathbf{x})} z_1 \quad (7.3)$$

$$u_2^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') = \sqrt{K^{\sigma,l-1}(\mathbf{x}', \mathbf{x}')} \left[\rho^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') z_1 + \sqrt{1 - \rho^{\sigma,l-1}(\mathbf{x}, \mathbf{x}')^2} z_2 \right] \quad (7.4)$$

with $\rho^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') = \frac{K^{\sigma,l-1}(\mathbf{x}, \mathbf{x}')}{\sqrt{K^{\sigma,l-1}(\mathbf{x}, \mathbf{x})K^{\sigma,l-1}(\mathbf{x}', \mathbf{x}')}}.$ The trick is that (z_1, z_2) is the joint density of independent standard Gaussians, and $\begin{pmatrix} u_1^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') \\ u_2^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') \end{pmatrix}$ is thus the density of a bivariate Gaussian with covariance matrix $\begin{pmatrix} K^{\sigma,l-1}(\mathbf{x}, \mathbf{x}) & K^{\sigma,l-1}(\mathbf{x}, \mathbf{x}') \\ K^{\sigma,l-1}(\mathbf{x}', \mathbf{x}) & K^{\sigma,l-1}(\mathbf{x}', \mathbf{x}') \end{pmatrix}.$ We find that:

$$\text{Var} \left(\left(\mathbf{K}_{\mathbf{x}, \sigma, \mathbf{h}^{L-1}}^d \right)_{i,j} \right) = \frac{1}{d} \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \sigma \left(u_1^{\sigma,L-1}(\mathbf{x}_i, \mathbf{x}_j) \right)^2 \sigma \left(u_2^{\sigma,L-1}(\mathbf{x}_i, \mathbf{x}_j) \right)^2 \mathcal{D}z_1 \mathcal{D}z_2 - K^{\sigma,L}(\mathbf{x}_i, \mathbf{x}_j)^2 \right] \quad (7.5)$$

□

The random matrix $\mathbf{K}_{\mathbf{x}, \sigma, \mathbf{h}^{L-1}}^d$ is the empirical covariance matrix of the features created by the NNGP (2.1) before the last width is taken to infinity.

Conditionally on $\mathbf{X}$, the values $K_{\text{NNGP}}(\mathbf{x}_i, \mathbf{x}_j)$ are constant and the $\left(\mathbf{K}_{\mathbf{x}, \sigma, \mathbf{h}^{L-1}}^d \right)_{i,j}$ are random variables whose randomness stems from $\mathbf{h}^{L-1}$. In turn, $\mathbf{K}_{\mathbf{x}, \sigma, \mathbf{h}^{L-1}}^d$ satisfies the kernel property (Rasmussen & Williams 2006):

$$\sum_{i=1}^n \sum_{j=1}^n a_i a_j \left(\mathbf{K}_{\mathbf{x}, \sigma, \mathbf{h}^{L-1}}^d \right)_{i,j} = \frac{1}{d} \mathbf{a}^T \Phi^T \Phi \mathbf{a} \geq 0 \quad (7.6)$$

for all $\mathbf{a} = (a_1, \dots, a_n)^T \in \mathbb{R}^n$, where $\Phi \in \mathbb{R}^{d \times n}$ with $\Phi_{jk} = \sigma \left(\mathbf{h}_j^{L-1}(\mathbf{x}_k) \right)$, which holds for any realisation of the random matrix Φ . Note, however, that $\mathbf{K}_{\mathbf{x}, \sigma, \mathbf{h}^{L-1}}^d$ does not define a kernel as it is not a well-defined function of $(\mathbf{x}, \mathbf{x}')$ but merely a countable family of random variables that can be indexed on $(\mathbf{x}, \mathbf{x}')$. This is problematic in our setting since the covariance function in a Gaussian process must be a kernel with respect to the full, continuous, space.

We next propose a way of converting the aforementioned family of random variables into a random kernel, i.e. a kernel-valued random variable.

Theorem 7.2. For $d \in (1, \infty)$, and $L \geq 2$ there exists a probability space $(\Omega_d, \mathcal{A}_d, \mathbb{P}_d)$ and a random variable $K^{\sigma,L,d} : \Omega_d \longrightarrow \mathbb{R}^{(\mathbb{R}^d)^2}$ with image in the functional space $\mathbb{R}^{(\mathbb{R}^d)^2}$ such that :

1. $K^{\sigma,L,d}(\omega)$ is a kernel for all $\omega \in \Omega_d$,

7. Mercer Spectrum of Empirical Kernels

2. for all sets of points $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T \in \mathbb{R}^{n \times d}$, the random matrix $\mathbf{K}_{\mathbf{X}, \sigma, \mathbf{h}^{L-1}}^d$, and the random matrix $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^{\sigma, L, d}$, defined as,

$$\Omega_d \longrightarrow \mathbb{R}^{n \times n} \quad \omega \mapsto \left(K^{\sigma, L, d}(\omega)(\mathbf{x}_i, \mathbf{x}_j) \right)_{ij \in \{1, \dots, n\}}$$

follow the same distribution: for all $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$, $\mathbb{E}_{K^{\sigma, L, d}}(K^{\sigma, L, d}(\mathbf{x}_i, \mathbf{x}_j)) = K^{\sigma, L}(\mathbf{x}_i, \mathbf{x}_j)$, where the expectation is taken over the random kernel $K^{\sigma, L, d}$.

Proof. The trick is to utilise a stochastic process which provides a probability space where one can sample from infinitely-, uncountably-, many Gaussian distributions. Here we chose the Brownian motion.

Consider the probability space $(\Omega_d, \mathcal{A}_d, \mathbb{P}_d)$ associated with an d -dimensional standard Brownian motion, $(B_t^k)_{t \in \mathbb{R}}$ for $k \in \{1, \dots, n\}$. Define the following random variable in $\mathbb{R}^{(\mathbb{R}^d)^2}$:

$$\begin{aligned} K^{\sigma, L, d} : \Omega_d &\longrightarrow \mathbb{R}^{(\mathbb{R}^d)^2} \\ \omega &\mapsto K^{\sigma, L, d}(\omega) \end{aligned} \tag{7.7}$$

with

$$\begin{aligned} K^{\sigma, L, d}(\omega) : (\mathbb{R}^d)^2 &\rightarrow \mathbb{R} \\ (\mathbf{x}, \mathbf{x}') &\mapsto \frac{1}{d} \sum_{k=1}^d \sigma \left(u_1^{\sigma, L-1, B^k}(\mathbf{x}, \mathbf{x}')(\omega) \right) \sigma \left(u_2^{\sigma, L-1, B^k}(\mathbf{x}, \mathbf{x}')(\omega) \right) \end{aligned} \tag{7.8}$$

where

$$u_1^{\sigma, L-1, B^k}(\mathbf{x}, \mathbf{x}') = B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})}^k \tag{7.9}$$

and

$$\begin{aligned} u_2^{\sigma, L-1, B^k}(\mathbf{x}, \mathbf{x}') &= \frac{\sqrt{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}')}}{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})} B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})}^k \\ &\quad + \sqrt{1 - \frac{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}')}{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})K^{\sigma, L-1}(\mathbf{x}', \mathbf{x}')}} \left(B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}) + K^{\sigma, L-1}(\mathbf{x}', \mathbf{x}')}^k - B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})}^k \right) \end{aligned} \tag{7.10}$$

Using the same technique introduced by (Poole et al. 2016) for the iterative map of the covariances in the NNGP kernel (see equation (7.1)), we have that

$$\left(u_1^{\sigma, L-1, B^k}(\mathbf{x}, \mathbf{x}'), u_2^{\sigma, L-1, B^k}(\mathbf{x}, \mathbf{x}') \right),$$

follows a bivariate Gaussian with covariance matrix $\begin{pmatrix} K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}) & K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}') \\ K^{\sigma, L-1}(\mathbf{x}', \mathbf{x}) & K^{\sigma, L-1}(\mathbf{x}', \mathbf{x}') \end{pmatrix}$, where we have used that $B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})}^k \sim N(0, K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}))$, and $B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x}) + K^{\sigma, L-1}(\mathbf{x}', \mathbf{x}')}^k - B_{K^{\sigma, L-1}(\mathbf{x}, \mathbf{x})}^k \sim N(0, K^{\sigma, L-1}(\mathbf{x}', \mathbf{x}'))$ are independent, by properties of the Brownian motion. We have that $(\mathbf{h}_k^{L-1}(\mathbf{x}), \mathbf{h}_k^{L-1}(\mathbf{x}'))$ follows the same bivariate Gaussian distribution since $\mathbf{h}_k^{L-1} \sim \mathcal{GP}(0, K^{\sigma, L-1})$, thus the real-valued random variable $\omega \mapsto$

7. Mercer Spectrum of Empirical Kernels

$K^{\sigma,L,d}(\omega)(\mathbf{x}_i, \mathbf{x}_j)$ has the same distribution as $\left(\mathbf{K}_{\mathbf{X},\sigma,\mathbf{h}^{L-1}}^d\right)_{i,j}$, for all i, j which gives us the point 2.

Equation (7.6) thus proves that $K^{\sigma,L,d}(\omega)$, which is a well-defined function of $\mathbb{R}^{(\mathbb{R}^d)^2}$, is a kernel, which terminates the proof. $\square$

We have thus defined a random variable $K^{\sigma,L,d}$ over a functional space, whose realisations are kernel functions interpolating the random matrices of interest. When there is no ambiguity, we use K^d to denote $K^{\sigma,L,d}$. We can now study the random matrices $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$, whose randomness stems from the random kernel function K^d and the random matrix $\mathbf{X}$, using the more convenient definition of $\mathbf{K}_{\mathbf{X},\sigma,\mathbf{h}^{L-1}}^d$, whose randomness stems from the random variables $\mathbf{h}^{L-1}$ and the random matrix $\mathbf{X}$. Conditionally on K^d , the corresponding Gaussian process $g_{K^d} \sim \mathcal{GP}(0, K^d)$ is well-defined and Bayesian inference can be performed. We refer to K^d as the *width-dependent NNGP kernel*, or *width-dependent random kernel*, as opposed to the actual (width-independent) NNGP kernel.

7.2. Limiting Spectral Distributions

The following theorem relates the limiting spectral distribution of the *actual* width-independent NNGP kernel random matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}$ and the *width-dependent* NNGP kernel random matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$.

Theorem 7.3. *Consider an NNGP obtained with the infinite-width limit of (2.1) with $L \geq 2$, $d_L = 1$ and the non-linear activation σ , a measurable, Lipschitz function. Consider the associated NNGP kernel denoted K_{NNGP} , the associated random kernel function K^d and the random matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$ defined by Theorem 7.2 for kernel K^d . Then, the NNGP kernel random matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}$ admits, in the limit $n, d_0 \rightarrow \infty$ with fixed ratio $\frac{n}{d_0} = \alpha_0 \in (0, \infty)$, a limiting nonrandom spectral measure $\mu_{\alpha_0}^\sigma$. Furthermore, in the limit $n, d, d_0 \rightarrow \infty$ with fixed ratio $\frac{n}{d} = \alpha \in (0, \infty)$, $\frac{n}{d_0} = \alpha_0 \in (0, \infty)$, the empirical spectral distribution of $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$ converges in distribution to the nonrandom measure $\rho_{\text{MP}}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$.*

The proof of Theorem 7.3, relies on a result in random matrix theory (Theorem 1 in (Banna et al. 2015)). As a corollary, for deep linear networks, if the data covariance matrix $\mathbf{X}\mathbf{X}^T$ admits a limiting spectral distribution μ_{α_0} , then the limiting spectral distribution of $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$ is $\rho_{\text{MP}}^\alpha \boxtimes \mu_{\alpha_0}$. In particular, if the covariance matrix is isotropic, then $\mu_{\alpha_0} = \rho_{\text{MP}}^{\alpha_0}$ and the limiting spectral distribution is the Marchenko-Pastur map of a Marchenko-Pastur distribution $\rho_{\text{MP}}^\alpha \boxtimes \rho_{\text{MP}}^{\alpha_0}$.

Here, we have made an important distinction between the random matrices $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$ and $\mathbf{K}_{\mathbf{X},\mathbf{X}}$, which was not made in previous works (Fan & Wang 2020). Indeed, it is not the same thing to consider the NNGP kernel K_{NNGP} , which appears *after* the width of a neural network is taken to infinity, and its counterpart K^d *before* the width is taken to infinity (which should be called the *empirical* NNGP kernel).

Theorem 7.3 tells us how the spectral distribution of the empirical covariance matrix of the features created by the neural network (2.1) depends on the *actual* NNGP kernel.

7. Mercer Spectrum of Empirical Kernels

The important fact to notice for the interpretation of the double-descent curve in neural networks is that, in the extremely overparametrised regime $\alpha \rightarrow 0$, the spectral distribution becomes that of the NNGP kernel itself. Indeed, the fixed-point equation (2.2), which characterises the Marchenko-Pastur map of $\mu_{\alpha_0}^\sigma$, becomes:

$$S(z) = \int \frac{1}{x-z} d\mu_{\alpha_0}^\sigma(x) = S_{\mu_{\alpha_0}^\sigma}(z). \quad (7.11)$$

To put it differently from a spectral perspective, the neural network exhibits behavior akin to its corresponding NNGP in the highly overparametrised regime.

In the next chapter, we will explore how the generalisation error of the corresponding GP and kernel regressions hinges on this spectral distribution and mirrors the double-descent pattern.

Proof. For clarity, we first prove the result for a two-layer linear network with isotropic input data, then we extend to a deep linear network with our mild assumption on the data distribution, and finally we generalise the result to a deep non-linear network under our mild assumptions on the nonlinearity.

Two-layer linear network with isotropic input data. We consider a two-layer linear network $L = 2$, $\sigma = \text{Id}_{\mathbb{R}}$, and isotropic input data $\mathbb{P}_{d_0} = N(0, \frac{1}{d_0} \mathbf{I}_{d_0})$.

Let $n, d, d_0 \in (0, \infty)$ such that $\frac{n}{d_0} = \alpha_0 \in (0, \infty)$, $\frac{n}{d} = \alpha \in (0, \infty)$.

Conditionally on $\mathbf{X}$, $\mathbf{h}_1^1, \dots, \mathbf{h}_d^1$ are d independent realisations of the n -dimensional multivariate Gaussian $N(0, \mathbf{X}\mathbf{X}^T)$ as implied by equation (7.2). We will note $\mathbf{H}_\mathbf{X} = (\mathbf{h}_1^1, \dots, \mathbf{h}_d^1)^T \in \mathbb{R}^{d \times n}$, and we have $\mathbf{K}_{\mathbf{X}, \sigma, \mathbf{h}^1}^d = \frac{1}{d} \mathbf{H}_\mathbf{X}^T \mathbf{H}_\mathbf{X}$ as per Proposition 7.1.

The idea is to diagonalise the symmetric matrix $\mathbf{X}\mathbf{X}^T$ in order to deal with uncorrelated random variables:

$$\mathbf{X}\mathbf{X}^T = \mathbf{V}_\mathbf{X} \Lambda_\mathbf{X} \mathbf{V}_\mathbf{X}^T \quad (7.12)$$

where $\Lambda_\mathbf{X}$ is a diagonal matrix, $\mathbf{V}_\mathbf{X}$ is an orthonormal matrix.

Now, consider $\hat{\mathbf{h}}_1, \dots, \hat{\mathbf{h}}_d$, d independent realisations of $N(0, \Lambda_\mathbf{X})$, and define $\hat{\mathbf{H}}_\mathbf{X} = (\hat{\mathbf{h}}_1, \dots, \hat{\mathbf{h}}_d)^T \in \mathbb{R}^{d \times n}$. We have that $\mathbf{H}_\mathbf{X}$ and $\hat{\mathbf{H}}_\mathbf{X} \mathbf{V}_\mathbf{X}^T$ have the same distribution conditionally on $\mathbf{X}$, thanks to the linear properties of the Gaussian. This implies that $\mathbf{H}_\mathbf{X}^T \mathbf{H}_\mathbf{X}$ and $\mathbf{V}_\mathbf{X} \hat{\mathbf{H}}_\mathbf{X}^T \hat{\mathbf{H}}_\mathbf{X} \mathbf{V}_\mathbf{X}^T$ have the same distribution conditionally on $\mathbf{X}$, thus the same distribution when the conditioning is relaxed. In particular, they have the same spectral distribution. We have reduced the problem to studying the existence of a limiting spectral distribution for $\frac{1}{d} \mathbf{V}_\mathbf{X} \hat{\mathbf{H}}_\mathbf{X}^T \hat{\mathbf{H}}_\mathbf{X} \mathbf{V}_\mathbf{X}^T$.

Furthermore, $\frac{1}{d} \hat{\mathbf{H}}_\mathbf{X}^T \hat{\mathbf{H}}_\mathbf{X}$ and $\frac{1}{d} \mathbf{V}_\mathbf{X} \hat{\mathbf{H}}_\mathbf{X}^T \hat{\mathbf{H}}_\mathbf{X} \mathbf{V}_\mathbf{X}^T$ have the same spectral distribution, since they only differ by a basis change. We have thus further reduced the problem to studying the existence of a limiting spectral distribution for $\frac{1}{d} \hat{\mathbf{H}}_\mathbf{X}^T \hat{\mathbf{H}}_\mathbf{X}$.

As per the definition of $\hat{\mathbf{H}}_\mathbf{X}$, it can be re-written:

$$\hat{\mathbf{H}}_\mathbf{X} = \sqrt{d} \mathbf{M} \Lambda_\mathbf{X}^{1/2} \quad (7.13)$$

where $\mathbf{M} = (\mathbf{m}_1, \dots, \mathbf{m}_d)^T \in \mathbb{R}^{d \times n}$ with $\mathbf{m}_1, \dots, \mathbf{m}_d$ d independent realisations of $N(0, \frac{1}{d} \mathbf{I}_n)$.

7. Mercer Spectrum of Empirical Kernels

In the limit of infinite quantities, since the data distribution is isotropic, the limit of the spectral measure of $\mathbf{X}\mathbf{X}^T$ exists almost surely: it is the Marchenko-Pastur distribution $\rho_{MP}^{\alpha_0}$ (Marchenko & Pastur 1967).

Conditionally on $\mathbf{X}$, $\Lambda_{\mathbf{X}}$ is deterministic and positive semi-definite (diagonal positive), thus the Marchenko-Pastur map applies, and $\frac{1}{d}\hat{\mathbf{H}}_{\mathbf{X}}^T\hat{\mathbf{H}}_{\mathbf{X}} = \left(\Lambda_{\mathbf{X}}^{1/2}\right)^T \mathbf{M}^T \mathbf{M} \Lambda_{\mathbf{X}}^{1/2}$ admits a limiting spectral distribution: $\rho_{MP}^{\alpha} \boxtimes \mu_{\mathbf{X}}$, where $\mu_{\mathbf{X}}$ is the infinite limit of the spectral measure of a given realisation of $\mathbf{X}\mathbf{X}^T$ (because of the conditioning), which exists almost surely.

Relaxing the conditioning on $\mathbf{X}$ gives us the limiting spectral distribution $\rho_{MP}^{\alpha} \boxtimes \rho_{MP}^{\alpha_0}$, because $\mu_{\mathbf{X}}$ is almost surely $\rho_{MP}^{\alpha_0}$. In other words, the empirical spectral distribution converges almost surely to $\rho_{MP}^{\alpha} \boxtimes \rho_{MP}^{\alpha_0}$, which terminates the proof for a two-layer linear network with isotropic input data.

Deep linear network and input data with bounded dot product. We now consider a deep linear network $\sigma = \text{Id}_{\mathbb{R}}$ and a data distribution $\mathbf{x} \sim \mathbb{P}_{d_0}$ such that $\mathbb{E}_{\mathbf{x} \sim \mathbb{P}_{d_0}}(\mathbf{x}^T \mathbf{x})$ is bounded as $d_0 \rightarrow \infty$.

The extension of the proof is immediate following closely the previous proof, since we only used that the limiting spectral distribution of the empirical covariance matrix $\mathbf{X}\mathbf{X}^T$ is a Marchenko-Pastur distribution in the last stage, when relaxing the conditioning on $\mathbf{X}$. The assumption on the data distribution ensures that $\mathbf{X}\mathbf{X}^T$ admits a nonrandom limiting spectral measure μ_{α_0} in the limit $n, d_0 \rightarrow \infty$, with $\frac{n}{d_0} = \alpha_0$ (Livan et al. 2018).

Furthermore, it can be shown by immediate induction, that when $\sigma = \text{Id}_{\mathbb{R}}$, the iterative map of the covariances of the NNGP kernel (see equation (7.1)) yields $K^l(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$ for all $l \geq 2$, thus $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^{\sigma, l, d} = \mathbf{X}\mathbf{X}^T$ for all $l \geq 2$. Since $\mathbf{h}_1^{L-1}, \dots, \mathbf{h}_d^{L-1}$ are d independent realisations of $N(0, \mathbf{K}_{\mathbf{X}, \mathbf{X}}^{\sigma, L-1, d})$, we can thus reproduce the previous proof with $\mathbf{H}_{\mathbf{X}}^{L-1} = (\mathbf{h}_1^{L-1}, \dots, \mathbf{h}_d^{L-1})^T \in \mathbb{R}^{d \times n}$, and obtain the same limiting spectral distribution for $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^{\sigma, L, d}$.

Deep non-linear network with measurable Lipschitz nonlinearity. First, let us briefly discuss why $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^{\sigma, L}$ has indeed a limiting nonrandom spectral measure $\mu_{\sigma}^{\alpha_0}$. Using that σ is c -Lipschitz, we have that:

$$\begin{aligned} |\mathbb{E}_{\mathbf{x} \sim \mathbb{P}_d, f \sim \mathcal{GP}(0, K^0)}(\sigma(f(\mathbf{x}))^2)| &\leq c^2 \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_d, f \sim \mathcal{GP}(0, K^0)}(|f(\mathbf{x})|^2) \\ &= c^2 \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_d}(\mathbf{x}^T \mathbf{x}) \\ &< \infty \end{aligned} \tag{7.14}$$

By immediate induction on the depth L , since $\mathbb{E}_{\mathbf{x} \sim \mathbb{P}_d, f \sim \mathcal{GP}(0, K^0)}(\sigma(f(\mathbf{x}))^2) < \infty$, it follows that $\mathbb{E}_{\mathbf{x}, \mathbf{x}'}(K^{\sigma, L}(\mathbf{x}, \mathbf{x}')) < \infty$ (by applying successively Cauchy-Schwarz inequalities). This is sufficient to apply Mercer's decomposition theorem (Mercer 1909; Minh et al. 2006), which provides eigenvalues independent from the sampling effects of the data, and therefore a nonrandom spectral measure. One can conclude by invoking (Baker 1977; Rasmussen & Williams 2006) which proves that the eigenvalues of the empirical kernel matrix $K_{\mathbf{X}, \mathbf{X}}^{\sigma, L}$ converge to the Mercer's eigenvalues.

7. Mercer Spectrum of Empirical Kernels

With a non-linear activation, the trick of diagonalising the covariance matrix $\mathbf{X}\mathbf{X}^T$ does not directly apply, since we need to consider $\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$, where σ acts entrywise, and we cannot ensure that $\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$ has the same distribution as $\sigma(\hat{\mathbf{H}}_{\mathbf{X}}^{L-1})\mathbf{V}_{\mathbf{X}}^{L-1}$. Instead, we diagonalise the NNGP kernel matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L}$ in order to utilise a somewhat similar property.

With the previous notation, we have that $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L,d} = \frac{1}{d}\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})^T\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$. We diagonalise the symmetric matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L}$.

$$\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L} = \mathbf{U}_{\mathbf{X}}^{\sigma,L} \Lambda_{\mathbf{X}}^{\sigma,L} (\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T \quad (7.15)$$

where $\Lambda_{\mathbf{X}}^{\sigma,L}$ is a diagonal matrix, and $\mathbf{U}_{\mathbf{X}}^{\sigma,L} \in \mathbb{R}^{n \times n}$ is an orthonormal matrix. Consider, as previously, $\hat{\mathbf{h}}_1^{\sigma,L}, \dots, \hat{\mathbf{h}}_d^{\sigma,L}$, d independent realisations of $N(0, \Lambda_{\mathbf{X}}^{\sigma,L})$ and note $\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L} = (\hat{\mathbf{h}}_1^{\sigma,L}, \dots, \hat{\mathbf{h}}_d^{\sigma,L})^T \in \mathbb{R}^{d \times n}$. In this case, $\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L}(\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T$ and $\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$ do not have the same distribution, because the distributions are not Gaussian anymore, but they have the same covariance structure, conditionally on $\mathbf{X}$:

$$\begin{aligned} \mathbb{E}_h \left(\sigma((\mathbf{h}_i^{L-1})_k) \sigma((\mathbf{h}_i^{L-1})_l) | \mathbf{X} \right) &= \left(\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L} \right)_{k,l} \\ &= \sum_{p=1}^n \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{k,p} \left(\Lambda_{\mathbf{X}}^{\sigma,L} (\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T \right)_{p,l} \\ &= \sum_{p=1}^n \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{k,p} \sum_{q=1}^n \left(\Lambda_{\mathbf{X}}^{\sigma,L} \right)_{p,q} \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{q,l}^T \\ &= \sum_{p=1}^n \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{k,p} \left(\Lambda_{\mathbf{X}}^{\sigma,L} \right)_{p,p} \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{l,p} \\ &= \sum_{p=1}^n \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{k,p} \mathbb{E}_h \left((\hat{\mathbf{h}}_i^{\sigma,L})_p (\hat{\mathbf{h}}_i^{\sigma,L})_p | \mathbf{X} \right) \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{l,p} \\ &= \mathbb{E}_h \left(\sum_{p=1}^n \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{k,p} (\hat{\mathbf{h}}_i^{\sigma,L})_p (\hat{\mathbf{h}}_i^{\sigma,L})_p \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{l,p} | \mathbf{X} \right) \\ &= \mathbb{E}_h \left(\sum_{p=1}^n (\hat{\mathbf{h}}_i^{\sigma,L})_p \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{p,k}^T \sum_{q=1}^n (\hat{\mathbf{h}}_i^{\sigma,L})_q \left(\mathbf{U}_{\mathbf{X}}^{\sigma,L} \right)_{q,l}^T | \mathbf{X} \right) \\ &= \mathbb{E}_h \left(\left(\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L} (\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T \right)_{i,k} \left(\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L} (\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T \right)_{i,l} | \mathbf{X} \right) \end{aligned} \quad (7.16)$$

where we have used successively the definition of the NNGP kernel, its eigendecomposition, and that $\hat{\mathbf{h}}_i^{\sigma,L} \sim N(0, \Lambda_{\mathbf{X}}^{\sigma,L})$. By the law of total expectations, they also have the same covariance structure with respect to $(\mathbf{X}, h)$.

Note that this was also the case in the linear case (with the same calculation), and it was precisely the covariance structure that gave us that the random matrices have the same distribution, since a multivariate Gaussian is uniquely determined by its covariance matrix.

We can again get rid of $\mathbf{U}_{\mathbf{X}}^{\sigma}$ in the last stage of the proof, since it corresponds to a mere basis change which does not change the spectral distribution.

7. Mercer Spectrum of Empirical Kernels

Now, we will utilise a result in random matrix theory (Banna et al. 2015), which states that the empirical spectral distribution of a symmetric matrix whose entries are measurable functions of independent random variables converges in distribution to that of a Gaussian symmetric matrix with the same covariance structure. In our case, since the random matrices $\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L}(\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T$ and $\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$ have the same covariance structure, then the associated Gram matrices $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L,d} = \frac{1}{d}\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})^T\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$ and $\frac{1}{d}\mathbf{U}_{\mathbf{X}}^{\sigma,L}(\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L})^T\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L}(\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T$ have the same limiting spectral distribution (if it exists), by Theorems 1 and 5 in (Banna et al. 2015).

Let us first verify that the assumptions in Theorems 1 and 5 in (Banna et al. 2015) hold true in our case. Again, we reason conditionally on $\mathbf{X}$. By conditioning on $\mathbf{X}$, we will show that the kernel random matrices of interest have the same limiting spectral distribution for any infinite realisation of $\mathbf{X}$, i.e. when the randomness only comes from h .

To stick to the way they define random matrices, namely as matrices extracted from infinite random fields, we need to consider from the beginning an infinite realisation $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L-1} \in \mathbb{R}^{\mathbb{N}^2}$, which therefore depends on α_0 . To re-utilise their notations, consider $(\xi_{i,j})_{(i,j) \in \mathbb{Z}^2}$ the random field defined by i.i.d. samples of $N(0,1)$. We define a function $g : \mathbb{R}^{\mathbb{Z}^2} \rightarrow \mathbb{R}$, again to re-utilise their notation, as follows: g only acts on a portion of $\mathbb{R}^{\mathbb{Z}^2}$, namely $\mathbb{R}^{\mathbb{N}^2}$, and considering $\Xi = (\xi_{i,j})_{(i,j) \in \mathbb{N}^2}$, g is then defined by:

$$g\left((\xi_{i,j})_{(i,j) \in \mathbb{Z}^2}\right) = \left(\sigma\left(\Xi(\Lambda_{\mathbf{X}}^{\sigma,L-1})^{1/2}(\mathbf{U}_{\mathbf{X}}^{\sigma,L-1})^T\right)\right)_{0,0} \quad (7.17)$$

where $\Lambda_{\mathbf{X}}^{\sigma,L-1}$ and $(\mathbf{U}_{\mathbf{X}}^{\sigma,L-1})^T$ is obtained by "diagonalisation" of the infinite matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L-1}$.

Here, we are in a particular case where it is possible to define formally what is meant by "diagonalisation" of an infinite matrix. Indeed, the infinite matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L-1}$ can be constructed step by step: to go from size $n \times n$ to size $n+1 \times n+1$, we sample a new $\mathbf{x}_{n+1}$ from the teacher distribution. Let us note, just for this paragraph, simply $\mathbf{K}_n$ and $\mathbf{K}_{n+1}$ for these two intermediate matrices. The sub-matrix of size $n \times n$, extracted from $\mathbf{K}_{n+1}$, stays unchanged (it is equal to $\mathbf{K}_n$), as we simply add a row and a column, namely $\left(K_{\mathbf{X},\mathbf{x}_{n+1}}^{\sigma,L-1}\right)^T$ and $K_{\mathbf{x}_{n+1},\mathbf{X}}^{\sigma,L-1}$.

We decompose $\frac{1}{d}\mathbf{K}_{n+1}$ into two symmetric matrices $\mathbf{A}$ and $\mathbf{B}$

$$\frac{1}{d}\mathbf{K}_{n+1} = \begin{pmatrix} \frac{1}{d}\mathbf{K}_n & 0 \\ 0 & \dots & 0 \end{pmatrix} + \begin{pmatrix} 0 & \frac{1}{d}K^{\sigma,L-1}(\mathbf{x}_1, \mathbf{x}_{n+1}) \\ \vdots & \vdots \\ \frac{1}{d}K^{\sigma,L-1}(\mathbf{x}_n, \mathbf{x}_{n+1}) & \frac{1}{d}K^{\sigma,L-1}(\mathbf{x}_{n+1}, \mathbf{x}_{n+1}) \end{pmatrix} \quad (7.18)$$

$\mathbf{A}$ and $\mathbf{B}$ are almost commuting with respect to the Frobenius norm.

$$\mathbf{AB} = \begin{pmatrix} \frac{1}{d^2} \sum_{i=1}^n K^{\sigma,L-1}(\mathbf{x}_1, \mathbf{x}_i) K^{\sigma,L-1}(\mathbf{x}_i, \mathbf{x}_{n+1}) \\ \vdots \\ \frac{1}{d^2} \sum_{i=1}^n K^{\sigma,L-1}(\mathbf{x}_n, \mathbf{x}_i) K^{\sigma,L-1}(\mathbf{x}_i, \mathbf{x}_{n+1}) \end{pmatrix} \quad (7.19)$$

7. Mercer Spectrum of Empirical Kernels

and

$$\mathbf{BA} = \begin{pmatrix} 0 & & \\ \frac{1}{d^2} \sum_{i=1}^n K^{\sigma,L-1}(\mathbf{x}_1, \mathbf{x}_i) K^{\sigma,L-1}(\mathbf{x}_i, \mathbf{x}_{n+1}) & \cdots & \frac{1}{d^2} \sum_{i=1}^n K^{\sigma,L-1}(\mathbf{x}_n, \mathbf{x}_i) K^{\sigma,L-1}(\mathbf{x}_i, \mathbf{x}_{n+1}) \end{pmatrix} \quad (7.20)$$

One can verify that the Frobenius norm of $\mathbf{AB} - \mathbf{BA}$ is with high probability of the order $\mathcal{O}_{n,d \rightarrow \infty}(\frac{n^{3/2}}{d^2})$, which is $o_{d \rightarrow \infty}(1)$ (i.e. $\rightarrow 0$), using that $K^{\sigma,L-1}(\mathbf{x}, \mathbf{x}') = \mathcal{O}_{d \rightarrow \infty}(1)$ (i.e. is bounded as $d \rightarrow \infty$) with high probability.

Thus, by (Glashoff & Bronstein 2013), there exists two matrices $\mathbf{A}'$ and $\mathbf{B}'$, which can be arbitrarily close to $\mathbf{A}$ and $\mathbf{B}$ (with respect to the operator norm), such that $\mathbf{A}'$ and $\mathbf{B}'$ commute.

$\mathbf{A}'$ and $\mathbf{B}'$ commute, thus they are co-diagonalisable, i.e. they share an eigenvector basis, which allows for their spectra to be summed. By continuity of the spectrum of a matrix with respect to the operator norm, it suffices to study the spectrum of $\mathbf{A}' + \mathbf{B}'$ in lieu of $\mathbf{A} + \mathbf{B}$. This ensures that the $n + 1$ eigenvalues and the $n + 1$ eigenvectors of $\frac{1}{d}\mathbf{K}_{n+1}$ can be obtained from the n eigenvalues and eigenvectors of $\frac{1}{d}\mathbf{K}_n$ by adding one eigenvalue and concatenating one dimension to the eigenvector at each step. It is thus possible to define sequences of eigenvalues and eigenvectors, respectively $(\lambda_n^{\mathbf{X},\sigma,L-1})_{n \in \mathbb{N}}$ and $(u_n^{\mathbf{X},\sigma,L-1})_{n \in \mathbb{N}}$ to characterize a realisation of the infinite random matrix $K_{\mathbf{X},\mathbf{X}}^{\sigma,L-1}$. Here, the infinite eigenvectors are given by $(u_n^{\mathbf{X},\sigma,L-1})_i = \Phi_n(\mathbf{x}_i)$ with Φ_n the Mercer's eigenfunctions of kernel $K^{\sigma,L-1}$. With these sequences, the entries of the infinite matrix $\tilde{\mathbf{H}}_{\mathbf{X}}^{L-1} := \Xi(\Lambda_{\mathbf{X}}^{\sigma,L-1})^{1/2}(\mathbf{U}_{\mathbf{X}}^{\sigma,L-1})^T$ are well-defined:

$$(\tilde{\mathbf{H}}_{\mathbf{X}}^{L-1})_{i,j} = \sum_{k=0}^{\infty} \xi_{i,k} \sqrt{\lambda_k^{\mathbf{X},\sigma,L-1}} (u_k^{\mathbf{X},\sigma,L-1})_j \quad (7.21)$$

$\Xi(\Lambda_{\mathbf{X}}^{\sigma,L-1})^{1/2}(\mathbf{U}_{\mathbf{X}}^{\sigma,L-1})^T$ is precisely designed to reproduce realisations of the random matrix $\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$ (whose rows are independently sampled from $\mathcal{GP}(0, K^{\sigma,L-1})$) with an underlying array of i.i.d. random variables Ξ . One can indeed verify that we have:

$$\mathbb{E}_{\xi} \left((\tilde{\mathbf{H}}_{\mathbf{X}}^{L-1})_{i,k} (\tilde{\mathbf{H}}_{\mathbf{X}}^{L-1})_{i,l} \right) = K^{\sigma,L-1}(\mathbf{x}_k, \mathbf{x}_l) \quad (7.22)$$

which holds $\forall i, k, l \in \mathbb{N}$.

One can then verify that the careful indexing yields:

$$g \left((\xi_{k-i,l-j})_{(i,j) \in \mathbb{Z}^2} \right) = \left(\sigma(\mathbf{H}_{\mathbf{X}}^{L-1}) \right)_{k,l} \quad (7.23)$$

Note that we are still reasoning conditionally on $\mathbf{X}$ so we should have written $g_{\mathbf{X}}$ instead of g . So far, we thus have that:

$$\left(\sigma(\mathbf{H}_{\mathbf{X}}^{L-1}) \right)_{k,l} | \mathbf{X} \sim g \left(\xi_{k-i,l-j} \right)_{(i,j) \in \mathbb{Z}^2} \quad (7.24)$$

Furthermore, g is a measurable function from $\mathbb{R}^{\mathbb{Z}^2}$ to $\mathbb{R}$, as composition, product and countable sum of measurable functions (Schilling 2005) (σ is measurable by assumption of the present theorem). We have reproduced the random field $\sigma(\mathbf{H}_{\mathbf{X}}^{L-1})$ as a

7. Mercer Spectrum of Empirical Kernels

measurable function of a random field of i.i.d variables, thus the Theorem 5 in (Banna et al. 2015) is applicable and we can study the limiting spectral distribution of the Gram matrix associated with a Gaussian matrix with the same covariance structure, in lieu of the spectral distribution of the associated Gram matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L,d}$.

We also have to prove that the assumptions of Theorem 1 in (Banna et al. 2015) also hold true for $\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L}$, to conclude that they have the same limiting spectral distribution (the common limiting spectral distribution of the Gram matrices of their corresponding Gaussian matrix). The exact same technique can be used to define g in a similar fashion, using the "diagonalisation" of $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L-1}$.

There is then the final stage of relaxing the conditioning on $\mathbf{X}$ to conclude that the random matrices $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L,d}$ and $\frac{1}{d}\mathbf{U}_{\mathbf{X}}^{\sigma,L} \left(\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L} \right)^T \hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L} (\mathbf{U}_{\mathbf{X}}^{\sigma,L})^T$, whose randomness come from $(\mathbf{X}, h)$, have the same limiting spectral measure. It is allowed to do so because by conditioning on $\mathbf{X}$, we have showed that the kernel random matrices of interest have the same limiting spectral distribution *for any infinite realisation of $\mathbf{X}$* , they thus have the same limiting spectral distribution without conditioning.

Thus, the proof can proceed as previously, since we have reduced the problem to studying the existence of the limiting spectral distribution of $\frac{1}{d} \left(\hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L} \right)^T \hat{\mathbf{H}}_{\mathbf{X}}^{\sigma,L}$. The limiting spectral distribution is therefore given by the Marchenko-Pastur map of the limiting spectral distribution of $\Lambda_{\mathbf{X}}^{\sigma,L}$ (if it exists). By assumption of the present theorem, it exists and it is the limiting spectral distribution $\mu_{\alpha_0}^{\sigma}$ of the NNGP kernel matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\sigma,L}$, which terminates the proof. □

7.3. The Modified NNGP Kernel

In this subsection, we start from the empirical kernel, and we study its behavior in the linear-width and sublinear-width regimes.

We assume that the training set $(\mathbf{X}, \mathbf{y})$ consists of n examples sampled i.i.d. from an unknown distribution $\mathbb{P}_{d_0}$ with compact support on $\mathbb{R}^{d_0} \times \mathbb{R}$. We assume that in the limit $d_0 \rightarrow \infty$, $\mathbb{P}_{d_0}$ converges (in distribution) to a well-defined distribution with compact support over $\mathbb{R}^{\mathbb{N}} \times \mathbb{R}$ noted $\lim_{d_0 \rightarrow \infty} \mathbb{P}_{d_0}$. We denote the evaluation of the empirical feature map on the training set by $\phi(\Theta, \mathbf{X}) = (\phi(\Theta, \mathbf{x}_1), \dots, \phi(\Theta, \mathbf{x}_n))^T \in \mathbb{R}^{d \times n}$; this induces an empirical kernel matrix $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ given by $\frac{1}{d}[\phi(\mathbf{X}, \Theta)]^T \phi(\mathbf{X}, \Theta) \in \mathbb{R}^{n \times n}$. Note that this random matrix differs from the previous ones in that all widths (not only the last one) are finite. The training data $\mathbf{X}$ also induces an empirical distribution $p_{\mathbf{X}}(\mathbf{x}) = \frac{1}{n} (\sum_{i=1}^n \delta_{\mathbf{x}_i}(\mathbf{x}))$ with $\delta_{\mathbf{x}_i}$ the Dirac measure.

Assumption 7.4 (Linear-width regime). Assume that $\frac{n}{d} \rightarrow \alpha$ and $\frac{n}{d_0} \rightarrow \alpha_0$ as $n, d, d_0 \rightarrow \infty$ with the ratios $\alpha, \alpha_0 \in (0, +\infty)$.

Assumption 7.5 (Sublinear-width regime). Assume that $\frac{n}{d \cdot d_0} \rightarrow \gamma$ as $n, d, d_0 \rightarrow \infty$ with the ratio $\gamma \in (0, +\infty)$.

7. Mercer Spectrum of Empirical Kernels

Mercer's theorem applied to the random kernel K_{Θ}^{d,d_0} and the data distribution $p_{\mathbf{X}}$ decomposes the kernel into terms of eigenvalues and eigenfunctions as follows: $K_{\Theta}^{d,d_0}(\mathbf{x}, \mathbf{x}') = \sum_k \lambda_k^{n,d,d_0} \phi_k^{n,d,d_0}(\mathbf{x}) \phi_k^{n,d,d_0}(\mathbf{x}')$. This defines a random spectral measure ρ_{Θ}^{n,d,d_0} with spectrum given by the random eigenvalues λ_k^{n,d,d_0} and random eigenfunctions $\phi_k^{n,d,d_0}(\cdot)$. Here, the dependency in n stems from the empirical training data distribution. We use the correspondence between Mercer's eigenvalues and the limiting spectral measure of the corresponding empirical kernel matrix to show that, to estimate the infinite random matrix $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$, there is no need to examine the joint distribution of its eigenvalues, as Mercer's eigenvalues can be sampled independently. This is a crucial observation because the correlations between kernel matrix eigenvalues in the classical eigendecomposition is an obstacle in the computation of the posterior distributions.

Theorem 7.6. *Assume that Assumption 7.4 (respectively, Assumption 7.5) holds. Assume that for each $k \in \mathbb{N}$ there is a random function $\phi_k^{\alpha,\alpha_0} : \mathbb{R}^{\mathbb{N}} \mapsto \mathbb{R}$ (respectively, ϕ_k^{γ}) such that $\phi_k^{n,d,d_0}(\mathbf{x}_i)$ converges in distribution to $\phi_k^{\alpha,\alpha_0}(\mathbf{x})$ (respectively, $\phi_k^{\gamma}(\mathbf{x})$), where $\mathbf{x} \sim \lim_{d_0 \rightarrow \infty} \mathbb{P}_{d_0}$. Assume that the spectrum of $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ (respectively, the strictly positive support of the spectrum) admits a limiting nonrandom measure ρ^{α,α_0} (respectively, ρ^{γ}). Consider the random matrix $\Phi \Lambda \Phi^T$, with $\Phi \in \mathbb{R}^{n \times M}$, $\Phi_{i,k} := \phi_k^{\alpha,\alpha_0}(\tilde{\mathbf{x}}_i)$ (respectively, ϕ_k^{γ}) and $\Lambda \in \mathbb{R}^{M \times M}$, $\Lambda_{k,l} := \delta_{k,l} \lambda_k$ with each λ_k follows independently ρ^{α,α_0} (respectively, ρ^{γ}) and each $\tilde{\mathbf{x}}_i$ follows independently $\lim_{d_0 \rightarrow \infty} \mathbb{P}_{d_0}^1$. Then, the random matrices $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ and $\Phi \Lambda \Phi^T$ converge (in distribution) to the same distribution over $\mathbb{R}^{\mathbb{N} \times \mathbb{N}}$ in the limit $\frac{M}{n} \rightarrow \infty$.*

Proof. In the linear-width (respectively, the sub-linear width limit), the positive semi-definiteness of any matrix extracted from K_{Θ}^{d,d_0} and $p_{\mathbf{X}}$ is maintained (the limit of a positive sequence remains positive), and this suffices to characterise the kernel property over a compact subset of an infinite-dimensional space (Saitoh & Sawano 2016). Thus, there is a random kernel $K_{\Theta}^{\alpha,\alpha_0}$ (respectively, K_{Θ}^{γ}) defined over $\mathbb{R}^{\mathbb{N}}$ which characterises the convergence in distribution of $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$. As per Mercer's theorem, $K_{\Theta}^{\alpha,\alpha_0}$ (respectively, K_{Θ}^{γ}) also defines a random spectral measure $\rho_{\Theta}^{\alpha,\alpha_0}$ (respectively, ρ_{Θ}^{γ}) associated with its Mercer's eigenvalues. By Baker's result (Baker 1977) stating the convergence of eigenvalues in a kernel matrix to the Mercer eigenvalues of the respective kernel, it follows that $\rho_{\Theta}^{\alpha,\alpha_0}$ (respectively, ρ_{Θ}^{γ}) is the limiting spectral distribution of the random matrices $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ in the linear-width limit (respectively, the sublinear-width regime). By assumption, this spectral measure (respectively, the strictly positive support of this spectral measure) is nonrandom $\rho_{\Theta}^{\alpha,\alpha_0} = \rho^{\alpha,\alpha_0}$ (respectively, $\rho_{\Theta}^{\gamma} = \rho^{\gamma}$). Thus, we can reformulate the empirical kernel matrix corresponding to the random kernel $K_{\Theta}^{\alpha,\alpha_0}$ (respectively, K_{Θ}^{γ}) as $\Phi \Lambda \Phi^T$, where λ_k are drawn independently according to ρ^{α,α_0} (respectively, ρ^{γ}). Since the spectral measure no longer depends on Θ , the eigenvalues can be sampled independently from the eigenfunctions. It follows that $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ and $\Phi \Lambda \Phi^T$ converge to the same distribution over $\mathbb{R}^{\mathbb{N} \times \mathbb{N}}$. $\square$

This result is non-trivial and only holds if the spectrum admits a *nonrandom* limit: this is the key argument that allows us to disregard, in the limit, the correlations

¹Expression $\Phi \Lambda \Phi^T$ is not the usual eigendecomposition of a square matrix: the evaluations of eigenfunctions yield rectangular (infinite) matrices. This decomposition is enabled by Mercer's theorem and applies to kernels.

between eigenvalues when using the Mercer decomposition. Note that the distribution of eigenfunctions $\varphi_k^{\alpha, \alpha_0}$ and φ_k^γ are not known in general. We will justify in the next chapter the SUA as a means for alleviating this limitation. Similarly, we will denote with Φ^* evaluations of the eigenfunctions on an unseen data point $\mathbf{x}^*$.

The *modified NNGP kernel* is the random kernel $K_\Theta^{\alpha, \alpha_0}$ (respectively, K_Θ^γ) defined over $\mathbb{R}^{\mathbb{N}}$ in the linear-width regime (respectively, the sublinear-width regime). In the limit, the feature map is not known explicitly, but it must exist by Riesz's representation theorem.

The nonrandom spectral measure is known in the linear-width regime. Observe that, in the linear-width regime, for many cases of interest (including ReLU activations), the limiting spectral measure $\rho_\Theta^{\alpha, \alpha_0}$ indeed no longer depends on Θ and hence becomes a nonrandom measure. To this end, let us first consider the kernel random matrix $\mathbf{K}_{\text{NNGP}}(\mathbf{X}, \mathbf{X})$ associated with the NNGP kernel K_{NNGP} . In the previous section, we have shown that, under mild assumptions on the activation functions σ^ℓ (namely measurability and Lipschitz continuity), $\mathbf{K}_{\text{NNGP}}(\mathbf{X}, \mathbf{X})$ admits a limiting nonrandom spectral measure $\rho_{\text{NNGP}}^{\alpha_0}$ as $n, d_0 \rightarrow \infty$ with constant ratio α_0 ; and furthermore, in the linear-width limit, the limiting spectral distribution of the same random matrix as $\mathbf{K}_\Theta^{n, d, d_0}(\mathbf{X}, \mathbf{X})$ but where the interior widths have already been taken to infinity (i.e. when the linear-width limit only pertains to the last-layer width) is $\rho_{MP}^\alpha \boxtimes \rho_{\text{NNGP}}^{\alpha_0}$. By immediate induction, successively applying the linear-width limit to the hidden-layer widths and keeping the remaining interior widths infinite until reaching the input layer, it follows as a direct corollary of Theorem 7.3 that, in the linear-width limit, $\mathbf{K}_\Theta^{n, d, d_0}(\mathbf{X}, \mathbf{X})$ also admits a limiting nonrandom spectral measure given by the composed Marchenko-Pastur maps $\rho_{MP}^\alpha \boxtimes^L \rho_{\text{NNGP}}^{\alpha_0}$.²

7.4. Experiments

In this section, we provide empirical evidence demonstrating the accuracy of our predictions of the spectral distribution of NNGP kernel random feature matrices. These experiments were carried out using GPU resources on Google Colab.

We have simulated the empirical spectral distribution of the kernel random matrix $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^d$ for high values of d, n, d_0 for ReLU and tanh with both a synthetic dataset with data drawn from an isotropic multivariate Gaussian distribution $\mathbb{P}_{d_0} = N(0, \frac{1}{d_0} I_{d_0})$, and the MNIST dataset (LeCun et al. 1998b).

As illustrated in Figures 7.1 and 7.2, we find excellent agreement with the theoretical prediction of the limiting spectral distributions. We used the Marchenko-Pastur fixed point equation (2.2) to compute the limiting spectral distribution $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$, by iterating over the recursive sequence it defines in the Stieltjes transform space and then inverting the Stieltjes transform using the inversion formula; this numerical method was already detailed in (Fan & Wang 2020). In the case of synthetic data drawn from $\mathbb{P}_{d_0} = N(0, \frac{1}{d_0} I_{d_0})$ and with no nonlinearity, the actual NNGP kernel matrix can be characterised exactly by $\mu_{\alpha_0}^\sigma = \rho_{MP}^{\alpha_0}$. In general, the actual NNGP kernel is not known,

²This result first appeared in the context of neural networks in Fan & Wang (2020). Our result extends it to a more general setting.

7. Mercer Spectrum of Empirical Kernels

hence we estimated the actual NNGP kernel matrix by sampling $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\hat{d}}$ with a very large value of $\hat{d} \gg n$, relying on the fact that $\rho_{MP}^0 \boxtimes \mu_{\alpha_0}^\sigma = \mu_{\alpha_0}^\sigma$. We focused on a subset of MNIST restricted to digits “0” and “1” in order to simplify the structure of the covariance matrices and their spectral distributions.



Figure 7.1.: Simulated empirical spectral distribution (blue histograms) versus theoretical limiting spectral distribution (orange curves) of the empirical covariance matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^{\hat{d}}$. The X-axis are indexed by the eigenvalues of the kernel random matrices by increments of 1 and 2 for the top and bottom figure respectively, and the Y-axis are indexed by the probability density by increments of 0.5 and 0.05 for the top and bottom figures, respectively. Top, we use a two-layer NNGP without non-linearities with teacher distribution $N(0, \frac{1}{d_0} I_{d_0})$ using $d = 300$, $n = 200$, and $d_0 = 400$. Bottom, we use a two-layer ReLU NNGP on a subset of MNIST taking $d = 600$, $n = 300$, and $d_0 = 784$ (pixels on MNIST images). The simulated distribution is obtained by sampling from the random matrix, and the theoretical distribution is obtained by solving the Marchenko-Pastur equation.

7.5. Concluding remarks of the chapter

The limiting spectral distributions of the kernel random matrices that we study in this chapter were first investigated in (Fan & Wang 2020). Our results are, however, stronger since they require less restrictive assumptions on the data generating process and the non-linear activations; for instance, we do not assume the non-linear activation to be twice differentiable nor the columns of the input data matrix to be “ (ϵ, B) -orthonormal”. This was made possible by deriving the analytical expression as a function of an implicit quantity: the spectral measure of the actual NNGP kernel (the assumptions taken in (Fan & Wang 2020) are precisely needed to derive an explicit formula for the spectrum of the actual NNGP kernel). Furthermore, we emphasise that the link between the “CK” kernel random matrix in (Fan & Wang 2020) and the actual NNGP kernel is not straightforward. This subtle distinction enables, in the next chapter, a more transparent interpretation of the double-descent phenomenon. This distinction was already noticed and exploited by (Louart et al. 2018), who found an expression of the limiting spectrum depending on the actual NNGP kernel matrix (not only its spectrum) under very general assumptions. In this chapter, we introduced a novel proof technique enabled by the result of (Banna et al. 2015) which allows us to isolate the dependency in the spectrum under mild assumptions such as boundedness of the second moment of the data distribution. Furthermore, we have generalised this distinction by noticing that

7. Mercer Spectrum of Empirical Kernels

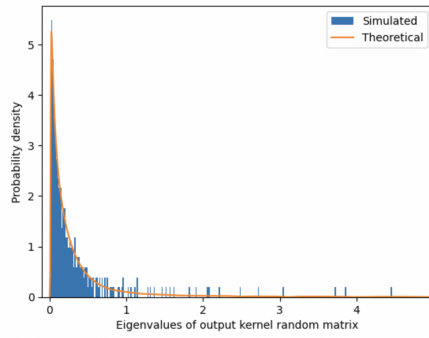


Figure 7.2.: Simulated empirical spectral distribution versus theoretical limiting spectral distribution of the empirical covariance matrix $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$. We use a two-layer tanh NNGP on a subset of MNIST taking $d = 600$, $n = 300$, and $d_0 = 784$ (number of pixels on MNIST images). The simulated distribution is obtained by sampling from the random matrix, and the theoretical distribution is obtained by solving the Marchenko-Pastur fixed-point equation.

every finite neural network readily defines a (parameterized) kernel, and that different kernels may arise depending on which quantities are taken to infinity (and whether they are taken simultaneously or not), as is the case with the modified NNGP kernel. We will see in the subsequent chapters that this viewpoint has proven to be a fruitful perspective for deriving generalisation results.

8. The Double-Descent Phenomenon

The generalisation error of neural networks as a function of the number of parameters, exhibiting double-descent curves, has been studied empirically (Belkin et al. 2019; Nakkiran et al. 2021), and a number of mathematical frameworks for explaining the double-descent phenomenon in a variety of neural architectures have been proposed (Mei & Montanari 2019; Hastie et al. 2019; Advani et al. 2020; Liu et al. 2021). Assuming a teacher-student setting (Seung et al. 1992), these works derive an analytical expression for the generalisation error as a function of the ratios $\alpha = \frac{n}{d}$ and $\alpha_0 = \frac{n}{d_0}$. For a fixed value of α_0 and varying α , the generalisation error derived in these approaches follows a double descent curve; furthermore, these works typically assume the linear-width regime (Assumption 7.4) and that all layers except the last one remain untrained, and thus coincide with random features models (Liao et al. 2020; Gerace et al. 2020).

Contributions. In this chapter, we establish a novel connection between this line of research, and the literature on the NNGP limit for which, as we have seen in the previous chapter, the feature map is not explicitly known and thus its study falls within the realm of the nonparametric kernel literature (Wand & Jones 1995; Györfi et al. 2002). To study the double-descent with NNGPs under the same lens, we leverage our width-dependent NNGP kernel and the limiting spectral distribution of its associated random kernel matrices. Indeed, our expression for the limiting spectral measure is given as a function of the spectral distribution of the NNGP kernel. As a result, our expression enables a new interpretation of the width-dependent spectral distribution as a perturbation of the width-independent spectral distribution that decreases as α tends to zero so that, in the limit of infinite width, the width-dependent spectral distribution converges to that of the actual NNGP. The kernel function and the analytical formula for the spectral distribution that we propose allow us compute the generalisation error of both GP and kernel regression as a function of α and α_0 . Similarly to (Mei & Montanari 2019; Hastie et al. 2019; Advani et al. 2020; Bordelon et al. 2020; Canatar et al. 2021), we also assume the linear-width regime. Furthermore, for a fixed α_0 and varying α , we recover that the generalisation error exhibits double descent behaviour. Our generalisation error expression is novel in that it explicitly isolates the term that contributes to the double-descent behavior.

Our results thus provide a new interpretation to the double-descent phenomenon where the behaviour of the generalisation error is governed by the discrepancy between the width-dependent random kernel characterising the network’s architecture and the width-independent NNGP kernel of the limit Gaussian process.

Problem setup. In this chapter, we consider a teacher-student setting where $\mathbf{x} \sim \mathbb{P}_{d_0}$ as previously, $\tau \sim N(0, \sigma_\tau^2)$ and $y = f_{d_0}(\mathbf{x}) + \tau$, i.e. the output value is obtained from $f_{d_0} : \mathbb{R}^{d_0} \rightarrow \mathbb{R}$ a family of functions verifying that $\mathbb{E}_{\mathbf{x} \sim \mathbb{P}_{d_0}} (f_{d_0}(\mathbf{x})^2)$ is bounded as

8. The Double-Descent Phenomenon

$d_0 \rightarrow \infty$. These assumptions, also taken in (Bordelon et al. 2020; Canatar et al. 2021), are quite mild as they only exclude pathological behaviors where the variances of the input or the output explode at infinity.

We sample a number n of examples, i.i.d. from the teacher. The training set then consists of:

$$\begin{aligned}\mathbf{X} &= (\mathbf{x}_1, \dots, \mathbf{x}_n)^T \in \mathbb{R}^{n \times d} \\ \mathbf{y} &= f_{d_0}(\mathbf{X}) + \mathbf{t} = (y_1, \dots, y_n)^T \in \mathbb{R}^n\end{aligned}\tag{8.1}$$

where $f_{d_0}(\mathbf{X}) = (f_{d_0}(\mathbf{x}_1), \dots, f_{d_0}(\mathbf{x}_n))^T \in \mathbb{R}^n$ and $\mathbf{t} = (\tau_1, \dots, \tau_n)^T \in \mathbb{R}^n$ is a noise sample.

We study the generalisation error in the linear-width regime of particular kernel regressions and Gaussian process regressions trained on this data.

8.1. Double Descent from the Perspective of Gaussian Processes Kernels

In the following, we isolate the dependency to the spectral distributions (computed in the previous chapter 7) in the generalisation errors of GP regression and kernel regression. This allows us to interpret the double-descent phenomenon concerning α , attributing it to the spectrum of the related random kernel being a perturbation of the NNGP kernel limit's spectrum. The degree of this perturbation varies depending on α and this variation is mirrored in the spectral distributions.

Armed with the random kernel K^d , we are now in position to study the generalisation error of the corresponding Gaussian process and kernel regressions. We will calculate the generalisation error of kernel regression with kernel K^d :

$$E_{\mathcal{K}}(K^d) := \mathbb{E}_{\mathbf{x}, \mathbf{y}, \mathbf{X}, \mathbf{Y}} \left((\bar{\mu}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^{K^d} - y)^2 \right)\tag{8.2}$$

and the generalisation error of Gaussian process regression with GP g_{K^d} :

$$E_{\mathcal{GP}}(K^d) := \mathbb{E}_{\mathbf{x}, \mathbf{y}, \mathbf{X}} \left(\mathbb{E}_{\mathbf{y}, \bar{g}_{K^d}} ((\bar{g}_{K^d} - y)^2 | \mathbf{X}, \mathbf{y}, \mathbf{x}) \right)\tag{8.3}$$

where $\bar{\mu}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^{K^d}$ is the prediction mean of the Gaussian process regression with prior g_{K^d} , $\bar{g}_{K^d} | \mathbf{X}, \mathbf{y}, \mathbf{x}$ is the posterior distribution of Gaussian process regression with prior g_{K^d} , and the expectations are taken over the out-of-sample data and the training samples. These predictions depend on the realisation of the kernel function K^d . We study these generalisation errors when all quantities go to infinity and averaging over the random kernel using $n = \alpha d$, and $d_0 = \frac{\alpha d}{\alpha_0}$:

$$\begin{aligned}E_{\mathcal{K}}(\alpha, \alpha_0) &:= \lim_{d \rightarrow \infty} \mathbb{E}_{K^d} \left(E_{\mathcal{K}}(K^d) \right) \\ E_{\mathcal{GP}}(\alpha, \alpha_0) &:= \lim_{d \rightarrow \infty} \mathbb{E}_{K^d} \left(E_{\mathcal{GP}}(K^d) \right)\end{aligned}\tag{8.4}$$

The following theorem highlights the dependence of the generalisation errors with some terms of interest that solely depend on the spectral measure that we studied in the previous section. The limits of these spectral measures will give us the double-descent behavior.

8. The Double-Descent Phenomenon

Theorem 8.1. *Under the same assumptions as in our Theorem 7.3, the limiting generalisation errors $E_{\mathcal{K}}(\alpha, \alpha_0)$ and $E_{\mathcal{GP}}(\alpha, \alpha_0)$ can be expressed:*

$$E_{\mathcal{K}}(\alpha, \alpha_0) = D(\alpha, \alpha_0) + C(\alpha, \alpha_0)g(\alpha, \alpha_0)^2 + B(\alpha, \alpha_0)g_2(\alpha, \alpha_0) + A(\alpha, \alpha_0)g(\alpha, \alpha_0) \quad (8.5)$$

$$E_{\mathcal{GP}}(\alpha, \alpha_0) = \bar{D}(\alpha, \alpha_0) + C(\alpha, \alpha_0)g(\alpha, \alpha_0)^2 + B(\alpha, \alpha_0)g_2(\alpha, \alpha_0) + \bar{A}(\alpha, \alpha_0)g(\alpha, \alpha_0) \quad (8.6)$$

where:

$$g(\alpha, \alpha_0) := \int_0^\infty \frac{1}{\lambda + \sigma_\epsilon^2} d(\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma)(\lambda) \\ g_2(\alpha, \alpha_0) := \int_0^\infty \frac{1}{(\lambda + \sigma_\epsilon^2)^2} d(\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma)(\lambda) \quad (8.7)$$

and $A(\alpha, \alpha_0), \bar{A}(\alpha, \alpha_0), B(\alpha, \alpha_0), C(\alpha, \alpha_0), D(\alpha, \alpha_0), \bar{D}(\alpha, \alpha_0)$ are bounded with respect to $n, d, d_0 \rightarrow \infty$, and $B(\alpha, \alpha_0)$ is non-zero.

The proof of Theorem 8.1 relies on the diagonalisation of the kernel random matrix $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^d$ and exploits Theorem 7.3 to compute the expectation of the inverse of the eigenvalues in the limit of infinite quantities.

Proof. The generalisation error of kernel regression with kernel K^d is given by:

$$E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y, \bar{g}_{K^d}) := y^2 - 2y \bar{\mu}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^{K^d} + (\bar{\mu}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^{K^d})^2 \quad (8.8)$$

Similarly, the generalisation error of Gaussian process regression with GP prior $g_{K^d} \sim \mathcal{GP}(0, K^d)$ is given by:

$$E_{\mathcal{GP}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y, \bar{g}_{K^d}) := y^2 - 2y \bar{g}_{K^d} + (\bar{g}_{K^d})^2 \quad (8.9)$$

Averaging over the prediction distribution $\bar{g}_{K^d}$ yields (using the formulas of GP regression):

$$E_{\mathcal{GP}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y) = y^2 - 2y \bar{\mu}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^{K^d} + (\bar{\mu}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^{K^d})^2 + \bar{K}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^d \\ = E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y) + \bar{K}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^d \quad (8.10)$$

We will first study $E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y)$, and then $\bar{K}_{\mathbf{X}, \mathbf{y}, \mathbf{x}}^d$.

Conditionally on $\mathbf{x}, y, \mathbf{X}, \mathbf{y}$, we expand $E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y)$ using the formula for the prediction mean of Gaussian process regression:

$$E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y) := y^2 - 2y(K_{\mathbf{x}, \mathbf{X}}^d)^T \left(K_{\mathbf{X}, \mathbf{X}}^d + \sigma_\epsilon^2 \mathbf{I}_n \right)^{-1} \mathbf{y} \\ + (K_{\mathbf{x}, \mathbf{X}}^d)^T \left(K_{\mathbf{X}, \mathbf{X}}^d + \sigma_\epsilon^2 \mathbf{I}_n \right)^{-1} \mathbf{y} \mathbf{y}^T \left(K_{\mathbf{X}, \mathbf{X}}^d + \sigma_\epsilon^2 \mathbf{I}_n \right)^{-1} K_{\mathbf{x}, \mathbf{X}}^d \quad (8.11)$$

8. The Double-Descent Phenomenon

Averaging over the out-of-sample data $\mathbf{x}$ and y , this yields:

$$\begin{aligned} E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}) &:= \mathbb{E}_{\mathbf{x}, y} \left(E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}, \mathbf{x}, y) \right) \\ &= \mathbb{E}_{\mathbf{x}} \left(f_{d_0}(\mathbf{x})^2 \right) + \sigma_{\tau}^2 - 2 \operatorname{Tr} \left(\mathbb{E}_{\mathbf{x}} \left(f_{d_0}(\mathbf{x}) K_{\mathbf{x}, \mathbf{x}}^d \right) \mathbf{y}^T \left(K_{\mathbf{x}, \mathbf{x}}^d + \sigma_{\epsilon}^2 \mathbf{I}_n \right)^{-1} \right) \\ &\quad + \operatorname{Tr} \left(\mathbb{E}_{\mathbf{x}} \left(K_{\mathbf{x}, \mathbf{x}}^d (K_{\mathbf{x}, \mathbf{x}}^d)^T \right) \left(K_{\mathbf{x}, \mathbf{x}}^d + \sigma_{\epsilon}^2 \mathbf{I}_n \right)^{-1} \mathbf{y} \mathbf{y}^T \left(K_{\mathbf{x}, \mathbf{x}}^d + \sigma_{\epsilon}^2 \mathbf{I}_n \right)^{-1} \right) \end{aligned} \quad (8.12)$$

where we have used the property $\mathbf{a}^T \mathbf{b} = \operatorname{Tr}(\mathbf{a} \mathbf{b}^T)$.

We use the Mercer's decomposition (Minh et al. 2006) of the random kernel K^d with data distribution $\mathbb{P}_{d_0}$, which gives us:

$$K_{\mathbf{x}, \mathbf{x}}^d = \Phi \Lambda \Phi^T \quad (8.13)$$

with $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_M)$ the Mercer's eigenvalues, and $\Phi_{i,j} = \Phi_j(\mathbf{x}_i)$ the Mercer's eigenfunctions evaluated at training examples. Note that $\Lambda \in \mathbb{R}^{M \times M}$, and $\Phi \in \mathbb{R}^{n \times M}$ with $\frac{n}{M} \rightarrow 0$ as $n, M \rightarrow \infty$ (i.e. M going to infinity at a much faster rate than n). More importantly, note that here, kernel eigenvalues λ_i do depend on the data distribution $\mathbb{P}_{d_0}$ by Mercer's theorem, but not on the particular realisation of the training set $\mathbf{X}$. The randomness of λ_i only comes from K^d . We then have:

$$\begin{aligned} E_{\mathcal{K}}(K^d, \mathbf{X}, \mathbf{y}) &= \mathbb{E}_{\mathbf{x}} \left(f_{d_0}(\mathbf{x})^2 \right) + \sigma_{\tau}^2 - \mathbb{E}_{\mathbf{x}} \left(f_{d_0}(\mathbf{x}) \sum_i^M \frac{1}{\lambda_i + \sigma_{\epsilon}^2} \left(\Phi^{\dagger} \mathbf{y} (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^{\dagger})^T \right)_{i,i} \right) \\ &\quad + \sum_i^M \frac{1}{(\lambda_i + \sigma_{\epsilon}^2)^2} \left(\Phi^{\dagger} \mathbf{y} \mathbf{y}^T (\Phi^{\dagger})^T \right)_{i,i} \mathbb{E}_{\mathbf{x}} \left(\left(\Phi^{\dagger} \mathbf{k}_{\mathbf{x}, \mathbf{x}}^d (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^{\dagger})^T \right)_{i,i} \right) \\ &\quad + \sum_i^M \sum_{j \neq i}^M \frac{1}{(\lambda_i + \sigma_{\epsilon}^2)(\lambda_j + \sigma_{\epsilon}^2)} \left(\Phi^{\dagger} \mathbf{y} \mathbf{y}^T (\Phi^{\dagger})^T \right)_{i,j} \mathbb{E}_{\mathbf{x}} \left(\left(\Phi^{\dagger} \mathbf{k}_{\mathbf{x}, \mathbf{x}}^d (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^{\dagger})^T \right)_{i,j} \right) \end{aligned} \quad (8.14)$$

where $\Phi^{\dagger}$ is the Moore-Penrose pseudo-inverse of Φ .

To relax the conditioning on K^d , our objective is to utilise that, in the limit of infinite quantities $d, n, d_0 \rightarrow \infty$, the expectations of $\frac{1}{\lambda_i + \sigma_{\epsilon}^2}$ and $\frac{1}{(\lambda_i + \sigma_{\epsilon}^2)^2}$, with respect to K^d , are easy to calculate: they are respectively given by $g(\alpha, \alpha_0)$ and $g_2(\alpha, \alpha_0)$ (see Equation (8.7)). But this cannot be used directly because in the three sums in equation (8.14), the $\frac{1}{\lambda_i}$ and $\frac{1}{\lambda_j}$ are multiplied by functions of $\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d$ which also have a dependency on λ_i, λ_j .

To tackle this issue, we rewrite $\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d = \Phi \Lambda \Phi^*$ with $\Phi_j^* = \Phi_j(\mathbf{x})$ the eigenfunction evaluated at the test point $\mathbf{x}$ and we decompose the different quantities:

$$\begin{aligned} \left(\Phi^{\dagger} \mathbf{y} (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^{\dagger})^T \right)_{i,i} &= \sum_{k,l} \left(\Phi^{\dagger} \mathbf{y} (\Phi^*)^T \right)_{i,k} \Lambda_{k,l} \left(\Phi^T (\Phi^{\dagger})^T \right)_{l,i} \\ &= \sum_k \lambda_k \left(\Phi^{\dagger} \mathbf{y} (\Phi^*)^T \right)_{i,k} \left(\Phi^{\dagger} \Phi \right)_{i,k} \end{aligned} \quad (8.15)$$

8. The Double-Descent Phenomenon

The first sum can therefore be separated into two parts:

$$\begin{aligned} \sum_i^M \frac{1}{\lambda_i + \sigma_\epsilon^2} \left(\Phi^\dagger \mathbf{y} (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^\dagger)^T \right)_{i,i} &= \sum_i^M \frac{\lambda_i}{\lambda_i + \sigma_\epsilon^2} \left(\Phi^\dagger \mathbf{y} (\Phi^*)^T \right)_{i,i} \left(\Phi^\dagger \Phi \right)_{i,i} \\ &\quad + \sum_i^M \frac{1}{\lambda_i + \sigma_\epsilon^2} \sum_{k \neq i} \lambda_k \left(\Phi^\dagger \mathbf{y} (\Phi^*)^T \right)_{i,k} \left(\Phi^\dagger \Phi \right)_{i,k} \end{aligned} \quad (8.16)$$

We have thus isolated the term $\frac{1}{\lambda_i + \sigma_\epsilon^2}$ in the second part which allows us to calculate the sum using the integral $\int_0^\infty \frac{1}{\lambda + \sigma_\epsilon^2} d(\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma)(\lambda)$ when averaging over K^d .

The same trick can be used to decompose the other terms in the sums:

$$\left(\Phi^\dagger \mathbf{k}_{\mathbf{x}, \mathbf{x}}^d (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^\dagger)^T \right)_{i,j} = \sum_{k_1, k_2} \lambda_{k_1} \lambda_{k_2} \left(\Phi^* (\Phi^*)^T \right)_{k_1, k_2} \left(\Phi^\dagger \Phi \right)_{i, k_1} \left(\Phi^\dagger \Phi \right)_{j, k_2} \quad (8.17)$$

which allows us to isolate the relevant terms and integrate the spectral measures when averaging over K^d . The last step to obtain Equation (8.5) is to check the terms remaining in front of each of the relevant quantities that we have isolated, namely $\frac{1}{\lambda_i + \sigma_\epsilon^2}$ for $A(\alpha, \alpha_0)$, $\frac{1}{\lambda_i^2 + \sigma_\epsilon^2}$ for $B(\alpha, \alpha_0)$ and $\frac{1}{(\lambda_i + \sigma_\epsilon^2)^2}$ for $C(\alpha, \alpha_0)$ and take the limit of these terms. $D(\alpha, \alpha_0)$ is the limit of the remaining terms.

The same techniques can be used to study $\bar{K}_{\mathbf{x}, \mathbf{y}, \mathbf{x}}^d$. We obtain:

$$\mathbb{E}_x \left(\bar{K}_{\mathbf{x}, \mathbf{y}, \mathbf{x}}^d \right) = \mathbb{E}_x \left(K^d(\mathbf{x}, \mathbf{x}) \right) - \sum_i^M \frac{1}{(\lambda_i + \sigma_\epsilon^2)} \mathbb{E}_x \left(\left(\Phi^\dagger \mathbf{k}_{\mathbf{x}, \mathbf{x}}^d (\mathbf{k}_{\mathbf{x}, \mathbf{x}}^d)^T (\Phi^\dagger)^T \right)_{i,i} \right) \quad (8.18)$$

where the average of $\frac{1}{(\lambda_i + \sigma_\epsilon^2)}$ can be isolated in the same way as previously, giving us the expression in Equation (8.6).

Lastly, it is tedious but rather straightforward to show that the quantities:

$(\Phi^\dagger \mathbf{y} (\Phi^*)^T)_{i,k}$, $(\Phi^\dagger \Phi)_{i,k}$ and $(\Phi^* (\Phi^*)^T)_{k_1, k_2}$, $(\Phi^\dagger \Phi)_{i, k_1}$, $(\Phi^\dagger \Phi)_{j, k_2}$ remain bounded when $n, d, d_0 \rightarrow \infty$, and, in particular, when $i = j$, that the expectation is non-zero. One way to calculate it is to develop the terms further to deal with sums over all entries of matrices Φ and $\Phi^\dagger$, notice that in the limit of large sample size, rows of Φ become orthogonal, thus its Moore-Penrose pseudo-inverse can be calculated as follows $\Phi^\dagger = \Phi^T (\Phi \Phi^T)^{-1}$ (Moore 1920), and invoke that $(\Phi \Phi^T)^{-1}$ follows the well-known inverse Wishart distribution (Livan et al. 2018; Karoui 2010).

□

Note that we do not need to compute the coefficients A , B , C and D analytically. Indeed, our objective is not to provide a fine-grained analysis of the generalisation error, for which other expressions are already available in the literature (Jacot et al. 2020; Canatar et al. 2021; Simon et al. 2023) and others), but rather to isolate the terms that contribute to the coarse-grained double-descent behavior, in order to provide a neat interpretation. Indeed, the boundedness of A , B , C and D and the boundedness away from zero of B are sufficient to capture the double descent behaviour as per the following corollary.

8. The Double-Descent Phenomenon

Corollary 8.2. *Suppose that the assumptions of Theorem 7.3 hold true. Then, in the limit of $\sigma_\epsilon \rightarrow 0$ (noise-free), the generalisation error $E_K(\alpha, \alpha_0)$ exhibits a double descent with respect to α . More precisely, the asymptote for the underparametrised regime is given by:*

$$\lim_{\alpha \rightarrow \infty} E_K(\alpha, \alpha_0) = \lim_{\alpha \rightarrow \infty} D(\alpha, \alpha_0). \quad (8.19)$$

The asymptote for the interpolation threshold is given by:

$$\lim_{\alpha \rightarrow 1^-} E_K(\alpha, \alpha_0) = \infty \quad \lim_{\alpha \rightarrow 1^+} E_K(\alpha, \alpha_0) = \infty. \quad (8.20)$$

Finally the asymptote $\lim_{\alpha \rightarrow 0} E_K(\alpha, \alpha_0)$ for the overparametrised regime is finite and given by

$$\begin{aligned} D(0, \alpha_0) + C(0, \alpha_0) & \left(\int_0^\infty \frac{1}{\lambda} d(\mu_{\alpha_0}^\sigma)(\lambda) \right)^2 \\ & + B(0, \alpha_0) \int_0^\infty \frac{1}{\lambda^2} d(\mu_{\alpha_0}^\sigma)(\lambda) \\ & + A(0, \alpha_0) \int_0^\infty \frac{1}{\lambda} d(\mu_{\alpha_0}^\sigma)(\lambda) \end{aligned}$$

The result also holds for $E_{GP}(\alpha, \alpha_0)$, replacing $A(\alpha, \alpha_0), D(\alpha, \alpha_0)$ by $\bar{A}(\alpha, \alpha_0), \bar{D}(\alpha, \alpha_0)$.

The possibility of convergence to a finite value in the overparametrised regime is enabled by the behavior of the Marchenko-Pastur map, as explained by equation (7.11). Indeed, the empirical spectral distribution converges to that of the *actual* NNGP kernel matrix. The divergence at the interpolation threshold is due to eigenvalues becoming arbitrarily close to zero, due to a structural property independent of the input data distribution: the strictly positive support of the nonrandom measure $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$ becomes arbitrarily close to zero when $\alpha \rightarrow 1$. In practice, the divergence is reduced by the effects of regularisation (in the case of NNGP regression, the noise model).

Proof. Note that the assumption of the noise-free limit $\sigma_\epsilon \rightarrow 0$ is motivated by the fact that it is known (Rasmussen & Williams 2006) that the predicted mean of Gaussian process regression is the same as the prediction of kernel ridge regression with the same kernel K and a ridge parameter of σ_ϵ^2 , hence a noise model is some form of ridge regularisation, when we actually want to recover the double descent in the ridgeless case, in order to have a proper divergence at the interpolation threshold.

In the noise-free limit, since the kernel random matrix $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^d \in \mathbb{R}^{n \times n}$ is not necessarily invertible (it is at most of rank $\min(n, d)$ which is $\leq n$), we need to consider the generalised inversion in the formula for the prediction mean of Gaussian process regression, namely the Moore-Penrose pseudo-inverse ((Moore 1920)). In this case, we integrate only over the strictly positive part of the spectral measures in Equation (8.7). This stems from the fact that the eigendecomposition of the Moore-Penrose pseudo-inverse of a matrix with eigendecomposition $\mathbf{V} \mathbf{\Lambda} \mathbf{V}^T$ is obtained by transposing $\mathbf{V}$ and $\mathbf{V}^T$, inverting the non-zero eigenvalues, and leaving in place the zero eigenvalues.

We next derive the behavior, with respect to α , of the limiting generalisation errors $E_K(\alpha, \alpha_0)$ and $E_{GP}(\alpha, \alpha_0)$.

8. The Double-Descent Phenomenon

Underparameterized regime In the limit $\alpha \rightarrow \infty$, the Marchenko-Pastur fixed-point equation (2.2) becomes $S(z) = 0$. The Stieltjes inversion formula tells us that the solution is a probability measure with density 0 for all $x \neq 0$: it is the Dirac measure at 0. Plugging this into Equation (8.7), and integrating over the strictly positive part of the spectral measures (we are precisely in the case where the rank of $\mathbf{K}_{\mathbf{X},\mathbf{X}}^d$ is $< n$, since $\frac{n}{d} \rightarrow \infty$), we recover Equation (8.19).

Overparameterised regime As per Equation (7.11), the spectral measure of interest is that of the *actual* NNGP kernel matrix. Plugging this into Equation (8.7), we recover Corollary 8.2.

Interpolation threshold For the case $\alpha \rightarrow 1$, we will show that $\lambda \mapsto \frac{1}{\lambda^2}$ is not integrable with respect to $\rho_{MP}^1 \boxtimes \mu_{\alpha_0}^\sigma$. The limits $\alpha \rightarrow 1^-$ and $\alpha \rightarrow 1^+$ yield the same result, simply by integrating only over the strictly positive part of $\rho_{MP}^1 \boxtimes \mu_{\alpha_0}^\sigma$ in the case $\alpha = \frac{n}{d} \rightarrow 1^+$. Indeed, in $B(\alpha, \alpha_0) \frac{1}{\lambda^2} + A(\alpha, \alpha_0) \frac{1}{\lambda}$ and $B(\alpha, \alpha_0) \frac{1}{\lambda^2} + \bar{A}(\alpha, \alpha_0) \frac{1}{\lambda}$, the terms in $\frac{1}{\lambda^2}$ dominate near 0, and they are thus the ones that will give us the divergence.

First, we notice that $\lambda \mapsto \frac{1}{\lambda^2}$ is not integrable with respect to the simple Marchenko-Pastur distribution ρ_{MP}^1 (which is the limiting spectral measure of $\mathbf{M}^T \mathbf{M}$, with the same notations as in equation (7.13), with $\alpha = 1$). Indeed, with the analytical formula for the Marchenko-Pastur distribution (Marchenko & Pastur 1967):

$$\rho_{MP}^1(\lambda) = \frac{1}{2\pi\lambda} \sqrt{\lambda(2-\lambda)} \quad (8.21)$$

we have that $\frac{1}{\lambda^{1/2}} = \mathcal{O}_{\lambda \rightarrow 0}(\rho_{MP}^1(\lambda))$, thus $\frac{1}{\lambda^{5/2}} = \mathcal{O}_{\lambda \rightarrow 0}(\frac{1}{\lambda^2} \rho_{MP}^1(\lambda))$, and $\frac{1}{\lambda^{5/2}}$ is not integrable at 0 (convergence of Riemann integrals), thus $\frac{1}{\lambda^2} \rho_{MP}^1(\lambda)$ is not integrable at 0.

The assumption on the integrability of $\lambda \mapsto \frac{1}{\lambda}$ and $\lambda \mapsto \frac{1}{\lambda^2}$ with respect to the spectral measure $\mu_{\alpha_0}^\sigma$ implies that, in the limit of infinite quantities, $\Lambda_{\mathbf{X}}$, with the same notations as in equation (7.13), is almost surely invertible. Indeed, if it had a non-zero probability of having a zero eigenvalue, in other words if we did not have $\mu_{\alpha_0}^\sigma(\lambda) = o_{\lambda \rightarrow 0}(1)$, $\lambda \mapsto \frac{1}{\lambda}$ would not have been integrable at 0.

Next, we show that the Marchenko-Pastur map of $\mu_{\alpha_0}^\sigma$ with $\alpha = 1$ can only worsen the non-integrability at 0. Precisely, we show that $\rho_{MP}^1(\lambda) = \mathcal{O}_{\lambda \rightarrow 0}((\rho_{MP}^1 \boxtimes \mu_{\alpha_0}^\sigma)(\lambda))$. To do so, using the same notations as in equation (7.13), we show that if, in the infinite limit, $\mathbf{M}^T \mathbf{M}$ has an arbitrarily small eigenvalue $\epsilon > 0$ with probability p , then $\Lambda_{\mathbf{X}}^{1/2} \mathbf{M}^T \mathbf{M} \Lambda_{\mathbf{X}}^{1/2}$ has, with probability p , a smaller eigenvalue (this way the density near zero of the spectral measure of $\mathbf{M}^T \mathbf{M}$ is dominated). Suppose that $\mathbf{M}^T \mathbf{M}$ has a small eigenvalue $\epsilon > 0$ with associated eigenvector e . Note λ_{min}^ϵ the smallest eigenvalue of $\Lambda_{\mathbf{X}}^{1/2} \mathbf{M}^T \mathbf{M} \Lambda_{\mathbf{X}}^{1/2}$, conditional on the existence of an eigenvalue ϵ for the random matrix $\mathbf{M}^T \mathbf{M}$. We have:

$$\begin{aligned} \left(\Lambda_{\mathbf{X}}^{-1/2} e \right)^T \Lambda_{\mathbf{X}}^{1/2} \mathbf{M}^T \mathbf{M} \Lambda_{\mathbf{X}}^{1/2} \left(\Lambda_{\mathbf{X}}^{-1/2} e \right) &= \epsilon \|e\|^2 = \epsilon \\ &\geq \lambda_{min}^\epsilon \| \Lambda_{\mathbf{X}}^{-1/2} e \|^2 \end{aligned} \quad (8.22)$$

where we have used that e is an orthonormal eigenvector of $\mathbf{M}^T \mathbf{M}$ with eigenvalue ϵ , and the standard inequality $\forall \mathbf{x} \in \mathbb{R}^n : \mathbf{x}^T \mathbf{A} \mathbf{x} \geq \lambda_{min}^{\mathbf{A}} \|\mathbf{x}\|^2$, which holds for any real

8. The Double-Descent Phenomenon

symmetric matrix $\mathbf{A}$. We can express the norm of the vector of interest, calculated in the eigenvector basis of $\mathbf{M}^T \mathbf{M}$:

$$\|\Lambda_{\mathbf{X}}^{-1/2} e\|^2 = \sum_i^n \frac{1}{\lambda_i} \quad (8.23)$$

with λ_i the eigenvalues of $\Lambda_{\mathbf{X}}$. By the law of large numbers, we have:

$$\lim_{n \rightarrow \infty} \frac{1}{n} \sum_i^n \frac{1}{\lambda_i} = \int_0^\infty \frac{1}{\lambda} d(\mu_{\alpha_0}^\sigma)(\lambda) \quad (8.24)$$

almost surely, and the limit is finite, by assumption on $\mu_{\alpha_0}^\sigma$ (limiting spectral measure of $\Lambda_{\mathbf{X}}$). Thus, we have $\lim_{n \rightarrow \infty} \|\Lambda_{\mathbf{X}}^{-1/2} e\|^2 = \infty$ almost surely. Note that we are still conditioned on $\mathbf{M}$, the “almost surely” statement refers to the randomness of $\mathbf{X}$. Therefore, in the limit of infinite quantities, we have almost surely:

$$\epsilon \geq \frac{\epsilon}{\|\Lambda_{\mathbf{X}}^{-1/2} e\|^2} \geq \lambda_{\min}^\epsilon \quad (8.25)$$

Relaxing the conditioning on the existence of a eigenvalue ϵ for $\mathbf{M}^T \mathbf{M}$, we have thus found, with probability p , a smaller eigenvalue $\lambda_{\min}^\epsilon$ for $\Lambda_{\mathbf{X}}^{1/2} \mathbf{M}^T \mathbf{M} \Lambda_{\mathbf{X}}^{1/2}$, which gives us the result we wanted. Now, since the Marchenko-Pastur map (with $\alpha = 1$) of $\mu_{\alpha_0}^\sigma$ dominates the Marchenko-Pastur distribution $\rho_{MP}^1(\lambda)$ near zero, it is not integrable at zero either. This gives us the result:

$$\begin{aligned} \lim_{\alpha \rightarrow 1} E_{\mathcal{K}}(\alpha, \alpha_0) &= \infty \\ \lim_{\alpha \rightarrow 1} E_{\mathcal{GP}}(\alpha, \alpha_0) &= \infty \end{aligned} \quad (8.26)$$

□

8.2. Experiments

In this section, we provide empirical evidence demonstrating the manifestation of the double-descent phenomenon in the generalisation error of NNGP kernel regression. These experiments were carried out using GPU resources on Google Colab.

We have simulated the generalisation errors of NNGP kernel regression both a synthetic dataset with data drawn from an isotropic multivariate Gaussian distribution $\mathbb{P}_{d_0} = N(0, \frac{1}{d_0} I_{d_0})$, and on the MNIST dataset (LeCun et al. 1998b). To calculate the generalisation errors, we relied on the spectral universality assumption (SUA) (Sollich & Halees 2002) to estimate eigenfunctions (and hence coefficients $A(\alpha, \alpha_0), B(\alpha, \alpha_0), C(\alpha, \alpha_0), D(\alpha, \alpha_0)$), which states that in high dimension eigenfunctions become unstructured and can be approximated by independent Gaussian entries. Determining in which cases the SUA is valid is still an active area of research, and an important topic in the kernel literature (Sollich & Halees 2002; Karoui 2010; Cheng & Singer 2013; Fan & Montanari 2015; Liu et al. 2021; Simon et al. 2023; Lu & Yau 2023). In the case of isotropic data and no nonlinearity, the SUA is exact (Karoui 2010), which helps explain the very good agreement shown in Figure 8.1.

8. The Double-Descent Phenomenon

In the case of MNIST with ReLU, there is evidence that the SUA does apply to some extent, see e.g. (Simon et al. 2023) who also use SUA to estimate generalisation errors of kernel regression in high dimensions on MNIST. Note that, while we rely on the SUA to provide numerical values for the coefficients

$$A(\alpha, \alpha_0), \bar{A}(\alpha, \alpha_0), B(\alpha, \alpha_0), C(\alpha, \alpha_0), D(\alpha, \alpha_0), \bar{D}(\alpha, \alpha_0) ;$$

our general theoretical results only require boundedness of these coefficients. As can be seen in Figure 8.1, we find reasonable agreement. The main sources of discrepancies stem from the fact that we only use a small subset (300 examples) to estimate the empirical spectral distribution and that the SUA may not be completely accurate in this particular setting. Although our predictions for the generalisation error are not perfectly accurate, we emphasise that they correctly predict the double-descent phenomenon, and thus support our claim that the double-descent phenomenon is only driven by the spectral distribution and its dependence with the width.



Figure 8.1.: Simulated generalisation error (red curves) versus theoretical limiting generalisation error (green curves), as a function of $\frac{1}{\alpha}$. The X-axis are indexed by $\frac{1}{\alpha}$ by increments of 2 and 0.5 for the left and right figure respectively, and the Y-axis are indexed by the generalisation error by increments of 0.5 and 1 for the left and right figure respectively. Left, the simulated error is obtained by sampling from the kernel random matrix of a single-hidden-layer NNGP with no non-linearity under a teacher distribution $N(0, \frac{1}{d_0} I_{d_0})$. The theoretical distribution and asymptotes (underparameterized asymptote in orange and overparameterized asymptote in blue) are obtained by integrating $\frac{1}{\lambda}$ and $\frac{1}{\lambda^2}$ over the map of the distribution $\rho_{MP}^{\alpha_0}$. Right, the simulated error is obtained by sampling from the kernel random matrix of a two-hidden-layer ReLU NNGP on a subset of MNIST, and the theoretical distribution is obtained by sampling from the Marchenko-Pastur map of the empirical NNGP kernel matrix $\mathbf{K}_{\mathbf{X}, \mathbf{X}}^{\hat{d}}$, and from independent Gaussian distributions in lieu of eigenfunctions.

Further experimental validations. Our results about generalisation errors of kernel and GP regression, including the double-descent phenomenon, rely on the fact that the random width-dependent NNGP kernel K^d can be written:

$$K^d(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Lambda \Phi(\mathbf{x}') \quad (8.27)$$

8. The Double-Descent Phenomenon

where $\Phi(\mathbf{x}) := (\Phi_1(\mathbf{x}), \dots, \Phi_M(\mathbf{x}))$, $\Phi(\mathbf{x}') := (\Phi_1(\mathbf{x}'), \dots, \Phi_M(\mathbf{x}'))$ the Mercer's eigenfunctions evaluated at points $\mathbf{x}, \mathbf{x}'$, and $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_M)$ with λ_i sampled independently from $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$. We have detailed in the previous chapter the numerical method we used to sample from $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$.

To estimate the eigenfunctions Φ_i , we rely on the SUA. In Figures 8.2 and 8.3, we illustrate to what extent this approximation and our equation (8.27) is correct on diagonal and off-diagonal elements of the random kernel matrix. This allows us to plot the distribution of generalisation errors of the corresponding kernel regressions (Figure 8.4). As expected, the spectral universality assumption provides a close match for (width-dependent) NNGP kernel with isotropic data and no nonlinearity, for which it has been proven to be exactly correct in infinite dimensions (Karoui 2010). The new insight here is to combine the SUA with the spectral distribution $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$. On the other hand, there are some discrepancies in the case of MNIST with ReLU nonlinearity but the overall agreement in terms of generalisation error is acceptable (Figure 8.4). This observation is consistent with that of (Simon et al. 2023). We then used these generalisation error estimates for varying values of $\alpha = \frac{n}{d}$ to construct the double-descent curves Figure 8.1. Note however, that although without the SUA, we would have no way to estimate coefficients $A(\alpha, \alpha_0), \bar{A}(\alpha, \alpha_0), B(\alpha, \alpha_0), C(\alpha, \alpha_0), D(\alpha, \alpha_0), \bar{D}(\alpha, \alpha_0)$, as long as they are bounded and $B(\alpha, \alpha_0)$ is nonzero, our theoretical result on the double-descent phenomenon remains valid.

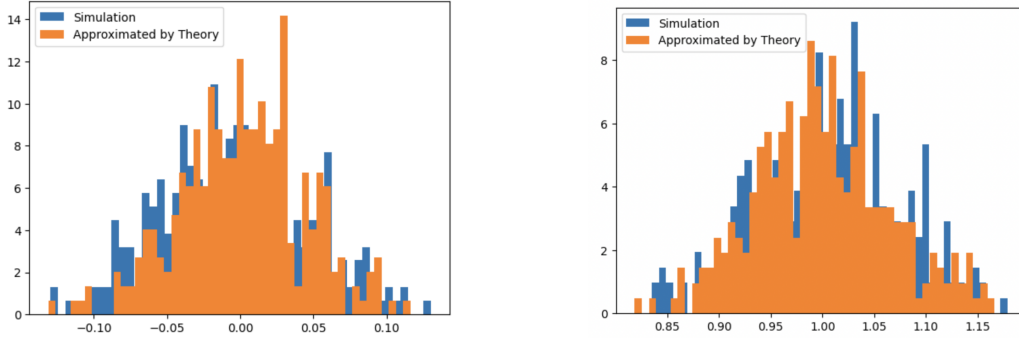


Figure 8.2.: Distributions of random width-dependent NNGP kernel values. X-axis are indexed by the kernel value and Y-axis are indexed by the number of samples. On the left, the simulated kernel value is obtained by sampling from the off-diagonal elements of the kernel random matrix of a two-layer NNGP with no non-linearity $K^d(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{d} \mathbf{x}_i^T \mathbf{W}^T \mathbf{W} \mathbf{x}_j$ under teacher distribution $N(0, \frac{1}{d_0} I_{d_0})$, and the theoretical distribution is obtained by sampling from $\Phi(\mathbf{x}_i)^T \Lambda \Phi(\mathbf{x}_j)$ using the spectral distribution $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$ and sampling from independent Gaussian entries in lieu of the eigenfunctions (SUA). On the right, the simulated kernel value is obtained by the same procedure for on-diagonal elements $K^d(\mathbf{x}_i, \mathbf{x}_i)$.

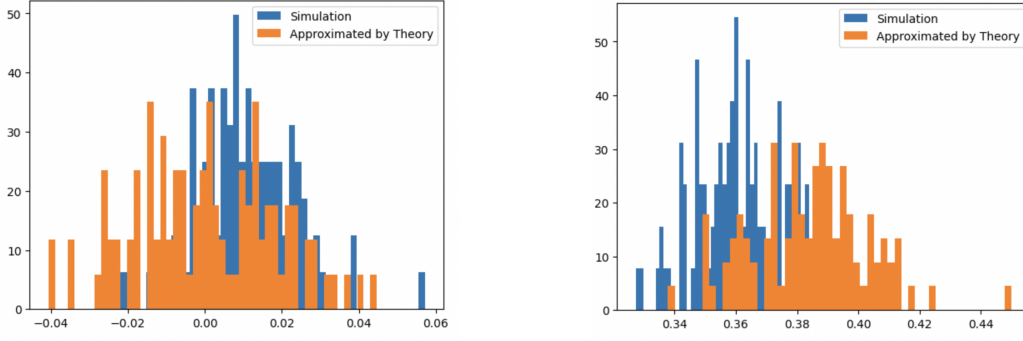


Figure 8.3.: Distributions of random width-dependent NNGP kernel values. X-axis are indexed by the kernel value and Y-axis are indexed by the number of samples. On the left, the simulated kernel value is obtained by sampling from the off-diagonal elements of the kernel random matrix of a two-layer NNGP with $\sigma = \text{ReLU}$ nonlinearity $K^{\sigma,d}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{d} \sigma(\mathbf{W}\mathbf{x}_i)^T \sigma(\mathbf{W}\mathbf{x}_j)$ on a subset of MNIST, and the theoretical distribution is obtained by sampling from $\Phi(\mathbf{x}_i)^T \Lambda \Phi(\mathbf{x}_j)$ using the spectral distribution of the actual NNGP kernel $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$ and sampling from independent Gaussian entries in lieu of the eigenfunctions (SUA). On the right, the simulated kernel value is obtained by the same procedure for on-diagonal elements $K^{\sigma,d}(\mathbf{x}_i, \mathbf{x}_i)$.

8.3. Concluding remarks of the chapter

A number of recent works have also provided analytical expressions for the generalisation error of high-dimensional kernel regressions (Canatar et al. 2021; Bordelon et al. 2020; Jacot et al. 2020; Simon et al. 2023; Cui et al. 2022). In particular, (Jacot et al. 2020) and (Simon et al. 2023) rely on the spectral universality assumption, just as we do to estimate the coefficients in our formula. As pointed out by (Simon et al. 2023), other works take the spectral universality assumption implicitly via, for instance, the replica method (Bordelon et al. 2020; Canatar et al. 2021). Our computation of the generalisation error is thus similar to the works of (Jacot et al. 2020; Simon et al. 2023). Their results however hold for frozen kernels and the dependence with the width is not studied.

The double-descent behaviour in the learning curves of high-dimensional kernel regression (including the NNGP and neural tangent kernels as particular cases) has been described in (Canatar et al. 2021; Bordelon et al. 2020). Our work improves on this line of research by introducing the idea of width-dependent kernels, which is especially well-suited to the context of neural networks where double descent manifests as the network width tends to infinity. Recent studies of the double-descent phenomenon have focused on random features regressions in the case of shallow networks **benign**; (Gerbelot et al. 2020; Liao et al. 2020; Emami et al. 2020; Gerace et al. 2020; Adlam & Pennington 2020; Belkin et al. 2020; Chen & Schaeffer 2021; Bosch et al. 2022; D’Ascoli et al. 2020), or kernel regression with no dependence on the width (Liu et al. 2021; Mallinar et al. 2022).

In this chapter, we have exploited our Mercer’s spectrum perspective and our novel results in random matrix theory to offer a new perspective on the double descent

8. The Double-Descent Phenomenon

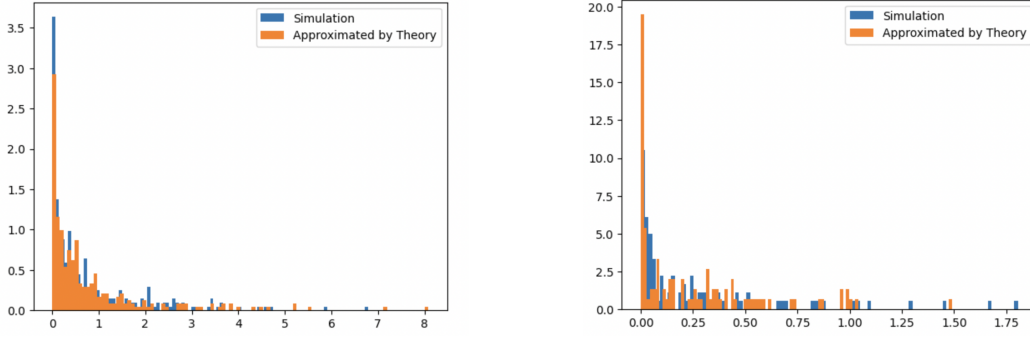


Figure 8.4.: Distributions of generalisation errors of (random) kernel regression. X-axis are indexed by the generalisation error and Y-axis are indexed by the number of samples. On the left, the simulated error is obtained by sampling from the prediction of kernel regression with the random (width-dependent) NNGP kernel $(\mathbf{x}, \mathbf{x}') \mapsto K^d(\mathbf{x}, \mathbf{x}')$ with no nonlinearity under teacher distribution $N(0, \frac{1}{d_0} I_{d_0})$, and the theoretical error is obtained by sampling from the approximate random kernel $(\mathbf{x}, \mathbf{x}') \mapsto \Phi(\mathbf{x})^T \Lambda \Phi(\mathbf{x}')$ using the spectral distribution $\rho_{MP}^\alpha \boxtimes \rho_{MP}^{\alpha_0}$ and sampling from independent Gaussian entries in lieu of the eigenfunctions (SUA). On the right, the simulated error is obtained by sampling from the prediction of kernel regression with the random (width-dependent) NNGP kernel $(\mathbf{x}, \mathbf{x}') \mapsto K^{\sigma, d}(\mathbf{x}, \mathbf{x}')$ with $\sigma = \text{ReLU}$ nonlinearity on a subset of MNIST, and the theoretical error is obtained by sampling from the approximate random kernel $(\mathbf{x}, \mathbf{x}') \mapsto \Phi(\mathbf{x})^T \Lambda \Phi(\mathbf{x}')$ using the spectral distribution of the actual NNGP kernel $\rho_{MP}^\alpha \boxtimes \mu_{\alpha_0}^\sigma$ and sampling from independent Gaussian entries in lieu of the eigenfunctions (SUA).

phenomenon in FCNs through the lens of Gaussian process kernels. We have derived analytical expressions for the generalisation error under teacher-student scenarios, which are applicable to networks of arbitrary depth and a large family of nonlinearities. This analysis allows us to predict the double descent behaviour as the width of the last layer changes relatively to the number of examples, and understand it as simply arising from the discrepancy between the spectrum of width-dependent random kernel (corresponding to the empirical covariance matrix of the features), and that of the width-independent NNGP kernel. Finally, we hope that the tools we have developed will motivate further research on the properties of the generalisation error of neural networks.

9. Infinite BNNs through Kernel Theory

Background. Bayesian Neural Networks (BNNs) are a variant of neural networks that incorporate Bayesian inference techniques to mitigate overfitting, enable learning from small datasets, and capture uncertainty in predictions (Neal 1996; Gal 2016). In a BNN, prior probability distributions are specified for weights and biases. During training, the posterior distribution, which represents the updated knowledge about the parameters after observing the data, is updated using Bayes’ rule. A trained BNN can be interpreted as an infinite ensemble of neural networks where each individual contribution in the ensemble is weighted by the posterior probability of its parameters given the training data. Although computing the posterior distribution is intractable and difficult to approximate, BNNs have gained significant traction with the development of effective estimation techniques (Gal 2016; Blei et al. 2017). BNNs demonstrate generalisation performance on par with deep neural networks trained using gradient descent (Lee et al. 2020; Magris & Iosifidis 2023). BNNs also showcase improved sensitivity to out-of-distribution examples (Gal 2016) and the ability to estimate uncertainty.

In an effort to analyse the generalisation properties of BNNs, researchers study idealised views of fully-connected neural architectures. The NNGP limit is a prime example of this, as it can be seen as the infinite-width limit of a BNN prior. This perspective based on kernel and GP theory has inspired formalisms that mimic different aspects of the behavior of BNNs in the infinite-width limit (Aitchison et al. 2021), including representation learning (Yang et al. 2023).

In addition to the NNGP limit, BNNs have also been studied under the linear-width limit (also referred to as *thermodynamic limit* or *proportional limit* in this literature). By employing techniques from statistical mechanics, such as saddle point approximations (Seung et al. 1992; Li & Sompolinsky 2021), the replica method (Barbier et al. 2018; Canatar et al. 2021), and random matrix theory, researchers have studied the mean and variance of the output generated by trained BNNs in this setting. A recent theoretical work (Cui et al. 2023) has derived the predictor learned by non-linear BNNs in the case of Gaussian data. More recently, *sublinear-width regimes*, where the width (or the input dimension) is small compared to the number of data points (Maillard et al. 2024), and related scalings (Meegen & Sompolinsky 2024) have been studied, and the emergence of strong feature learning has been demonstrated in these scenarios.

One of the most prominent results in this literature is the *renormalisation theory* (Li & Sompolinsky 2021) of linear BNNs (i.e., those without non-linear activations) in the linear-width regime, which establishes that the mean predictor and the predictor variance of the BNN coincide with that of Bayesian linear regression, but surprisingly the variance must be renormalised by a factor dependent on the training data and problem dimensions. Subsequent developments have provided more detailed analysis on the linear setting including non-asymptotic results (Hanin & Zlokapa 2023), and comparison with deep random feature models (Zavatone-Veth et al. 2022). It remained an open question, however, whether the insights from the renormalisation theory

for linear BNNs can be extended to non-linear networks, as suggested by empirical evidence (Li & Sompolinsky 2021; Pacelli et al. 2023), and how the theory should be adapted to sublinear-width regimes, where discrepancies with empirical results have been observed (Li & Sompolinsky 2021).

Contributions. In this chapter, we establish new connections between the kernel-theoretic perspective associated with the NNGP limit and the statistical mechanics viewpoint associated with the linear-width and sublinear-width limits, and contribute new insights to the generalisation properties of BNNs. First, we demonstrate that training a (non-linear) BNN in the linear-width and sublinear-width limits result in a predictor with identical mean and variance to that of GP regression with the modified NNGP kernel. Second, using the observation that the Mercer spectrum of this kernel is known in the linear-width regime (Section 7.3), we prove necessary and sufficient conditions (on the data and the architecture) for the application of renormalisation theory to non-linear BNNs in the linear-width limit. These conditions also provide a criterion for determining the applicability of the SUA from kernel theory in the context of BNNs. Third, we present initial findings on our sublinear-width regime (Assumption 7.5). In particular, we provide a novel mechanism for estimating the mean and variance of the predictions of non-linear BNNs in this setting, for which renormalisation theory is not applicable.

9.1. BNNs as Modified GP Regression

In this following we first exploit the Mercer’s spectrum of the modified NNGP kernel to extend the renormalisation theory to encompass nonlinear networks in the linear-width regime. Then, we outline the fundamental arguments supporting the expansion of this theory to the sublinear-width regime.

9.1.1. Training BNNs with the Modified NNGP Kernel

We can now study the predictor statistics of trained BNNs in the linear-width limit and the sublinear-width limit. In particular, the following theorem provides integral formulae to estimate, under the SUA, the first and second moments of the trained BNN using only the limiting spectral measure. In this section, the results hold indistinctively of the linear-width or the sublinear-width limit, so to simplify notations, we will note ρ for both ρ^{α, α_0} and ρ^γ and K_Θ for both $K_\Theta^{\alpha, \alpha_0}$ and K_Θ^γ .

Theorem 9.1. *Assume that Assumption 7.4 or Assumption 7.5 holds. Let ρ be the nonrandom spectral measure characterising the modified NNGP kernel K_Θ , and assume that the SUA holds. The mean $\langle \mathcal{N} \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y})$ and variance $\langle (\delta \mathcal{N})^2 \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y})$ of the predictor associated to a BNN*

9. Infinite BNNs through Kernel Theory

with training data $(\mathbf{X}, \mathbf{y})$ is given by expressions (9.1) and (9.2) respectively:¹

$$\langle \mathcal{N} \rangle = \int \left(\Phi^{*T} \Lambda \Phi^T \Phi^{T\dagger} \Lambda^{-1} \Phi^\dagger \mathbf{y} \right) \cdot \frac{p(\mathbf{y}, \Phi | \Lambda, \mathbf{X})}{p(\mathbf{y} | \mathbf{X})} d\rho(\Lambda) \mathcal{D}\Phi \mathcal{D}\Phi^* \quad (9.1)$$

$$\langle (\delta \mathcal{N})^2 \rangle = \int \left(\Phi^{*T} \Lambda \Phi^* - \Phi^{*T} \Lambda \Phi^T \Phi^{T\dagger} \Lambda^{-1} \Phi^\dagger \Phi \Lambda \Phi^* \right) \cdot \frac{p(\mathbf{y}, \Phi | \Lambda, \mathbf{X})}{p(\mathbf{y} | \mathbf{X})} d\rho(\Lambda) \mathcal{D}\Phi \mathcal{D}\Phi^* \quad (9.2)$$

where $\mathcal{D}\Phi$ is a standard Gaussian matrix measure, $\Phi_{i,j} \sim_{\text{iid}} N(\mu_{K_\Theta}, \sigma_{K_\Theta}^2)$ obtained from the SUA; the likelihood is given by $p(\mathbf{y}, \Phi | \Lambda, \mathbf{X}) \sim N(\Phi^T \mathbf{y} | \mathbf{0}, \Lambda)$; the marginal likelihood is given by $p(\mathbf{y} | \mathbf{X}) = \int p(\mathbf{y}, \Phi | \Lambda, \mathbf{X}) d\rho(\Lambda) \mathcal{D}\Phi$.

Proof. We calculate the conditional expectation $\langle \mathcal{N} \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta)$ and variance $\langle (\delta \mathcal{N})^2 \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta)$ of the predictor by marginalising over the readout weights $\mathbf{W}^L$:

$$\langle \mathcal{N} \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta) = \int \mathbf{W}^{LT} \phi(\Theta, \mathbf{X}) p(\mathbf{W}^L | \mathbf{X}, \mathbf{y}, \Theta) d\mathbf{W}^L$$

$$\langle (\delta \mathcal{N})^2 \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta) = \int \left[\mathbf{W}^{LT} \phi(\Theta, \mathbf{X}) \right]^2 p(\mathbf{W}^L | \mathbf{X}, \mathbf{y}, \Theta) d\mathbf{W}^L - [\langle \mathcal{N} \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta)]^2$$

where $p(\mathbf{W}^L | \mathbf{X}, \mathbf{y}, \Theta)$ can be expressed by Bayes rule using Gaussian likelihoods. The result can be expressed analytically and yields the same prediction as GP regression with prior $\mathcal{GP}(0, K_\Theta^{d,d_0})$:

$$\langle \mathcal{N} \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta) = [\mathbf{k}_\Theta^{n,d,d_0}(\mathbf{x}^*, \mathbf{X})]^T [\mathbf{K}_\Theta^{n,d,d_0}(\mathbf{X}, \mathbf{X})]^{-1} \mathbf{y}$$

$$\langle (\delta \mathcal{N})^2 \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}, \Theta) = K_\Theta^{n,d,d_0}(\mathbf{x}^*, \mathbf{x}^*) - [\mathbf{k}_\Theta^{n,d,d_0}(\mathbf{x}^*, \mathbf{X})]^T [\mathbf{K}_\Theta^{n,d,d_0}(\mathbf{X}, \mathbf{X})]^{-1} \mathbf{k}_\Theta^{n,d,d_0}(\mathbf{x}^*, \mathbf{X}).$$

To marginalise over $\Theta \sim p(\Theta | \mathbf{X}, \mathbf{y})$, we perform the change of variables $\Theta \mapsto (\Phi^*, \Phi, \Lambda)$, relying on the fact that all quantities of interest involving Θ can be expressed in the limit solely using eigenvalues and eigenfunctions, namely $\mathbf{K}_\Theta(\mathbf{X}, \mathbf{X}) = \Phi \Lambda \Phi^T$, $\mathbf{k}_\Theta(\mathbf{x}^*, \mathbf{X}) = \Phi \Lambda \Phi^*$, and $K_\Theta(\mathbf{x}^*, \mathbf{x}^*) = \Phi^{*T} \Lambda \Phi^*$. Since $\Phi \in \mathbb{R}^{n \times M}$ has orthogonal rows, $\Phi^\dagger = \Phi^T (\Phi \Phi^T)^{-1}$, and $\Phi^{T\dagger} = (\Phi \Phi^T)^{-1} \Phi$. This allows us to express the mean and variance of the predictor as follows:

$$\langle \mathcal{N} \rangle = \int \left(\Phi^{*T} \Lambda \Phi^T \Phi^{T\dagger} \Lambda^{-1} \Phi^\dagger \mathbf{y} \right) \cdot p(\Lambda, \Phi, \Phi^* | \mathbf{X}, \mathbf{x}^*) \cdot \frac{p(\mathbf{y} | \Lambda, \Phi, \Phi^*, \mathbf{X}, \mathbf{x}^*)}{p(\mathbf{y} | \mathbf{X})} d\Lambda d\Phi d\Phi^* \quad (9.3)$$

¹Here, $\mathbf{X}$ is the infinite matrix representing the linear-width or sublinear-width limits of the training data. To be completely rigorous, we should write $\langle \mathcal{N} \rangle(\mathbf{x}^*, \mathbf{X}, \mathbf{y}) = \lim_{n,d,d_0 \rightarrow \infty} \langle \mathcal{N}_{n,d,d_0} \rangle(\mathbf{x}_{d_0}^*, \mathbf{X}_{n,d_0}, \mathbf{y}_n)$, but by slight abuse of notation the same notation is used for both. In practice, one would use finite (but large) objects in calculations.

9. Infinite BNNs through Kernel Theory

$$\begin{aligned} \langle (\delta \mathcal{N})^2 \rangle &= \int \left(\Phi^{*T} \Lambda \Phi^* - \Phi^{*T} \Lambda \Phi^T \Phi^{T^\dagger} \Lambda^{-1} \Phi^\dagger \Phi \Lambda \Phi^* \right) \cdot \\ &\quad p(\Lambda, \Phi, \Phi^* | \mathbf{X}, \mathbf{x}^*) \frac{p(\mathbf{y} | \Lambda, \Phi, \Phi^*, \mathbf{X}, \mathbf{x}^*)}{p(\mathbf{y} | \mathbf{X})} d\Lambda d\Phi d\Phi^* \quad (9.4) \end{aligned}$$

Furthermore, it holds that $p(\Lambda, \Phi, \Phi^* | \mathbf{X}, \mathbf{x}^*) = p(\Lambda | \mathbf{X}, \mathbf{x}^*) p(\Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*)$ if $p(\Lambda | \mathbf{X}, \mathbf{x}^*) \neq 0$ and also $p(\mathbf{y} | \Lambda, \Phi, \Phi^*, \mathbf{X}, \mathbf{x}^*) = \frac{p(\mathbf{y}, \Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*)}{p(\Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*)}$ if $p(\Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*) \neq 0$, which yields:

$$\begin{aligned} \langle \mathcal{N} \rangle &= \int \left(\Phi^{*T} \Lambda \Phi^T \Phi^{T^\dagger} \Lambda^{-1} \Phi^\dagger \mathbf{y} \right) \cdot p(\Lambda | \mathbf{X}, \mathbf{x}^*) \frac{p(\mathbf{y}, \Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*)}{p(\mathbf{y} | \mathbf{X})} d\Lambda d\Phi d\Phi^* \\ \langle (\delta \mathcal{N})^2 \rangle &= \int \left(\Phi^{*T} \Lambda \Phi^* - \Phi^{*T} \Lambda \Phi^T \Phi^{T^\dagger} \Lambda^{-1} \Phi^\dagger \Phi \Lambda \Phi^* \right) \cdot \\ &\quad p(\Lambda | \mathbf{X}, \mathbf{x}^*) \frac{p(\mathbf{y}, \Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*)}{p(\mathbf{y} | \mathbf{X})} d\Lambda d\Phi d\Phi^* \end{aligned}$$

where the integral over Λ is restricted to segments where $p(\Lambda | \mathbf{X}, \mathbf{x}^*) \neq 0$ and the integrals over Φ and Φ^* are restricted to where $p(\Phi, \Phi^* | \Lambda, \mathbf{X}, \mathbf{x}^*) \neq 0$. We obtain equations (9.1) and (9.2) by replacing $d\Phi$ and $d\Phi^*$ by standard Gaussian matrix measures and the density of Λ by the spectral measure. $\square$

The integral forms (9.1) and (9.2) provide a new estimation of the predictor statistics of a trained BNN. While these expressions are exact only in the limit, we will present empirical evidence suggesting that they constitute a reasonable approximation. A practical challenge arises from the need to estimate the spectral distribution $\rho(\Lambda)$, which often involves diagonalising a kernel matrix. This process can be computationally intensive, especially for large training datasets; however, it is worth mentioning that in many applications of BNNs, where the training datasets are relatively small, this computational difficulty becomes less significant. In particular, in the linear-width regime, $\rho(\Lambda)$ is the Marchenko-Pastur map of the empirical spectral distribution of the NNGP kernel $\rho_{MP}^\alpha \boxtimes^L \rho_{\text{NNGP}}^{\alpha_0}$ which can be computed by diagonalising the NNGP kernel (Choi & Saul 2009) to estimate $\rho_{\text{NNGP}}^{\alpha_0}$ and solving numerically the Marchenko-Pastur fixed-point equation in the Stieltjes transform space (Marchenko & Pastur 1967; Fan & Wang 2020).

Theorem 9.1 and its proof also offer valuable new perspectives on the applicability of the SUA. In particular, in the last steps of the proof, the probability density of Φ, Φ^* no longer appears directly in the integral. For given $\Lambda, \mathbf{X}, \mathbf{x}^*$, if each orthogonal matrix Φ, Φ^* has a non-zero probability of occurring, the integral spans uniformly over the entire space of orthogonal matrices of size $n \times M$. This is useful because in the limit of infinite dimensions, this space coincides with that of Gaussian matrices with independent entries (independent infinite Gaussian vectors are orthogonal). Remarkably, this property precisely corresponds to the SUA in kernel theory (Karoui 2010; Jacot et al. 2020; Simon et al. 2023), which posits that, in terms of the generalisation error statistics in kernel regression, the eigenfunctions can be approximated by Gaussian matrices with independent entries, denoted $\Phi_{i,k} \sim N(\mu_K, \sigma_K^2)$. Note that this approximation is not the same as the Gaussian equivalence assumption (Schröder et al. 2023; Cui et al. 2023), which assumes Gaussianity of the predictor (here, the eigenfunctions

are Gaussian but the predictor is non-Gaussian). To the best of our knowledge, this marks a first connection between BNNs and the SUA from kernel theory (i.e. applied to Mercer’s eigenfunctions).

We can now reframe the question concerning the correctness of the SUA approximation as follows: given Λ and $\mathbf{X}$, is it the case that all orthogonal matrices Φ have non-zero probabilities (according to Θ) to satisfy $\mathbf{K}_\Theta(\mathbf{X}, \mathbf{X}) = \Phi \Lambda \Phi^T$? If this condition holds, the SUA is applicable and Gaussian eigenfunctions can be used for estimating (9.1) and (9.2). Since the prior is a Gaussian matrix, any matrix has a non-zero probability of occurrence, thus it suffices to show that for any orthogonal Φ , there exists a Θ such that $\mathbf{K}_\Theta(\mathbf{X}, \mathbf{X}) = \Phi \Lambda \Phi^T$. In particular, it is easy to show that the SUA always holds in the linear case: for any orthogonal Φ , there exists Θ such that $\mathbf{X}^T \Theta^T \Theta \mathbf{X} = \Phi \Lambda \Phi^T$. With a non-linearity, the problem is less obvious: is there a Θ such that $\phi(\Theta, \mathbf{X})^T \phi(\Theta, \mathbf{X}) = \Phi \Lambda \Phi^T$ for any orthogonal Φ ? In the next section, we show that, in the linear-width limit, this criterion about $\phi(\Theta, \cdot)$ and $\mathbf{X}$ is a necessary and sufficient assumption for the renormalisation theory to hold.

9.1.2. An Extended Renormalisation Theory

This section only concerns the linear-width regime. In this limit, we can explicitly derive the results of our integral estimators because the corresponding limiting nonrandom spectral measure is known (see Paragraph 7.3).

The renormalisation theory for linear BNNs establishes that, in the linear-width limit, the marginal likelihood $p(\mathbf{y}|\mathbf{X})$ follows the probability density of sampling $\mathbf{y}$ from a multivariate Gaussian with mean vector $\mathbf{0}$ and covariance matrix $u_0^L \mathbf{K}_0$, with $\mathbf{K}_0 = \frac{1}{d_0} \mathbf{X} \mathbf{X}^T$ and u_0 the renormalisation factor fulfilling the fixed-point equation $1 - u_0 = \alpha(1 - \frac{r_0}{u_0^L})$ with $r_0 = \frac{1}{n} \mathbf{y}^T \mathbf{K}_0^{-1} \mathbf{y}$. This result was obtained in [Li & Sompolinsky \(2021\)](#) by successively applying the saddle point method when integrating out the weights $\Theta, \mathbf{W}^L$.

The following theorem shows that this result generalises to BNNs with nonlinear activations if and only if the SUA is correct (i.e., it gives the correct estimate for the marginal likelihood). Here, we exploit the characterisation of the correctness of the SUA developed as a corollary of Theorem 9.1.

Theorem 9.2. *Assume that Assumption 7.4 holds. Let u_{NNGP} fulfil the fixed-point equation*

$$1 - u_{\text{NNGP}} = \alpha \left(1 - \frac{r_{\text{NNGP}}}{u_{\text{NNGP}}^L} \right) \quad (9.5)$$

with $r_{\text{NNGP}} = \frac{1}{n} \mathbf{y}^T \mathbf{K}_{\text{NNGP}}^{-1} \mathbf{y}$. The marginal likelihood for a nonlinear BNN verifies $p(\mathbf{y}|\mathbf{X}) \sim N(\mathbf{y}|\mathbf{0}, u_{\text{NNGP}}^L \mathbf{K}_{\text{NNGP}})$ if and only if, for given $\Lambda, \mathbf{X}$ and orthogonal Φ , there exists Θ such that $\phi(\Theta, \mathbf{X})^T \phi(\Theta, \mathbf{X}) = \Phi \Lambda \Phi^T$.

Proof. In the linear case, the true NNGP kernel is simply $K_{\text{NNGP}}(\mathbf{x}, \mathbf{x}') = \frac{1}{d_0} \mathbf{x}^T \mathbf{x}'$; hence, $\rho_{\text{NNGP}}^{\alpha_0}$ is the limiting spectral distribution of the kernel random matrix $\mathbf{K}_0$, which we denote ρ_0 . The renormalisation theory of linear networks thus implies that:

$$\int p(\mathbf{y}, \Phi | \Lambda, \mathbf{X}) d \left(\rho_{\text{MP}}^\alpha \boxtimes^L \rho_0 \right) (\Lambda) \mathcal{D}\Phi \sim N(\mathbf{y}|\mathbf{0}, u_0^L \mathbf{K}_0) \quad (9.6)$$

9. Infinite BNNs through Kernel Theory

This identity is exact in the linear-width limit and holds in general without assumption on $\mathbf{X}, \mathbf{y}$, as long as the integral $\mathcal{D}\Phi$ is uniform on the space of orthogonal matrices.

Assume the SUA holds in the nonlinear case. Thus, we can express the marginal likelihood as $p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}, \Phi|\Lambda, \mathbf{X}) d(\rho_{MP}^\alpha \boxtimes^L \rho_{NNGP}^{\alpha_0})(\Lambda) \mathcal{D}\Phi$. Furthermore, we can freely interchange the role of $\mathbf{K}_0$ and $\mathbf{K}_{NNGP}(\mathbf{X}, \mathbf{X})$ in (9.6). Indeed, it suffices to consider the linear case and a new training dataset $\tilde{\mathbf{X}}$ which exhibits the same covariance structure $\frac{1}{d_0} \tilde{\mathbf{X}} \tilde{\mathbf{X}}^T$ as that of $\mathbf{K}_{NNGP}(\mathbf{X}, \mathbf{X})$. As a result, a similar equation to (9.6) applies to nonlinear networks by replacing the linear kernel $(\mathbf{x}, \mathbf{x}') \mapsto \frac{1}{d_0} \mathbf{x}^T \mathbf{x}'$ with the true NNGP kernel in the equations, provided that the SUA holds:

$$\int p(\mathbf{y}, \Phi|\Lambda, \mathbf{X}) d(\rho_{MP}^\alpha \boxtimes^L \rho_{NNGP}^{\alpha_0})(\Lambda) \mathcal{D}\Phi \sim N(\mathbf{y}|\mathbf{0}, u_{NNGP}^L \mathbf{K}_{NNGP}) \quad (9.7)$$

Conversely, if the SUA does not hold, the integral with respect to Φ does not span the space of orthogonal matrices, the identity (9.7) is no longer exact (all integrands are strictly positive), nor is the renormalisation. Thus, the SUA is necessary and sufficient for the renormalisation to hold. $\square$

This result characterises the renormalisation theory in the nonlinear case and describes a continuous transition between an accurate and a poor approximation. Specifically, if the SUA significantly deviates (the feature map spans a small fraction of the space of orthogonal matrices) then the equivalence (9.7) also deviates substantially from the correct value. For example, in the spiked kernel case, which occurs for one step of feature learning (Dandi et al. 2024), the orthogonal matrices permissible for constructing the prior kernel is significantly constrained, thus we anticipate that the spectral universality assumption would fail in this scenario. Conversely, if the SUA is nearly accurate (meaning that the feature map encompasses a large portion of the space of orthogonal matrices) then (9.7) closely approximates the true marginal likelihood. Thanks to these insights, future research on BNNs can benefit from research advances on the accuracy of the SUA (Liu et al. 2021).

9.1.3. Application to the Sublinear-Width Regime

In this section, we consider the application of our integral estimators to the sublinear-width regime.

Assume that Assumption 7.5 holds. In this regime, the ratios $\alpha = \frac{n}{d}$ and $\alpha_0 = \frac{n}{d_0}$ from the linear-width regime tend to infinity and hence are no longer bounded. Here, the renormalisation theory breaks even in the linear case, because the random matrix $\mathbf{K}_{\Theta}^{n, d, d_0}(\mathbf{X}, \mathbf{X})$ becomes degenerate and its limiting spectral distribution is the Dirac distribution at 0 and (9.6) no longer holds. A mismatch with the predictions of the renormalisation theory has indeed been observed empirically for high values of α and α_0 (Li & Sompolinsky 2021), hence the need for a new theory.

Remarkably, our kernel-theoretic description of BNNs (Theorem 9.1) still holds as its validity relies only on the dot product of random feature maps $\phi(\Theta, \cdot)$ defining a random kernel (see Chapter 7 and in particular, proof of Theorem 7.6). This remains true for the sublinear-width regime (as well as for other regimes of interest). Additionally, zero eigenvalues in Mercer's decomposition can be disregarded since they do not

9. Infinite BNNs through Kernel Theory

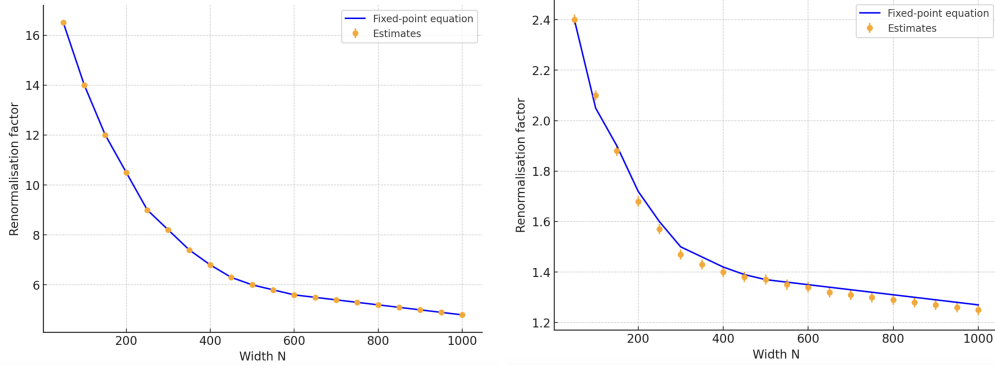


Figure 9.1.: Comparison with Li & Sompolinsky (2021) in the linear-width regime. On the left, the linear setting on our synthetic dataset with $d_0 = 500$ and $n = 200$. On the right, the nonlinear setting on the subset of MNIST. In both cases, the blue line is computed using the fixed-point (9.5), and the orange dots are the ratio between the result of our integral estimator (9.2) and the variance of Bayesian linear regression (respectively, NNGP regression) on the left (respectively, on the right). In the nonlinear case, we use a large width $\hat{d} = 10000$ to estimate the NNGP kernel matrix for ReLU.

contribute to the kernel evaluation. An alternative perspective is that, when calculating the first two moments, one takes into account the Moore-Penrose pseudo-inverse $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})^{\dagger}$ of the kernel random matrix. Consequently, only the contributions from the *strictly positive support* of the limiting spectral distribution are considered (see Theorem 7.6). As a result, our integral estimators for the mean and variance of the predictor remain applicable under the SUA. The only missing element in the argument is whether $p(\Lambda|\mathbf{X}, \mathbf{x}^*)$, which is now the strictly positive support of the limiting spectral distribution of $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ (rescaled to integrate to 1), also converges to a nonrandom spectral measure (in order to apply Theorem 7.6). Thus, precisely characterising the asymptotic behavior of this Mercer’s random spectral measure and assessing the SUA in this regime is an interesting avenue for further research. Note however that this approach only concerns the predictor statistics and is not derived in weight space, thus one limitation of the approach that we anticipate is that it might be difficult to characterize (strong) feature learning from this standpoint.

Although we do not have an analytical formula for the limiting spectral measure, it is still possible to numerically compute the strictly positive support of the random matrix $\mathbf{K}_{\Theta}^{n,d,d_0}(\mathbf{X}, \mathbf{X})$ and use our integral forms (9.3) and (9.4) to estimate the predictor of a trained BNNs in this regime.

9.2. Experiments

We consider a synthetic dataset generated by a multivariate Gaussian $\mathbf{x} \sim N(0, \frac{1}{d_0} \mathbf{I}_{d_0})$ to which we apply a linear teacher and noise $y = \beta^T \mathbf{x} + \epsilon$ with $\epsilon \sim N(0, \sigma_{\epsilon}^2)$ and $\beta = (1, \dots, 1)^T$. We also consider a subset of MNIST restricted to classes "0" and "1" of size $n = 105$ and with $d_0 = 784$ pixels per image.

9. Infinite BNNs through Kernel Theory

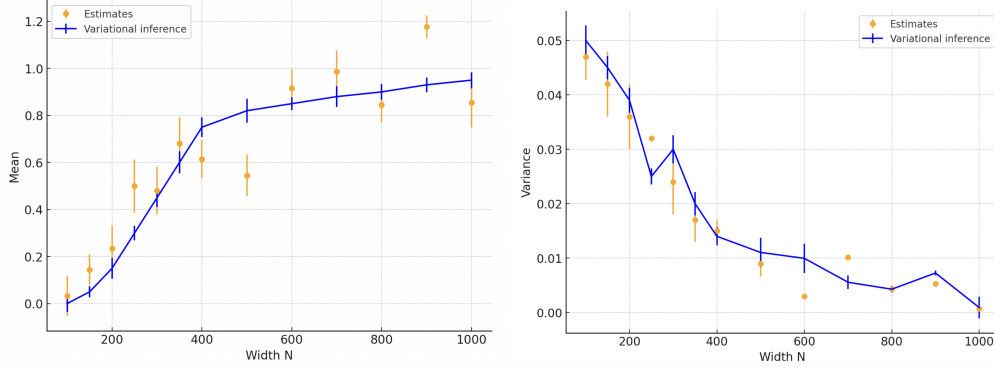


Figure 9.2.: Sublinear-width regime. Mean and variance of the predictor against the width d of the single ReLU hidden-layer on our synthetic dataset with $n = 200$ and $d_0 = 40$. In both cases, the blue line is computed using the probabilistic predictions of a BNN trained with variational inference on the synthetic data, and the orange dots correspond to our integral estimates.

Our first experiment verifies that our estimators coincide with the predictions of the renormalisation theory in the linear-width limit both for a single hidden-layer network with ReLU activations and a linear network with a hidden layer. We computed the renormalisation factors using the fixed-point (9.5) and used (9.1) and (9.2) to estimate the mean and the variance of the predictor in our approach. To compute (9.1) and (9.2) we first computed the Marchenko-Pastur maps of the empirical spectral distributions (of the NNGP kernel) by solving numerically the Marchenko-Pastur fixed-point equation in the Stieltjes transform space (Marchenko & Pastur 1967; Fan & Wang 2020); then, we relied on the SUA to estimate the integral forms. In a second experiment, we simulated the sublinear-width regime $n \propto d \cdot d_0$ (for which the renormalisation theory breaks, see Figure 14 in Li & Sompolinsky (2021)) using a small value of d_0 (thus making α_0 high). We compared our estimators for the regime as described in the previous section with the predictions of BNNs trained with variational inference using the library Pyro (Bingham et al. 2019). For the spectral distribution, we computed the strictly positive support of the empirical spectral distributions by sampling and diagonalising the empirical kernel matrices several times and shuffling the eigenvalues; we continued to use the SUA for eigenfunctions.

As shown in Figure 9.1, our estimates align with the renormalisation theory in the linear-width limit. As shown in Figure 9.2, for a regime where the renormalisation theory is inaccurate, our estimators provide reasonable matches to the actual predictions. These results suggest that our estimates are better suited to regimes where key assumptions of the renormalisation theory do not hold. Sources of discrepancies include: the fact that we used finite values of n, d, d_0 (whereas the theory is only exact in the limit); the SUA may not be fully accurate in this configuration; the limiting spectral measure may not be nonrandom.

9.3. Concluding remarks of the chapter

In this chapter, we have explored bridges between BNNs trained under interesting idealised limits and kernel theory, which enable an extension of the renormalisation

9. Infinite BNNs through Kernel Theory

theory to non-linear networks. Our theory offers a new way to estimate the prediction of BNNs with better accuracy in the sublinear-width regime. Finally, we hope that the theory developed here will motivate further research on the application of existing kernel-theoretic results in the context of BNNs.

10. Adversarial Attacks and Near-Zero Eigenvalues

Background. The foundational paper introducing adversarial examples (Szegedy et al. 2014) characterised them as “intriguing properties” of neural networks and raised a compelling question: how can neural networks exhibit strong generalisation performance on test examples drawn from a data distribution, while simultaneously being susceptible to adversarial examples? An interesting hypothesis, which we recapitulate next, was already outlined in the paper’s concluding remarks.

“A possible explanation is that the set of adversarial negatives is of extremely low probability, and thus is never (or rarely) observed in the test set, yet it is dense (much like the rational numbers), and so it is found near virtually every test case.”

Despite the substantial attention dedicated by researchers to explaining the prevalence of adversarial examples, and despite progress in identifying new attack variants and developing defensive approaches, robustness measures (Yu et al. 2019; Carlini et al. 2019; Wang et al. 2023b), and theoretical guarantees based on such measures (Bhagoji et al. 2019; Shafahi et al. 2019), the fundamental nature of adversarial examples continues to elude complete understanding (Guo et al. 2018; Ilyas et al. 2019; Madry et al. 2018; Qin et al. 2019).

Several explanations have been proposed for the pervasiveness of adversarial attacks in neural networks. The *linearity hypothesis* (Goodfellow et al. 2015) posits that, because of the (local) linear nature of trained neural networks, small changes to each component of a high-dimensional vector amount to a large change in the network’s output. The universality of ℓ_2 -adversarial attacks on ReLU networks with random weights (Daniely & Shacham 2020) provides a strong argument supporting this hypothesis, especially considering the locally linear characteristics of ReLU networks. The linearity hypothesis has motivated a series of research endeavors analysing the topological characteristics of decision boundaries, with the aim to elucidate the nature of adversarial examples and develop techniques for enhancing adversarial robustness. Tanay & Griffin (2016) argues that the linearity hypothesis is not fully satisfactory, and rather attributes adversarial examples to the distance of the sampling subspace to the decision boundary. In other related works (Simon-Gabriel et al. 2019), the susceptibility to adversarial attacks escalates with the increase in input dimensionality. Deficiencies in the topology become more pronounced in higher dimensions, a phenomenon exacerbated by the curse of dimensionality. In some instances, particularly for synthetic data distributions with sufficiently high dimensions, adversarial attacks become nearly unavoidable, as noted in Shafahi et al. (2019). Moreover, data sparsity in relation to the input space heightens vulnerability to adversarial attacks, as discussed in Paknezhad et al. (2022) and Weitzner & Giryas (2023). The dimensionality of the parameter space also plays a significant role in this context, with parameter redundancy (Paknezhad et al. 2022) and a large ℓ_1 -norm of the parameters (Guo et al. 2018) representing situations associated with increased vulnerability.

Another line of research identifies features as the key object driving the occurrence of adversarial examples. In Ilyas et al. (2019), it was shown that *non-robust features* (those resulting from spurious data correlations that are nevertheless highly predictive) are responsible for the presence of adversarial examples. Similarly, Wang et al. (2017a) established that the existence of an unnecessary feature introduced to replicate the true underlying target function renders a system vulnerable to adversarial attacks. This perspective is further supported by the observation that saliency methods tend to emphasise class-discriminative features which can be exploited by attackers (Gu & Tresp 2019).

Contributions. In this chapter, we provide novel theoretical results that further substantiate the original hypothesis by Szegedy et al. (2014). Our argument relies on the Mercer spectrum of the empirical kernel of finite neural networks (Chapter 7).

We adopt the perspective that the introduction of an adversarial example can be regarded as a modification of the input data distribution, expanded to incorporate the adversarial example. By Mercer’s theorem, even when maintaining the same empirical kernel associated to a trained neural network, this process results in a new set of Mercer eigenvalues and eigenfunctions, since the data distribution used to compute the Mercer’s decomposition is changed. Our first technical contribution is to characterise adversarial examples in interpolating regression settings as those producing near-zero Mercer eigenvalues in the updated decomposition of the neural network’s empirical kernel. This characterisation then allows us to show that adversarial examples have measure zero in the limit where they become infinitesimally close to any test example sampled from the original data distribution. Our results provide, in a restricted setting, a rigorous mathematical explanation for what appears to be a paradoxical empirical observation: while neural networks demonstrate strong generalisation to novel test examples, they are also susceptible to adversarial attacks. Indeed, it follows from our results that adversarial examples are exceedingly unlikely to occur in the test set, thereby providing compelling support for the hypothesis by Szegedy et al. (2014). Our theory could also open the way to new approaches in designing defense mechanisms and detection methods for adversarial attacks (Tramer 2022).

We conducted experiments on FCNs and on CNNs on a subsets of MNIST and subsets of CIFAR10, by achieving interpolation on the training set and perfect generalisation performance on a subset of the test sets. We then used several widely-known attack methods; Projected Gradient Descent (PGD) (Madry et al. 2018), Fast Gradient Sign Method (FGSM) (Goodfellow et al. 2015), DeepFool (Moosavi-Dezfooli et al. 2016) and Fast Adaptive Boundary (FAB) (Croce & Hein 2020); on the FCNs and the CNNs to generate adversarial examples, and used the numerical method described in Baker (1977) and Rasmussen & Williams (2006) to compute Mercer’s decomposition on the empirical kernel associated with both the original empirical data distribution and our updated empirical data distributions, which incorporate the adversarial examples. Our numerical experiments align with our theory, demonstrating that adversarial examples induce a shift in the distribution of Mercer eigenvalues towards zero, and corroborate that our theory is indeed architecture- and data-agnostic.

Position. We argue that our approach is positioned at the intersection of the streams of research on linearity hypothesis and non-robust features. By taking Mercer’s eigenvalues into account, our theory operates in the high-dimensional embedding space where the predictions follow a linear pattern. Additionally, shifting towards zero eigenvalues can be understood as exploiting directions of non-robust features. Indeed, our approach is inspired by the double-descent phenomenon (Chapter 8), in which the spectrum of the empirical kernel also shifts towards zero at the divergence in terms of generalisation error. Intuitively, near-zero eigenvalues can be understood as stochastic cancellations giving rise to the emergence of spurious directions within the feature space, causing them to overfit.

A complementary perspective connects our spectral analysis to smoothness-based approaches to robustness, such as randomised smoothing. In particular, [Cohen et al. \(2019\)](#) shows that robustness guarantees can be obtained by enforcing stability of predictions under input perturbations via randomised smoothing, effectively controlling the Lipschitzness of the model. With our perspective, randomised smoothing can be interpreted as regularising the spectrum of the empirical kernel: it suppresses near-zero Mercer eigenvalues arising from low-probability stochastic cancellations, thus limiting the presence of highly sensitive, non-robust directions in feature space. Conversely, the emergence of small eigenvalues corresponds to directions along which the feature map is locally non-smooth and exhibits high local variability, making the model vulnerable to adversarial perturbations.

A final perspective, which bears loose connections with our approach, is based on information geometry ([Zhao et al. 2019](#); [Naddeo et al. 2022](#)). This perspective centers on the use of the *Fisher Information Matrix* which quantifies, for any pair of data points, the correlation between parameter gradients of the log likelihood of the data. A technique involving the iterative elimination of the dominant eigenvalue direction in the Fisher Information Matrix leads to the generation of adversarial examples. This process appears to yield configurations where variations in network parameters exhibit strong linear dependencies with respect to the data. These configurations may manifest whenever certain features become perfectly correlated within the dataset and this, in turn, results in the empirical kernel having Mercer eigenvalues that approach zero.

Chapter-specific assumptions. In this chapter, we assume a more general definition of neural networks. A neural network with L layers is a tuple $\mathcal{N} = \langle \{\Theta^\ell\}_{1 \leq \ell \leq L}, \{\sigma^\ell\}_{1 \leq \ell \leq L} \rangle$. Each layer $\ell \in \{1, \dots, L\}$ consists of parameters Θ^ℓ (matrices and vectors), and a non-linear operator σ^ℓ . On input $\mathbf{x} \in \mathbb{R}^{d_0}$, $\mathcal{N}$ sets $\mathbf{x}^0 := \mathbf{x}$ and then computes, for each layer $1 \leq \ell \leq L$, a sequence of vector representations $\mathbf{x}^\ell = \sigma^\ell(\Theta^\ell, \mathbf{x}^{\ell-1})$. The output $\mathcal{N}(\mathbf{x})$ is given by $\mathbf{x}^L$. This description encompasses various common architectures including FCNs, CNNs and their variants.

For the last layer, we continue to note d the dimension of $\mathbf{x}^{L-1}$ and to assume Θ^L is a column matrix $\mathbf{W}^L \in \mathbb{R}^d$ and $\sigma^L(\Theta^L, \mathbf{x}^{L-1}) = (\mathbf{W}^L)^T \mathbf{x}^{L-1}$ (readout weights).

We assume that $\mathcal{N}$ is Lipschitz continuous and that each component of the feature map is Lipschitz continuous with respect to inputs $\mathbf{x}$, with a global Lipschitz constant noted C . This assumption holds for most commonly-used architectures including FCNs, CNNs and their variants, since the activation functions ReLU, sigmoid, tanh, as well as pooling and other commonly-used nonlinear operators are all Lipschitz continuous

(Virmaux & Scaman 2018).

Furthermore, we consider a training set $(\mathbf{X}, \mathbf{y})$ with $n > d$ examples sampled i.i.d. from an unknown Borel distribution p with compact support on $\mathbb{R}^{d_0} \times \mathbb{R}$. The evaluation $\phi_{\mathcal{N}}(\mathbf{X}) = (\phi_{\mathcal{N}}(\mathbf{x}_1), \dots, \phi_{\mathcal{N}}(\mathbf{x}_n))^T \in \mathbb{R}^{d \times n}$ of the empirical feature map on the training set induces an *empirical feature covariance matrix* $\mathbf{K}_{\mathcal{N}}(\mathbf{X}, \mathbf{X}) \in \mathbb{R}^{n \times n}$, where element (i, j) for $1 \leq i \leq j \leq n$ is given by $K_{\mathcal{N}}(\mathbf{x}_i, \mathbf{x}_j)$. Finally, augmenting the training data $\mathbf{X} \sim_{\text{i.i.d.}} p$ with an additional example $\mathbf{x}'$ (which is not required to be sampled from p) induces an *updated data distribution* $p_{\mathbf{x}'}$ defined as $p_{\mathbf{x}'} = \frac{n}{n+1}p + \frac{1}{n+1}\delta_{\mathbf{x}'}$ with $\delta_{\mathbf{x}'}$ the Dirac measure.

Adversarial examples. Consider $\mathcal{N}$ trained on the regression training set $(\mathbf{X}, \mathbf{y})$ to exact interpolation (zero error). Let $M_{\text{adv}} > M_{\text{nat}} > 0$ and $\epsilon > 0$. A vector $\mathbf{x}'$ is an $(\epsilon, M_{\text{adv}}, M_{\text{nat}})$ -adversarial example for $\mathcal{N}$ and p if there exists an example $(\mathbf{x}^*, y^*) \sim p$ such that $\|\mathbf{x}^* - \mathbf{x}'\| \leq \epsilon$ and $|y^* - \mathcal{N}(\mathbf{x}^*)| \leq M_{\text{nat}}$, but $|y^* - \mathcal{N}(\mathbf{x}')| \geq M_{\text{adv}}$.

The prediction discrepancy factors $M_{\text{adv}}, M_{\text{nat}}$ are introduced to adjust the standard definition of adversarial example for classification to the regression setting, where there is no a-priori notion of what it means for the adversarial example to change the prediction (in classification, changing the prediction means predicting a different class). The prediction discrepancy factors $M_{\text{adv}}, M_{\text{nat}}$ allows practitioners to quantify when a modification of the output is significant for the regression task at hand, in particular they provide criteria for when an error is adversarial vs when an error is natural. To simplify the presentation, we fix arbitrary such $M_{\text{adv}}, M_{\text{nat}}$ and speak from now onwards of ϵ -adversarial examples. The first condition in the definition requires that the adversarial example $\mathbf{x}'$ is close in norm to some test example for which $\mathcal{N}$ generalises to small error; the second requirement ensures that the corresponding error w.r.t. $\mathbf{x}'$ is large.

10.1. Adversarial Attacks Shift Mercer's Spectrum Towards Zero

In this section, we show that adversarial examples can be characterised, in interpolation regression settings, as those that shift the Mercer's spectrum of the empirical kernel corresponding to a trained neural network to yield near-zero eigenvalues with sufficient probability.

Assume that $\mathcal{N}$, which has been trained (technically, to zero error) on a regression dataset $(\mathbf{X}, \mathbf{y})$ drawn from an unknown Borel data distribution p . By Mercer's theorem, we have the existence of a unique collection of Mercer's eigenvalues $\lambda_i^{K_{\mathcal{N}}, p}$ and eigenfunctions $\phi_i^{K_{\mathcal{N}}, p}$ associated to the empirical kernel $K_{\mathcal{N}}$ and the data distribution p underlying the training data. We denote $\rho^{K, p}(\lambda)$ the density of Mercer's eigenvalues, the measure with support on $\mathbb{R}_{\geq 0}$ defined by $\lim_{M \rightarrow \infty} \frac{1}{M} \sum_{i=1}^M \delta_{\lambda_i^{K, p}}(\lambda)$ (convergence in distribution), where $\delta_{\lambda_i^{K, p}}$ is the Dirac measure.

Now, suppose that we sample a new example $(\mathbf{x}^*, y^*)$ using the true data distribution p for which the $\mathcal{N}$ exhibits reasonable generalisation performance (technically, the error is smaller than the given $M_{\text{nat}} > 0$), and consider any data point $\mathbf{x}'$ in the vicinity of $\mathbf{x}^*$. After augmenting the training set with the new example $\mathbf{x}'$, we reconsider the *updated*

10. Adversarial Attacks and Near-Zero Eigenvalues

Mercer spectrum $\lambda_i^{K_{\mathcal{N}}, p_{\mathbf{x}'}}$ for the same kernel $K_{\mathcal{N}}$ and the *updated* data distribution $p_{\mathbf{x}'}$. In the next theorem, we show that $\mathbf{x}'$ is adversarial if and only if the eigenvalues in the updated Mercer spectrum exhibit sufficient density near zero. Consequently, in this configuration, the generation of adversarial attacks, using any known technique, can be conceptualised as a progression towards the eigenvalue space's zero point within the empirical kernel.

The main insight behind the proof is the observation that the local Lipschitz constant of $\mathcal{N}$ around $\mathbf{x}^*$, which quantifies the growth rate between the new data point $\mathbf{x}'$ and the true example $\mathbf{x}^*$ can be written as a sum of terms involving Mercer eigenvalues and eigenfunctions in the updated spectrum. In this decomposition, the value of terms depending on the eigenfunctions can be bounded as $\epsilon \rightarrow 0$, and hence the only way to have an adversarial local Lipschitz constant for small values of $\epsilon > 0$ is for the terms involving the eigenvalues to implement a diverging function of $\epsilon \rightarrow 0$.

Theorem 10.1. *Let $\mathcal{N}$ be a Lipschitz continuous neural network trained on $(\mathbf{X}, \mathbf{y}) \sim p$ to zero error. There exists $\epsilon_0 > \epsilon_{\min}$ (where $\epsilon_{\min}$ is the smallest $\epsilon > 0$ for which an ϵ -adversarial example exists) such that for all $\epsilon \in]0, \epsilon_0]$, the following equivalence holds true: a data point $\mathbf{x}' \in \mathbb{R}^{d_0}$ such that $\|\mathbf{x}^* - \mathbf{x}'\| \leq \epsilon$ for some example $(\mathbf{x}^*, \mathbf{y}^*) \sim p$ satisfying $(\mathbf{y}^* - \mathcal{N}(\mathbf{x}^*))^2 \leq M_{\text{nat}}$ is an ϵ -adversarial example for $\mathcal{N}$ and p if and only if there is a real-valued function f verifying $\frac{1}{z^2} = \mathcal{O}_{z \rightarrow 0}(f(z))$ such that $\int_0^1 \frac{1}{\lambda^2} \rho^{K_{\mathcal{N}}, p_{\mathbf{x}'}}(\lambda) d\lambda = f(\epsilon)$ ¹.*

Proof. In this proof, for succinctness we denote matrix $\mathbf{K}_{\mathcal{N}}(\mathbf{X}, \mathbf{X})$ as $\mathbf{K}$. For $\epsilon > 0$ and $\mathbf{x}'$ an ϵ -adversarial example for $\mathcal{N}$ and p , consider the local Lipschitz constant of $\mathcal{N}$:

$$A_{\mathbf{x}', \mathbf{x}^*} := \frac{|\mathcal{N}(\mathbf{x}') - \mathcal{N}(\mathbf{x}^*)|}{\|\mathbf{x}' - \mathbf{x}^*\|}$$

By the second triangular inequality, we have:

$$\begin{aligned} A_{\mathbf{x}', \mathbf{x}^*} \cdot \epsilon &\geq A_{\mathbf{x}', \mathbf{x}^*} \cdot \|\mathbf{x}' - \mathbf{x}^*\| \\ &= |\mathcal{N}(\mathbf{x}') - \mathcal{N}(\mathbf{x}^*)| \\ &\geq ||\mathcal{N}(\mathbf{x}') - \mathbf{y}^*| - |\mathcal{N}(\mathbf{x}^*) - \mathbf{y}^*|| \\ &\geq M_{\text{adv}} - M_{\text{nat}} \end{aligned} \tag{10.1}$$

thus the local Lipschitz constant of $\mathcal{N}$ dominates a function of $\epsilon \rightarrow 0$ which diverges as $\frac{1}{\epsilon}$ when $\mathbf{x}'$ is adversarial.

Since $\mathcal{N}$ interpolates $(\mathbf{X}, \mathbf{y})$ and $n > d$, the readout weights column matrix $\mathbf{W}^L$ of $\mathcal{N}$ is the unique solution of the linear system $[\phi_{\mathcal{N}}(\mathbf{X})]^T \mathbf{W}^L = \mathbf{y}$, which is given by the following expression involving the empirical feature map $\phi_{\mathcal{N}}$ and covariance matrix $\mathbf{K}$ of $\mathcal{N}$ on the training set $(\mathbf{X}, \mathbf{y})$: $\mathbf{W}^L = \phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y}$. Thus, the difference between evaluations of $\mathcal{N}$ on data points $\mathbf{x}', \mathbf{x}^*$ is given by the following expression:

$$\mathcal{N}(\mathbf{x}') - \mathcal{N}(\mathbf{x}^*) = [\phi_{\mathcal{N}}(\mathbf{x}') - \phi_{\mathcal{N}}(\mathbf{x}^*)]^T \left(\phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y} \right). \tag{10.2}$$

Dividing by $\|\mathbf{x}' - \mathbf{x}^*\|$, we have:

¹In other words, $\lambda \mapsto \frac{1}{\lambda^2} \rho^{K_{\mathcal{N}}, p_{\mathbf{x}'}}(\lambda)$ is not integrable near zero as $\epsilon \rightarrow 0$, which implies a spectral mass near zero for the density of Mercer's eigenvalues.

10. Adversarial Attacks and Near-Zero Eigenvalues

$$\frac{\mathcal{N}(\mathbf{x}') - \mathcal{N}(\mathbf{x}^*)}{\|\mathbf{x}' - \mathbf{x}^*\|} = \mathbf{c}_{\mathbf{x}', \mathbf{x}^*}^T \left(\phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y} \right). \quad (10.3)$$

The left hand side is equal to $\pm A_{\mathbf{x}', \mathbf{x}^*}$ depending on the sign of $\mathcal{N}(\mathbf{x}') - \mathcal{N}(\mathbf{x}^*)$ and on the right hand side, the vector $\mathbf{c}_{\mathbf{x}', \mathbf{x}^*}$ is such that each component $\pm(\mathbf{c}_{\mathbf{x}', \mathbf{x}^*})_i$ is the local Lipschitz constant of the i -th component of $\phi_{\mathcal{N}}$ (depending on the sign of $(\phi_{\mathcal{N}}(\mathbf{x}') - \phi_{\mathcal{N}}(\mathbf{x}^*))_i$). In particular each component verifies $|(\mathbf{c}_{\mathbf{x}', \mathbf{x}^*})_i| \leq C$. Combining (10.1) and (10.3), we have that the following term of interest $A_{\mathbf{x}', \mathbf{x}^*}^2 = \left(\mathbf{c}_{\mathbf{x}', \mathbf{x}^*}^T (\phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y}) \right)^2$ dominates a function of $\epsilon \rightarrow 0$ which diverges as $\frac{1}{\epsilon^2}$ if $\mathbf{x}'$ is adversarial.

We note the vector $\mathbf{k} := \mathbf{c}_{\mathbf{x}', \mathbf{x}^*}^T \phi_{\mathcal{N}}(\mathbf{X})$ and calculate this term as:

$$\left(\mathbf{c}_{\mathbf{x}', \mathbf{x}^*}^T (\phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y}) \right)^2 = \text{Tr} \left(\mathbf{k} \mathbf{k}^T \mathbf{K}^\dagger \mathbf{y} \mathbf{y}^T \mathbf{K}^\dagger \right) \quad (10.4)$$

where we have used that $\mathbf{a}^T \mathbf{b} = \text{Tr}(\mathbf{a} \mathbf{b}^T)$.

To conclude, let $\left(\lambda_i^{K_{\mathcal{N}, p_{\mathbf{x}'}}}, \varphi_i^{K_{\mathcal{N}, p_{\mathbf{x}'}}} \right)_{i \in \mathbb{N}}$ be the Mercer's decomposition of kernel $K_{\mathcal{N}}$ and the updated data distribution $p_{\mathbf{x}'}$. Then, Mercer's theorem provides an expression of the application of the kernel to data points in terms of the aforementioned eigenvalues and eigenfunctions; it follows that the empirical covariance matrix $\mathbf{K}$ can be written as follows for some large enough number $M \gg n$:

$$\mathbf{K} := \Phi \Lambda \Phi^T \quad (10.5)$$

where $\Phi_{j,k} = \varphi_k^{K_{\mathcal{N}, p_{\mathbf{x}'}}}(\mathbf{x}_j)$ for each $1 \leq j \leq n$ and $1 \leq k \leq M$ and $\Lambda_{k,l} = \delta_{k,l} \lambda_k^{K_{\mathcal{N}, p_{\mathbf{x}'}}}$ for each $1 \leq k, l \leq M$.² Hence, using Mercer's decompositions, we can further expand (10.4) as follows:

$$\begin{aligned} \left(\mathbf{c}_{\mathbf{x}', \mathbf{x}^*}^T (\phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y}) \right)^2 &= \sum_j^M \frac{1}{(\lambda_j^{K_{\mathcal{N}, p_{\mathbf{x}'}}})^2} \\ &\quad \left(\Phi^\dagger \mathbf{y} \mathbf{y}^T (\Phi^T)^\dagger \right)_{j,j} \left(\Phi^\dagger \mathbf{k} \mathbf{k}^T (\Phi^T)^\dagger \right)_{j,j} \\ &\quad + \sum_j^M \sum_{k \neq j}^M \frac{1}{(\lambda_j^{K_{\mathcal{N}, p_{\mathbf{x}'}}})(\lambda_k^{K_{\mathcal{N}, p_{\mathbf{x}'}}})} \\ &\quad \left(\Phi^\dagger \mathbf{y} \mathbf{y}^T (\Phi^T)^\dagger \right)_{j,k} \left(\Phi^\dagger \mathbf{k} \mathbf{k}^T (\Phi^T)^\dagger \right)_{j,k} \end{aligned}$$

With this expression at hand, we are ready to show the statement in the theorem. For the forward direction, assume that $\mathbf{x}'$ is adversarial, then $\left(\mathbf{c}_{\mathbf{x}', \mathbf{x}^*}^T (\phi_{\mathcal{N}}(\mathbf{X}) \mathbf{K}^\dagger \mathbf{y}) \right)^2$ dominates a function of $\epsilon \rightarrow 0$ which diverges as $\frac{1}{\epsilon^2}$.

The first step is to show that as $\epsilon \rightarrow 0$, the following quantities remain bounded: $\left(\Phi^\dagger \mathbf{y} \mathbf{y}^T (\Phi^T)^\dagger \right)_{j,j}$, $\left(\Phi^\dagger \mathbf{k} \mathbf{k}^T (\Phi^T)^\dagger \right)_{j,j}$, and $\left(\Phi^\dagger \mathbf{y} \mathbf{y}^T (\Phi^T)^\dagger \right)_{j,k} \left(\Phi^\dagger \mathbf{k} \mathbf{k}^T (\Phi^T)^\dagger \right)_{j,k}$. Each of these

²Expression (10.5) is not the usual eigendecomposition of a square matrix: the evaluations of eigenfunctions yield rectangular (infinite) matrices. This decomposition is enabled by Mercer's theorem and applies to kernels.

10. Adversarial Attacks and Near-Zero Eigenvalues

can be written as sums over entries of the relevant matrices. In particular, since the rectangular matrix $\Phi \in \mathbb{R}^{n \times M}$ has orthogonal rows, we have $\Phi^\dagger = \Phi^T (\Phi \Phi^T)^{-1}$, and $(\Phi^T)^\dagger = (\Phi \Phi^T)^{-1} \Phi$.

For the i -th entry of $\mathbf{k}$, we have $|\mathbf{k}_i| \leq \sqrt{d}C \|\phi_{\mathcal{N}}(\mathbf{x}_i)\|$ by Cauchy-Schwartz inequality. The entries Φ remain bounded as $\epsilon \rightarrow 0$ as evaluations of Mercer's eigenfunctions. Similarly, the entries of $\Phi \Phi^T$ do not diverge as $\epsilon \rightarrow 0$, and neither do entries of $(\Phi \Phi^T)^{-1}$.

Therefore, if the term of interest is to diverge towards infinity, at least one of the sums $\sum_j^M \frac{1}{(\lambda_j^{K_{\mathcal{N}}, p_{\mathbf{x}'}})^2}$, or $\sum_j^M \sum_{k \neq j}^M \frac{1}{(\lambda_j^{K_{\mathcal{N}}, p_{\mathbf{x}'}})(\lambda_k^{K_{\mathcal{N}}, p_{\mathbf{x}'}})}$ must diverge. For small eigenvalues, the first sum dominates. This sum can be expressed using the density of Mercer's eigenvalues for $K_{\mathcal{N}}$ and $p_{\mathbf{x}'}$ as follows: $\lim_{M \rightarrow \infty} \sum_j^M \frac{1}{(\lambda_j^{K_{\mathcal{N}}, p_{\mathbf{x}'}})^2} = \int \frac{1}{\lambda^2} \rho^{K, p_{\mathbf{x}'}}(\lambda) d\lambda$. Thus, the quantity $\int_0^1 \frac{1}{\lambda^2} \rho^{K, p_{\mathbf{x}'}}(\lambda) d\lambda$ dominates a function of $\epsilon \rightarrow 0$ which diverges as $\frac{1}{\epsilon^2}$.

Conversely, assume that $\int_0^1 \frac{1}{\lambda^2} \rho^{K, p_{\mathbf{x}'}}(\lambda) d\lambda$ dominates a function of $\epsilon \rightarrow 0$ which diverges as $\frac{1}{\epsilon^2}$. We have that, as $\epsilon \rightarrow 0$, $(\Phi^\dagger \mathbf{y} \mathbf{y}^T (\Phi^\dagger)^T)_{j,j} (\Phi^\dagger \mathbf{k} \mathbf{k}^T (\Phi^\dagger)^T)_{j,j}$ is bounded away from zero. Indeed, these terms can be written as sums of squared values, which can only converge to zero if each term of the sum converges to zero, which, in turn, only happens if $\mathbf{y} = 0$ or $\mathbf{k} = 0$ ³. Thus, the term

$$\sum_j^M \frac{1}{(\lambda_j^{K_{\mathcal{N}}, p_{\mathbf{x}'}})^2} (\Phi^\dagger \mathbf{y} \mathbf{y}^T (\Phi^\dagger)^T)_{j,j} \cdot (\Phi^\dagger \mathbf{k} \mathbf{k}^T (\Phi^\dagger)^T)_{j,j}$$

implies that the square of local Lipschitz constant of $\mathcal{N}$ dominates a function of $\epsilon \rightarrow 0$ which diverges as $\frac{1}{\epsilon^2}$ and in turn, example $\mathbf{x}'$ is adversarial, as required. $\square$

It follows directly from the theorem that introducing an ϵ -adversarial example $\mathbf{x}'$ with a sufficiently small $\epsilon > 0$ yields a Mercer's decomposition where eigenvalues have a spectral mass near zero. Indeed, if eigenvalues had insufficient density near zero then the function $\frac{1}{\lambda^2} \rho^{K_{\mathcal{N}}, p_{\mathbf{x}'}}(\lambda)$ would be integrable near zero. For instance, according to the convergence/divergence of Riemann integrals, $\rho^{K_{\mathcal{N}}, p_{\mathbf{x}'}}(\lambda) \sim \lambda^\beta$ as $\lambda \rightarrow 0$ leads to non-integrability if $\beta \leq 1$ and to integrability if $\beta > 1$. As a direct corollary of our result, if an example is close to a training or a test example and induces a substantial eigenvalue density near zero, then it is an adversarial example. This could inspire new defense mechanisms and new detection methods; this is left for future work and we note that an important bottleneck, however, could be to quantify *a priori* whether a candidate example is close to a real (unobserved) example. We note that (Tramer 2022) established a hardness reduction between classifying and detecting adversarial examples which is in line with our theory. Indeed, our results suggest that the difficulty in detecting adversarial examples resides in identifying a point in the true distribution which is close to the candidate (finding a candidate which yields vanishing eigenvalues is not enough) thereby classifying it: this substantiates that detecting is as hard as classifying.

Our result is formally derived in the limit $\epsilon \rightarrow 0$. We would like to emphasize, however, that this assumption is needed to make a general statement about adversarial attacks.

³Note that these excluded edge cases are already taken in account as particular cases of (10.2).

Indeed, small ϵ is implicit in the very definition of L_2 adversarial example, since it must be a small perturbation of a real example. In particular, as hinted in the Theorem, the $\epsilon_0 > 0$, for which we have proved existence, relates to the adversarial robustness of a trained network: $\epsilon_{\min} := \operatorname{argmin}_{\epsilon_0 > 0} \{ \exists \epsilon \in]0, \epsilon_0] : \exists \epsilon - \text{adversarial example} \}$ which characterizes the adversarial example with smallest distance to a real example.

10.2. Adversarial Examples Are Exceedingly Unlikely

In this section, we exploit the result in Theorem 10.1 to show that adversarial examples have zero measure with respect to the true data distribution p . As in the previous section, we assume that the neural network is trained to zero error and that its generalisation error remains small for all examples drawn from the same distribution p as the training data. In this setting, we prove that the probability of randomly sampling an ϵ -adversarial example from p tends to zero as $\epsilon \rightarrow 0$. Intuitively, Theorem 10.1 tells us that the density function $\frac{1}{\lambda^2} \rho^{K_{\mathcal{N}}, p}(\lambda)$ for the training data is integrable near zero; this highly restricts the probability of sampling near-zero eigenvalues, and consequently also the probability of sampling adversarial examples.

Theorem 10.2. *Let $\mathcal{N}$ be a Lipschitz continuous neural network trained to zero error on $(\mathbf{X}, \mathbf{y}) \sim p$ and let $\mathbb{E}_{(\mathbf{x}, \mathbf{y}) \sim p} ((y - \mathcal{N}(\mathbf{x}))^2) \leq M_{\text{nat}}$. Consider the indicator random variable $\mathbb{1}_{\epsilon}^{\mathcal{N}}$ which determines whether a vector $\mathbf{x} \sim p$ is an ϵ -adversarial example for $\mathcal{N}$ and p . There exists $\epsilon_0 > \epsilon_{\min}$, where $\epsilon_{\min}$ is the smallest $\epsilon > 0$ for which an ϵ -adversarial example exists, such that for all $\epsilon \in]0, \epsilon_0]$, it holds that $p(\mathbb{1}_{\epsilon}^{\mathcal{N}} = 1) = 0$.*

Proof. Before proving the statement of the theorem, we define a set of events with useful probabilities. Let λ^j be the random variable assigning to each $\mathbf{x} \sim p$ the j -th Mercer eigenvalue for $K_{\mathcal{N}}$ and $p_{\mathbf{x}} = \frac{n}{n+1}p + \frac{1}{n+1}\delta_{\mathbf{x}}$. For an interval $[a, b]$, let $E_{[a, b]}^j$ be the event of λ^j taking values within $[a, b]$ and define the event $\{\Lambda = \lambda\} := \bigcup_j E_{[\lambda, \lambda + d\lambda]}^j$.

Let $\mathbb{1}_{\epsilon}^{\mathcal{N}}$ be the indicator random variable determining whether $\mathbf{x} \sim p$ is an ϵ -adversarial example for $\mathcal{N}$ and p . By the law of total probability applied to $\mathbb{1}_{\epsilon}^{\mathcal{N}}$, the following holds:

$$\begin{aligned} p(\Lambda = \lambda) &= p(\Lambda = \lambda | \mathbb{1}_{\epsilon}^{\mathcal{N}} = 1) \cdot p(\mathbb{1}_{\epsilon}^{\mathcal{N}} = 1) \\ &\quad + p(\Lambda = \lambda | \mathbb{1}_{\epsilon}^{\mathcal{N}} = 0) \cdot p(\mathbb{1}_{\epsilon}^{\mathcal{N}} = 0) \end{aligned}$$

The probability $p(\Lambda = \lambda)$ coincides with the average density of Mercer's eigenvalues with respect to the random vector $\mathbf{x} \sim p$:

$$\begin{aligned} p(\Lambda = \lambda) &= \int p(\Lambda = \lambda | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} \\ &= \int \rho^{K_{\mathcal{N}}, p_{\mathbf{x}}}(\lambda) p(\mathbf{x}) d\mathbf{x} \end{aligned}$$

Since $\mathbb{E}_{(\mathbf{x}, \mathbf{y}) \sim p(\mathbf{x}, \mathbf{y})} ((y - \mathcal{N}(\mathbf{x}))^2) \leq M_{\text{nat}}$, it is easy to show, using the same arguments as in the proof of Theorem 10.1, that $\lambda \mapsto \frac{1}{\lambda^2} p(\Lambda = \lambda)$ is integrable near zero.

By definition, $p(\Lambda = \lambda | \mathbb{1}_{\epsilon}^{\mathcal{N}} = 1)$ is the average density of Mercer eigenvalues for $K_{\mathcal{N}}$ and $p_{\mathbf{x}}$ given that $\mathbf{x}$ is an ϵ -adversarial example. By Theorem 10.1, in the limit $\epsilon \rightarrow 0$,

10. Adversarial Attacks and Near-Zero Eigenvalues

$p(\Lambda = \lambda | \mathbb{1}_\epsilon^\mathcal{N} = 1)$ is strictly positive near zero, which gives us the following identity as $\lambda \rightarrow 0$ and $\epsilon \rightarrow 0$:

$$\begin{aligned} p(\mathbb{1}_\epsilon^\mathcal{N} = 1) &= \frac{p(\Lambda = \lambda)}{p(\Lambda = \lambda | \mathbb{1}_\epsilon^\mathcal{N} = 1)} \\ &\quad - \frac{p(\Lambda = \lambda | \mathbb{1}_\epsilon^\mathcal{N} = 0)}{p(\Lambda = \lambda | \mathbb{1}_\epsilon^\mathcal{N} = 1)} \cdot p(\mathbb{1}_\epsilon^\mathcal{N} = 0) \end{aligned}$$

By Theorem 10.1, $\lambda \mapsto \frac{1}{\lambda^2} p(\Lambda = \lambda | \mathbb{1}_\epsilon^\mathcal{N} = 1)$ is not integrable near-zero whereas $\lambda \mapsto \frac{1}{\lambda^2} p(\Lambda = \lambda | \mathbb{1}_\epsilon^\mathcal{N} = 0)$ is integrable near-zero. In particular, $\frac{p(\Lambda=\lambda|\mathbb{1}_\epsilon^\mathcal{N}=0)}{p(\Lambda=\lambda|\mathbb{1}_\epsilon^\mathcal{N}=1)} \rightarrow 0$ as $\lambda \rightarrow 0$. Similarly, $\frac{p(\Lambda=\lambda)}{p(\Lambda=\lambda|\mathbb{1}_\epsilon^\mathcal{N}=1)} \rightarrow 0$ as $\lambda \rightarrow 0$, which yields $p(\mathbb{1}_\epsilon^\mathcal{N} = 1) = 0$, as required. $\square$

Our findings indicate that in practical evaluations of neural models, it is highly improbable for test sets to include adversarial examples, given that they are sampled from the same underlying distribution p as the training data. In essence, adversarial examples created through artificial perturbations of samples drawn from p are considered out-of-distribution and thus extremely unlikely to originate from the same underlying data generation process that produced the training and test data. This substantiates the intuition that adversarial examples do not naturally manifest in real-world scenarios.

10.3. Experiments

In this section, to illustrate our experiments, we describe the details of our methodology and we present results for two architectures; FCN and (pre-trained) CNN; and two datasets: a subset of MNIST and a subset of CIFAR10. We include different combinations of dataset, attack methods and architecture. The results are largely consistent across experiments.

We conducted these experiments on a subset of MNIST consisting exclusively of classes "0" and "1", and a subset of CIFAR10 consisting exclusively of classes "plane" and "car". The subset of MNIST comprised 253 examples, each with 784 pixels per image. The subset of CIFAR10 comprised 100 examples each with 1024 coloured pixels. Additionally, we established a test set consisting of 17 examples (respectively, 10 examples) for MNIST (respectively, for CIFAR10). We deliberately opted for these reduced dataset sizes to accommodate computational constraints, as our computations require diagonalising kernel matrices with dimensions of $(n + 1) \times (n + 1)$. Furthermore, we used only two classes per experiment to conveniently turn classification tasks into one-dimensional regression tasks. However, it is important to note that the size of the dataset is inconsequential to our theoretical results, as they remain independent of dataset scale. All experiments were conducted on a GPU-enabled platform within Google Colab for enhanced computational efficiency.

Using Pytorch, we trained ReLU FCNs with one hidden layer of size $d = 512$ and the pre-trained network VGG (Simonyan & Zisserman 2014) to zero error on our MNIST and CIFAR10 training datasets and 100% accuracy on our restricted test sets.

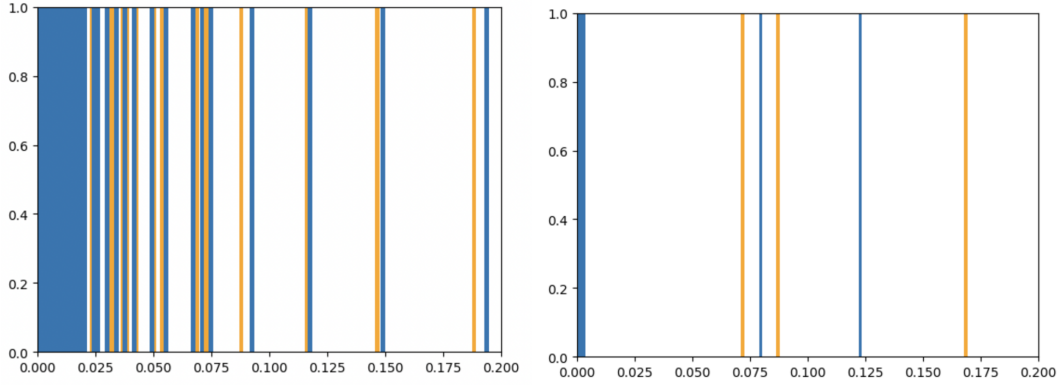


Figure 10.1.: PGD: Eigenvalue distribution for kernel K_N and the original data distribution (respectively, the same data distribution extended with one adversarial example) in orange (respectively, in blue). The X-axis is indexed by the value of the eigenvalues and the Y-axis is a probability density. On the left, the result for the MNIST experiment and on the right, the result for the CIFAR10 experiment with PGD.

Subsequently, we exploited the PGD algorithm (Madry et al. 2018), DeepFool (Moosavi-Dezfooli et al. 2016), Fast Adaptive Boundary (Croce & Hein 2020) and Fast Gradient Sign Method (FGSM) (Goodfellow et al. 2015) to generate one adversarial example for each of the 17 test examples. We used the library torchattacks for their implementation of the different attack methods.

For each experiment, we computed the eigenvalue distributions of the empirical kernel for the original training data distribution. Then, for each adversarial example, we computed the updated eigenvalue distribution for the same empirical kernel and the training data extended with the adversarial example. Thus, this gives us 17 different updated eigenvalue distributions for the MNIST experiment and 10 different updated eigenvalue distributions for the CIFAR10 experiment. To compute these distributions, we constructed 17 (respectively 10) new kernel matrices by adding the relevant row and column corresponding to the adversarial example and diagonalised them using PyTorch. By the classical results of Baker (1977); Rasmussen & Williams (2006), the Mercer eigenvalues roughly coincide with the eigenvalues obtained by diagonalising the corresponding kernel matrix. We have plotted for each experiment examples of such updated distributions against the original ones (Figures 10.1 to 10.5); note how introducing an adversarial example shifts the eigenvalue distribution towards zero. For one set of experiments with PGD, we also increased the number of examples to 500 training points and 100 test points, to showcase that the distribution shift remains apparent, corroborating that our theory is agnostic to dataset scale.

10.4. Concluding remarks of the chapter

Our results are currently directly applicable to one-dimensional regression tasks when achieving exact interpolation, but we anticipate that our core insights could be generalised to encompass classification and multidimensional regression, and could be relaxed to characterize approximate interpolation. Additionally, conducting compre-

10. Adversarial Attacks and Near-Zero Eigenvalues

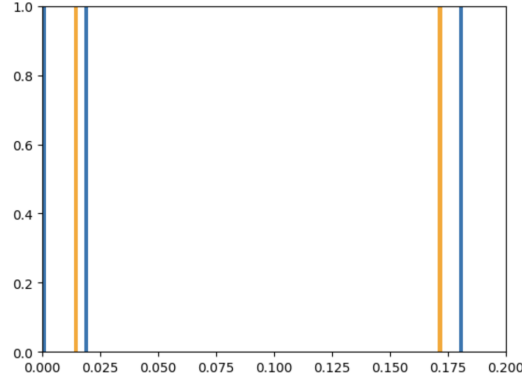


Figure 10.2.: PGD with 500 examples: Eigenvalue distribution for kernel K_N and the original data distribution (respectively, the same data distribution extended with one adversarial example) in orange (respectively, in blue). The X-axis is indexed by the value of the eigenvalues and the Y-axis is a probability density.

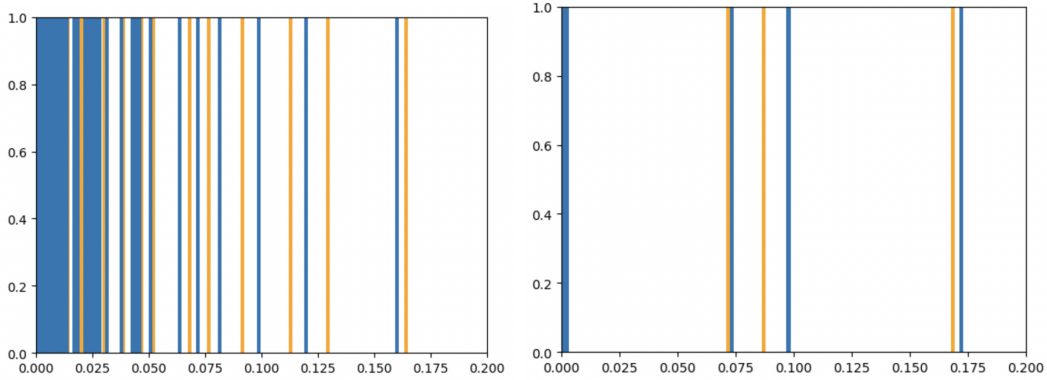


Figure 10.3.: DeepFool: Eigenvalue distribution for kernel K_N and the original data distribution (respectively, the same data distribution extended with one adversarial example) in orange (respectively, in blue). The X-axis is indexed by the value of the eigenvalues and the Y-axis is a probability density. On the left, the result for the MNIST experiment and on the right, the result for the CIFAR10 experiment with DeepFool.

hensive experiments on more extensive and diverse datasets would be beneficial. Importantly, our findings remain unaffected by variations in problem dimensions and data characteristics, offering a high degree of versatility.

Our results are derived in the limit where adversarial examples become infinitesimally close to data points. Specifically, we have not characterised the rate at which the probability of encountering adversarial examples diminishes with the distance to an example from the true data distribution. This remains a subject for future research and requires the derivation of precise bounds for the generalisation error. Exploring specific criteria governing the data distribution and neural architecture that lead to rapid convergence rates (as $\epsilon \rightarrow 0$) and result in low probabilities of generating adversarial examples is a promising avenue for which our characterisation provide a new lens. Achieving this understanding could lay the groundwork for designing more robust architectures. Additionally, while we have proved that adversarial examples have a

10. Adversarial Attacks and Near-Zero Eigenvalues

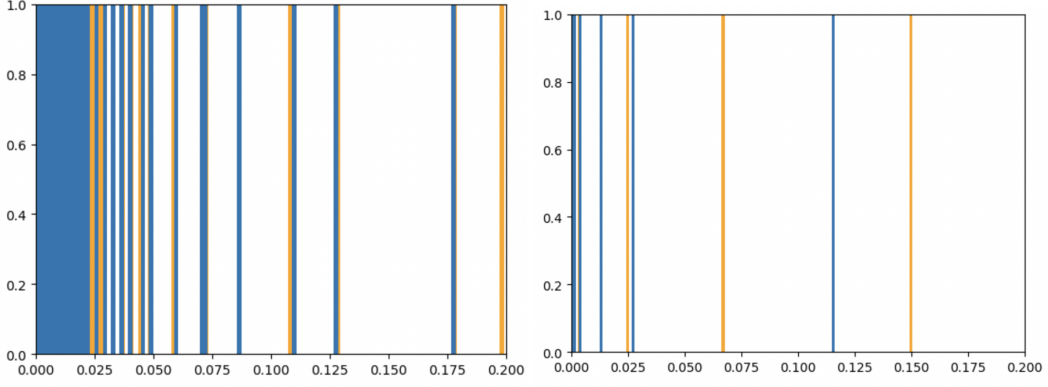


Figure 10.4.: FAB: Eigenvalue distribution for kernel K_N and the original data distribution (respectively, the same data distribution extended with one adversarial example) in orange (respectively, in blue). The X-axis is indexed by the value of the eigenvalues and the Y-axis is a probability density. On the left, the result for the MNIST experiment and on the right, the result for the CIFAR10 experiment with FAB.

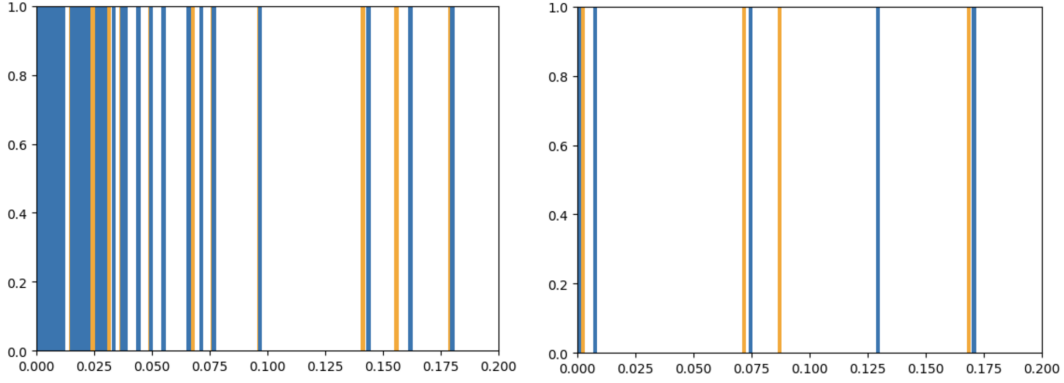


Figure 10.5.: FGSM: Eigenvalue distribution for kernel K_N and the original data distribution (respectively, the same data distribution extended with one adversarial example) in orange (respectively, in blue). The X-axis is indexed by the value of the eigenvalues and the Y-axis is a probability density. On the left, the result for the MNIST experiment and on the right, the result for the CIFAR10 experiment with FGSM.

measure zero with respect to the data distribution in the limit, we have not shown that examples producing near-zero Mercer eigenvalues for the empirical kernel of a neural network always exist.

In conclusion, we anticipate that our findings could serve as a catalyst for research into the characteristics and prevalence of adversarial examples. Furthermore, we hope to encourage the exploration and development of novel defense mechanisms and detection methods against adversarial attacks.

11. Algorithmic Stability and Low-Rank Structure

Background. The stochastic gradient descent (SGD) family of algorithms has emerged as the go-to tool for training machine learning models on a vast amount of data. While the generalisation ability of such algorithms is reasonably well-understood when learning involves solving convex or quasi-convex optimisation problems, the picture is much less clear for non-convex learning problems involving overparameterized models, such as that of training a deep neural network. Yet, the empirical evidence strongly suggests that the generalisation ability of SGD algorithms remains high despite overparameterization in non-convex settings. These observations appear to be in conflict with the classical learning-theoretic *complexity-fit tradeoff* viewpoint (Bousquet et al. 2004; Zhang et al. 2021).

In recent years, there has been growing interest in the hypothesis that the favorable generalisation performance of optimisation algorithms can be accounted for by the algorithm’s *implicit bias* or inherent model capacity control. A prominent idea in this setting is that of *minimum-complexity interpolation*, which states that the algorithm finds the simplest solution among those achieving zero (or near-zero) training error. While this phenomenon is well established in the context of linear regression (e.g. a pseudo-inverse solution to the least-squares problem recovers parameters with a minimum L_2 norm), the suggestion that it also applies to the training of a deep overparameterized ReLU neural networks is only rather recent. In particular, it has been shown theoretically that the gradient flow (GF) algorithm (gradient descent can be seen as a discretization of GF) asymptotically fits an interpolating ReLU neural network with the minimum L_2 norm of the parameters (Lyu & Li 2019; Ji & Telgarsky 2020; Phuong & Lampert 2020). GF is thought to be a good proxy for gradient descent algorithms because the approximation error introduced by discretization remains controlled under certain regularity assumptions (Elkabetz & Cohen 2021). Recent works have attempted to reconcile minimum-norm interpolation with classical generalisation theories based on the uniform convergence principle such as Rademacher complexity (Bartlett & Mendelson 2002; Bartlett et al. 2017). Indeed, dimension-free bounds have provided some explanation as to why we do not observe overfitting in the absence of label noise (Ji & Telgarsky 2019; Telgarsky 2022), and observe only *benign overfitting* otherwise (Bartlett et al. 2020; Koehler et al. 2021; Frei et al. 2023).

In this chapter, we explore the generalisation capabilities of minimum-norm interpolating neural networks from the alternative viewpoint of *algorithmic stability*. Stable learning algorithms are insensitive to small perturbations (e.g. removal or replacement of data points) of the training set and generalise well under mild assumptions (Bousquet & Elisseeff 2002; Shalev-Shwartz et al. 2010). Moreover, stability analysis provides valuable insights beyond generalisation, such as controlling the variance of the algorithm, which is crucial for uncertainty quantification methods like bootstrapping (Elisseeff et al. 2005). Many algorithms have been shown to be stable, including nonparametric predictors (e.g. nearest-neighbors) (Devroye & Wagner 1979), minimizers of strongly

convex problems (such as the ridge regression estimator) (Bousquet & Elisseeff 2002), GD-type algorithms minimizing convex and smooth objectives (Hardt et al. 2016; Lei & Ying 2020), as well as quasi-convex objectives (Charles & Papailiopoulos 2018; Richards & Kuzborskij 2021).

Despite these advances, success has so far been limited in the context of neural networks. Although several works have analyzed the stability of SGD in the non-convex setting, their results come with significant limitations. First, vacuous bounds are typically obtained when the number of training steps (or time) far exceeds the sample size, i.e. $t \gg n$ (Hardt et al. 2016; Kuzborskij & Lampert 2018; Richards & Rabbat 2021; Wang et al. 2023a); at the same time, for a large enough model capacity, we expect interpolation to happen when $t \gg n$. Second, strong assumptions such as Lipschitzness of the loss function *in the parameters* or penalization of the objective (Hardt et al. 2016; Kuzborskij & Lampert 2018; Farghly & Rebeschini 2021) are often required. Third, these works assume *parameter stability*, meaning that parameters are expected to remain close (typically in the Euclidean distance) if the training set is perturbed slightly; this is often unrealistic in the context of neural networks since the same predictor can be expressed using very different parameters thanks to symmetries in the weight matrices and because of non-convexity of the objective function. Finally, these works focus on optimisation aspects, which seldom reveal insight into the structural properties of neural networks obtained by stable algorithms.

Contributions. In this chapter, we hypothesize that stability of deep nonlinear networks originates in the early layers, and is preserved throughout the subsequent layers. Specifically, we study the algorithmic stability of minimum-norm interpolation with deep ReLU neural networks, and identify sufficient conditions for stability. In particular, such interpolations of neural networks are stable if they contain a contiguous, *stable sub-network* (for instance, the few first layers), and this sub-network is followed by at least one *low-rank weight matrix*. See Figure 11.1 for an illustration.

For the data-generating process, we assume the existence of a neural network with the same architecture that can interpolate the data using bounded weights, although not necessarily with the minimal norm. This implies a setting where data cannot be arbitrarily complex ensuring that weights of such a network remain bounded as the training set grows.

Our analysis is inspired by recent theoretical and empirical findings suggesting that deep interpolating neural networks exhibit a low-rank weight matrix structure (Frei et al. 2022; Timor et al. 2023; Galanti et al. 2023). Thus, the idea behind the proof is to ensure that if a stable sub-network exists, the remaining low-rank weight matrices will preserve its stability. Our proof separates the concepts of low-rank bias and sub-network stability into distinct phenomena. This separation enables isolated analysis, simplifying the study of stability in neural networks to the potentially easier task of understanding why stable sub-networks exist. Furthermore, we conduct a series of experiments in which we observe that stable prefix sub-networks indeed occur in practice.

In our analysis we also take a step towards addressing some limitations in the literature discussed earlier. Algorithmically, interpolating neural networks studied here can be obtained as solutions of a GF and so, unlike previous results, our findings hold for

11. Algorithmic Stability and Low-Rank Structure

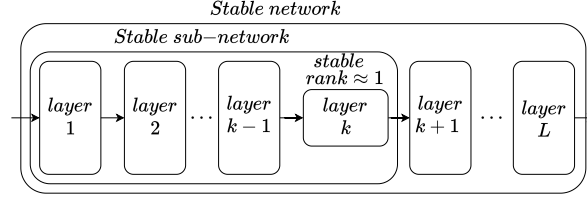


Figure 11.1.: **A diagram of our main result.** Our main result relies on three arguments: a) the data is expressible by a neural network with finite weight matrix norm (Assumption 11.1), b) the minimum-norm interpolating ReLU neural network contains at least one layer with a low stable rank matrix (see Lemma 11.3) and c) the sub-network is stable (Hypothesis 11.4). In this scenario, we show in Theorem 11.5 that the full network is also stable.

$t \gg n$ and potentially $t \rightarrow \infty$. In contrast to existing works, the analysis does not require regularity assumptions such as Lipschitzness or smoothness in the parameters, and parameter stability. We note however that our results make different assumptions, in particular low-rank bias and sub-network stability, which could implicitly require some of the same regularity assumptions to hold.

In the following, we first remind the implication of algorithmic stability on generalisation, then we present our technical result: sufficient conditions for algorithmic stability of minimum-norm deep interpolating neural networks.

Stability implies generalisation. Let $l : \mathbb{R}^2 \rightarrow [0, M]$ be a fixed known loss function. Then the *risk* and the *empirical risk* of the predictor parameterized by θ are respectively defined as $R(\theta) = \int l(\mathcal{N}(\mathbf{x}; \theta), y) d\mathbf{p}(\mathbf{x}, y)$ and $\hat{R}(\theta) = \frac{1}{n} \sum_{i=1}^n l(\mathcal{N}(\mathbf{x}_i; \theta), y_i)$. It is known that if $\hat{\theta}$ is generated by an ϵ -uniformly-stable algorithm $\mathcal{A}$, there exist universal constants $c_1, c_2 > 0$ such that for any $\delta \in (0, 1)$ (Feldman & Vondrak 2018; Bousquet et al. 2020),

$$\mathbb{P} \left(|R(\hat{\theta}) - \hat{R}(\hat{\theta})| \leq c_1 \ln(n) \ln(1/\delta) \epsilon + c_2 M \sqrt{\frac{\ln(1/\delta)}{n}} \right) \quad (11.1)$$

$$\geq 1 - \delta. \quad (11.2)$$

Hence, the gap between the population loss and empirical loss is controlled with high probability as long as the algorithm is stable.

Problem set-up. In this chapter, we assume that the training set $(\mathbf{X}, \mathbf{y})$ consists of n examples $(\mathbf{x}_i, y_i)$ sampled i.i.d. from an unknown distribution p on $\mathcal{X} \times \{-1, +1\}$ where the input space $\mathcal{X}$ is an Euclidean ball of radius one.

Furthermore, we assume that the activation function for each layer is ReLU and that each layer has no bias $\mathbf{b}^\ell = \mathbf{0}$.

A useful property of ReLU is *positive homogeneity*: $\text{ReLU}(\alpha x) = \alpha \text{ReLU}(x)$ for each $x \in \mathbb{R}$ and $\alpha \geq 0$. In particular, we have $\mathcal{N}(\alpha \mathbf{x}; \theta) = \alpha \mathcal{N}(\mathbf{x}; \theta)$ and $\mathcal{N}(\mathbf{x}; \alpha \theta) = \alpha^L \mathcal{N}(\mathbf{x}; \theta)$.

11.1. From sub-network stability to prediction stability

In this section, we present our technical result and discuss its implications. Our results depend on one technical assumption and one main hypothesis, which we introduce next. The first assumption concerns the complexity of the learning problem (Timor et al. 2023):

Assumption 11.1 (*B-admissible training set*). Given a finite $B > 0$, we call a training set B -admissible if there exists a neural network $\mathcal{N}$ of architecture $\langle L^*, d, d_0 \rangle$ for some $L^* \geq 2$ with parameters θ^* such that $\mathcal{N}(\mathbf{x}_i; \theta^*) = y_i$ for $i \in \{1, \dots, n\}$ and its weight matrices satisfy $\max_k \|\mathbf{W}_k^*\|_F \leq B$.

Under this assumption, the training set can be viewed as generated by a hypothetical teacher network whose weight matrices have bounded norms. Note that Assumption 11.1 is only meaningful in learning scenarios where data cannot be arbitrarily complex (otherwise, one can construct instances such that $B \rightarrow \infty$ as $n \rightarrow \infty$).

We next define our (chapter-specific) notion of sub-network.

Definition 11.2 (*Sub-network*). Given $1 \leq k \leq L - 1$, consider the following decomposition of weight matrix $\mathbf{W}_k \in \mathbb{R}^{d \times d}$: $\mathbf{W}_k = \lambda_k \mathbf{u}_k \mathbf{v}_k^\top + \mathbf{W}_k^\epsilon$ where $\lambda_k > 0$ is the leading singular value of $\mathbf{W}_k$, $\mathbf{u}_k, \mathbf{v}_k$ are unitary vectors, $\mathbf{u}_k$ is the leading left singular vector of $\mathbf{W}_k$ and $\mathbf{v}_k$ is the leading right singular vector of $\mathbf{W}_k$. Then, a **sub-network** at position k is defined as

$$f_k(\mathbf{x}; \theta) := \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}; \theta). \quad (11.3)$$

Invoking the recent results from Timor et al. (2023), it is easy to see that under Assumption 11.1, a trained deep network contains at least one layer with low stable rank (the proof is given in Section 11.2.1).

Lemma 11.3. *Suppose that datasets are B-admissible according to Assumption 11.1. Given $a > 0$ and $\epsilon = M/n^{-\alpha}$, for some $M \geq 0, \alpha > 0$, there exists $L \geq L^*$ and $1 \leq k \leq L - 1$ such that with parameters $\hat{\theta}$ generated by algorithm $\mathcal{T}$ for architecture $\mathbb{A} = \langle L, d, d_0 \rangle$, the following holds:*

$$S(\hat{\mathbf{W}}_k) \leq 1 + a\epsilon. \quad (11.4)$$

Finally, we state our key hypothesis, namely the existence of a stable sub-network.

Hypothesis 11.4 (β -uniformly ϵ -stable sub-network). Given $\beta \in [0, 1]$ and $\epsilon = M/n^{-\alpha}$, for some $M \geq 0, \alpha > 0$, and $L \geq L^*$, with parameters $\hat{\theta}$ and $\hat{\theta}^{(i)}$ generated by algorithm $\mathcal{T}$ for architecture $\mathbb{A} = \langle L, d, d_0 \rangle$, the following holds:

$$\mathbb{P} \left(\left| f_k(\mathbf{x}; \hat{\theta}) - f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right| \leq \epsilon \right) \geq 1 - \beta.$$

In support of this hypothesis, we present empirical evidence in Figures 11.2 and 11.3 taken from (El Harzli et al. 2024b): trained FCNs contain sub-networks (followed by a layer with low stable rank) whose stability is on par with that of the full network.

11. Algorithmic Stability and Low-Rank Structure

Now we present our main result (a complete statement is given in Theorem 11.5):

Sufficient conditions for algorithmic stability (sketch). Let $a > 0$ and $\epsilon = n^{-\alpha}$ for some $\alpha > 0$ and suppose that there exist (B, k, L, β) such that Assumption 11.1, and Hypothesis 11.4 are satisfied. Then, assuming that sample size satisfies $n = \Omega(\max(a B^L, B^{L-k+1}))$, there is a universal constant $C > 0$ such that $\mathcal{T}$ is β -uniformly

$$C (1 + B^{2L-k+1} + a B^{3L-k+1}) \epsilon \text{ --- stable .}$$

The main implication of the above result is that the stability of the entire minimum-norm interpolating neural network is controlled by the stability of its sub-network. This fact is non-trivial because since $\hat{\theta}$ and $\hat{\theta}^{(i)}$ are different (thus in particular the deeper layers $\hat{\mathbf{W}}_{k+1}, \dots, \hat{\mathbf{W}}_L$ and $\hat{\mathbf{W}}_{k+1}^{(i)}, \dots, \hat{\mathbf{W}}_L^{(i)}$), there is no *a priori* reason to believe that they would preserve any stability that originates in early layers. The proof requires the key assumption that the stable sub-network is followed by a layer with a weight matrix of a low stable rank, which will preserve the signal as it propagates into deeper layers. While this assumption might initially seem strong, (El Harzli et al. 2024b) observe it to hold in practice (Figure 11.2). Furthermore, recently Timor et al. (2023) showed that the stable rank decays roughly at the rate $B^{\frac{L}{k}}$ where B can be interpreted as a problem complexity term (see Assumption 11.1).¹ In our case, the stable rank decay rate is assumed to be $a \epsilon$ with a free parameter $a \geq 0$.

Note that while the overall stability scales linearly with the stability of the sub-network, the bound is attenuated by a B -dependent factor. One extreme (pessimistic) case is $a \gg 0$, that is when weight matrices are sufficiently far from rank-one. Then, the factor in the worst case becomes of order B^{3L} . Intuitively, this occurs because, learning all perturbation matrices $(\mathbf{W}_k^\epsilon)_k$ (see Definition 11.2) becomes unavoidable. In another extreme case of rank-one matrices and a shallow stable sub-network ($a = 0, k = 2$), we have an overall stability bound of order $B^{2L} \epsilon$.

In a more optimistic scenario, a is sufficiently small such that weight matrices have a stable-rank close to one. In this case we have a dominant factor B^{2L-k+1} , which captures the cost of training a neural network that is deeper than the sub-network. In particular, observe that the deeper the stable sub-network is (increasing k), the smaller the cost. For a deep stable sub-network ($k = L - 1$), we reach a factor B^L which is similar to dimension-free analysis of deep ReLU neural networks (Golowich et al. 2018).

Finally, Figure 11.3 (taken from (El Harzli et al. 2024b)) provides some empirical validation of our main hypothesis and our main result by showing that the prediction stability of the entire network is roughly of the same order (with respect to n) as that of the sub-network stability: this is captured by similar slopes in the log-scale, which corresponds to $-\alpha$ in $\epsilon = n^{-\alpha}$ assumption.

Applications. The stability bound we presented can also be used in some applications. In particular, when combined with Equation (11.1), it implies a high-probability bound

¹In fact, they showed a slightly stronger upper bound on the harmonic mean: $\frac{k}{\sum_{j=1}^k (1/S(\hat{\mathbf{W}}_j))} \leq B^{\frac{L}{k}}$.

11. Algorithmic Stability and Low-Rank Structure

on the generalisation error:

$$\begin{aligned} & |R(\hat{\theta}) - \hat{R}(\hat{\theta})| \\ &= \tilde{\mathcal{O}} \left((1 + B^{2L-k+1} + a B^{3L-k+1}) \epsilon + \frac{1}{\sqrt{n}} \right). \end{aligned}$$

Under assumption $\epsilon = n^{-\alpha}$ for $\alpha > 0$, the generalisation error converges 0 as $n \rightarrow \infty$ in probability.

Finally, the stability bound can also be used to give a bound on the variance of a trained neural network, which is not normally achievable through the uniform convergence bounds. Controlling the variance of trained predictors is interesting in the context of uncertainty quantification (e.g., through ensembling, as discussed in (Elisseeff et al. 2005)). In particular, Efron-Stein inequality (Boucheron et al. 2013) implies that

$$\text{Var}(\mathcal{N}(\hat{\theta})) \leq \frac{C^2}{2} (1 + B^{2L-k+1} + a B^{3L-k+1})^2 n \epsilon^2.$$

Once again, assuming $\epsilon = n^{-\alpha}$ it turns out that the variance converges to 0 asymptotically only if $\alpha > 1/2$, that is when sub-network is very stable with $\epsilon < 1/\sqrt{n}$.

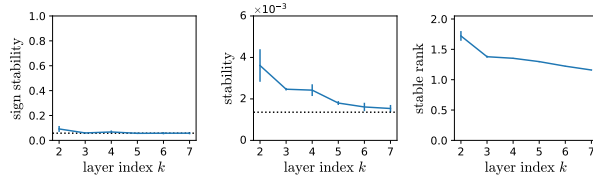


Figure 11.2.: **Stability of sub-networks and stable rank of the layers.** (El Harzli et al. 2024b) trained an 8-layer FCN on a uniformly drawn 10^4 MNIST sample by minimizing a mean square error (MSE) loss to near zero, classifying the first 5 classes as -1 and others as 1 . They performed multiple trials, where each trial is with identical initialization and a different portion of the training set is replaced for each trial. The error bars are 1 standard deviation of the trial. Using the models, They measured the sign stability (**left**), i.e. $|\text{sign}(f_k(\mathbf{x}; \hat{\theta})) - \text{sign}(f_k(\mathbf{x}; \hat{\theta}^{(i)}))|$, stability (**middle**), and the stable rank of weight matrix (**right**) for each sub-network f_k for $2 \leq k \leq 7$. The horizontal dotted lines are the (sign) stability of the full network. For the details of the experiment, and additional experiments on Fashion-MNIST, see (El Harzli et al. 2024b).

11.1.1. Proof of the main technical result

Theorem 11.5. Suppose that datasets are B -admissible according to Assumption 11.1. Let $\epsilon = M/n^{-\alpha}$, for some $M \geq 0, \alpha > 0$ and let $a > 0$, consider $L \geq L^*$ and $1 \leq k \leq L-1$ that satisfy (11.4). Suppose that the sub-network is β -uniformly ϵ -stable according to Hypothesis 11.4. Then, assuming that the sample size satisfies $n \geq \max(a \cdot 2MB^L, 4MB^{L-k+1})^{\frac{1}{\alpha}}$, $\mathcal{T}$ is β -uniformly ϵ' -prediction stable with

$$\begin{aligned} \epsilon' &= (1 + 8B^{2L-k+1}) \epsilon \\ &\quad + 2a \left(B^L + B^{2L-k+1} \epsilon + 4B^{3L-k+1} \right) \epsilon. \end{aligned}$$

11. Algorithmic Stability and Low-Rank Structure

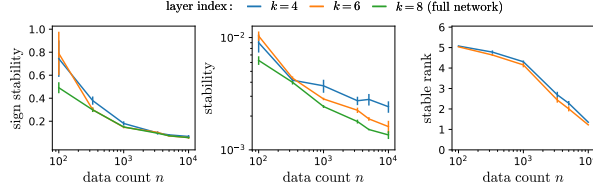


Figure 11.3.: **Stability as a function of number of data points.** (El Harzli et al. 2024b) followed the same setting as in Figure 11.2 while varying training data set sizes. Both the sign stability (**left**) and stability (**middle**) of sub-networks (blue and orange) decay at a rate similar to that of the full network (green). The stable rank of weight matrices (**right**) also decreases as a function of n , suggesting that Lemma 11.3 holds in the large n limit. Observe that the slopes for the sub-networks and the full network are similar which validates that the respective stabilities have the same dependency in n (Theorem 11.5).

The proof relies on Lemmas 11.3, 11.6 and 11.7, shown Section 11.2. First, we discuss high-level proof ideas.

Some proof ideas For simplicity consider a rank-one case ($a = 0$). The key Lemma 11.6 shows that if a deep ReLU network interpolates the data, then the prediction done at the rank-one layer is maintained throughout the rest of the layers. Consider a decomposition $\mathbf{W}_k = \lambda_k \mathbf{u}_k \mathbf{v}_k^\top$, where $\mathbf{v}_k$ defines a hyperplane which separates inputs in the feature space of the previous layer given by $\mathcal{N}^{1:k-1}(\cdot, \hat{\theta})$. Then, multiplication by $\lambda_k \mathbf{u}_k$ and the propagation through all subsequent layers is just a rescaling of the predictions to fit the labels. In other words, the sign of $f_k(\mathbf{x}) = \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}, \hat{\theta})$ for each $\mathbf{x}$ already determines the final output and the prediction is done at the sub-network level.

Next, in Lemma 11.7, we show that if a deep ReLU network is a minimum-norm interpolant of the data, the intermediate prediction of data points at the sub-network level must have substantial margin γ . Intuitively, if the margin were infinitesimally small, this would require a large contribution in (one of the weights of) the subsequent layers to compensate and yield an output of order 1, which is forbidden by the fact that the network has minimum norm.

Lemma 11.6. Consider a subnetwork f_k as defined in Definition 11.2. Then, there exists a bounded function $b(\cdot; \theta)$, and $C_\theta^+, C_\theta^- > 0$ (independent from the input), such that, for all $\mathbf{x} \in \mathbb{R}^{d_0}$ the following is true:

1. If $f_k(\mathbf{x}; \theta) > 0$, then $\mathcal{N}(\mathbf{x}) + b(\mathbf{x}; \theta) \cdot \epsilon = C_\theta^+ \cdot f_k(\mathbf{x}; \theta)$.
2. If $f_k(\mathbf{x}; \theta) \leq 0$, then $\mathcal{N}(\mathbf{x}) + b(\mathbf{x}; \theta) \cdot \epsilon = C_\theta^- \cdot f_k(\mathbf{x}; \theta)$.
3. For all $\mathbf{x} \in \mathbb{R}^{d_0}$, $|b(\mathbf{x}; \theta)| \leq a \left(\prod_{j=1}^L \|\mathbf{W}_j\|_2 \right)$.
4. $\|\mathbf{W}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x})\| \leq a \epsilon \prod_{j=1}^k \|\mathbf{W}_j\|_2$.

Lemma 11.7. Consider a subnetwork f_k as defined in Definition 11.2. Suppose that datasets are B -admissible according to Assumption 11.1. Let $\hat{\theta} = \mathcal{T}(\mathbf{X}, \mathbf{y})$ and $\hat{\theta}^{(i)} = \mathcal{T}(\mathbf{X}^{(i)}, \mathbf{y}^{(i)})$

11. Algorithmic Stability and Low-Rank Structure

and assume that $|y_i| \geq 1$ for all $i \in \{1, \dots, n\}$. Then, assuming that the sample size satisfies $n \geq (2 \cdot M \cdot a \cdot B^L)^{\frac{1}{\alpha}}$, for any $\gamma \leq 1/(2 \cdot B^{L-k+1})$, and $\mu \geq B^{k-1}$,

$$\gamma \leq |f_k(\mathbf{x}_i; \hat{\theta}) \cdot y_i| \leq \mu \quad (\forall i \in \{1, \dots, n\}).$$

Proof of Theorem 11.5. In the following let $(\mathbf{x}, y)$ be the point replacing the i th example in the training set $(\mathbf{X}, \mathbf{y})$ and so by interpolation we have $y = \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)})$.

By Lemma 11.6 there exist $C_{\hat{\theta}}, C_{\hat{\theta}^{(i)}}$ independent from the input and $b(\cdot; \hat{\theta}), b(\cdot; \hat{\theta}^{(i)})$ such that

$$\mathcal{N}(\mathbf{x}; \hat{\theta}) + b(\mathbf{x}; \hat{\theta}) \cdot \epsilon = C_{\hat{\theta}} \cdot f_k(\mathbf{x}; \hat{\theta}), \quad (11.5)$$

$$\mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)}) + b(\mathbf{x}; \hat{\theta}^{(i)}) \cdot \epsilon = C_{\hat{\theta}^{(i)}} \cdot f_k(\mathbf{x}; \hat{\theta}^{(i)}). \quad (11.6)$$

Observe that,

$$\begin{aligned} & \left| \mathcal{N}(\mathbf{x}; \hat{\theta}) - \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)}) \right| - \left| (b(\mathbf{x}; \hat{\theta}) - b(\mathbf{x}; \hat{\theta}^{(i)})) \cdot \epsilon \right| \\ & \leq \left| \mathcal{N}(\mathbf{x}; \hat{\theta}) - \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)}) + (b(\mathbf{x}; \hat{\theta}) - b(\mathbf{x}; \hat{\theta}^{(i)})) \cdot \epsilon \right| \\ & \leq \left| C_{\hat{\theta}} \cdot f_k(\mathbf{x}; \hat{\theta}) - C_{\hat{\theta}} \cdot f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right| \\ & \quad + \left| C_{\hat{\theta}} \cdot f_k(\mathbf{x}; \hat{\theta}^{(i)}) - C_{\hat{\theta}^{(i)}} \cdot f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right| \\ & \leq |C_{\hat{\theta}}| \cdot \left| f_k(\mathbf{x}; \hat{\theta}) - f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right| \\ & \quad + \left| f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right| \cdot |C_{\hat{\theta}} - C_{\hat{\theta}^{(i)}}| \\ & \leq |C_{\hat{\theta}}| \cdot \epsilon + \left| f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right| \cdot |C_{\hat{\theta}} - C_{\hat{\theta}^{(i)}}| \quad (\text{By sub-network stability assumption}) \\ & \leq |C_{\hat{\theta}}| \cdot \epsilon + \mu \cdot |C_{\hat{\theta}} - C_{\hat{\theta}^{(i)}}|. \quad (\text{By Lemma 11.7}) \end{aligned}$$

First note that by Lemma 11.6,

$$|b(\mathbf{x}; \hat{\theta})| \leq a \prod_{j=1}^L \|\hat{\mathbf{W}}_j\|_2 \leq a B^L =: b$$

where $\|\hat{\mathbf{W}}_j\|_2 \leq B$ comes as in the first part of the proof of Lemma 11.7. Similarly, $|b(\mathbf{x}; \hat{\theta}^{(i)})| \leq b$. Then,

$$\left| (b(\mathbf{x}; \hat{\theta}) - b(\mathbf{x}; \hat{\theta}^{(i)})) \cdot \epsilon \right| \leq 2 a \epsilon B^L.$$

Now, the idea is to use Equation (11.5) to control $|C_{\hat{\theta}}|$ and to control $|C_{\hat{\theta}} - C_{\hat{\theta}^{(i)}}|$ in terms of difference of sub-network predictions and invoke sub-network stability once again. W.l.o.g., let $i \neq 1$ and so

$$\begin{aligned} C_{\hat{\theta}} &= \frac{\mathcal{N}(\mathbf{x}_1; \hat{\theta}) + b(\mathbf{x}_1; \hat{\theta}) \cdot \epsilon}{f_k(\mathbf{x}_1; \hat{\theta})} = \frac{y_1 + b(\mathbf{x}_1; \hat{\theta}) \cdot \epsilon}{f_k(\mathbf{x}_1; \hat{\theta})}, \\ C_{\hat{\theta}^{(i)}} &= \frac{\mathcal{N}(\mathbf{x}_1; \hat{\theta}^{(i)}) + b(\mathbf{x}_1; \hat{\theta}^{(i)}) \cdot \epsilon}{f_k(\mathbf{x}_1; \hat{\theta}^{(i)})} = \frac{y_1 + b(\mathbf{x}_1; \hat{\theta}^{(i)}) \cdot \epsilon}{f_k(\mathbf{x}_1; \hat{\theta}^{(i)})}. \end{aligned}$$

11. Algorithmic Stability and Low-Rank Structure

So, by Lemma 11.6 and Lemma 11.7 (with γ defined therein),

$$|C_{\hat{\theta}}| \leq \frac{|y_1| + b \cdot \epsilon}{|f_k(\mathbf{x}_1; \hat{\theta})|} \leq \frac{|y_1| + b \cdot \epsilon}{\gamma}.$$

Now we turn our attention to the gap:

$$\begin{aligned} |C_{\hat{\theta}} - C_{\hat{\theta}^{(i)}}| &= \left| \frac{y_1 + b(\mathbf{x}_1; \hat{\theta}) \cdot \epsilon}{f_k(\mathbf{x}_1; \hat{\theta})} - \frac{y_1 + b(\mathbf{x}_1; \hat{\theta}^{(i)}) \cdot \epsilon}{f_k(\mathbf{x}_1; \hat{\theta}^{(i)})} \right| \\ &\leq |y_1| \left| \frac{f_k(\mathbf{x}_1; \hat{\theta}) - f_k(\mathbf{x}_1; \hat{\theta}^{(i)})}{f_k(\mathbf{x}_1; \hat{\theta}) \cdot f_k(\mathbf{x}_1; \hat{\theta}^{(i)})} \right| \\ &\quad + \left| \frac{b(\mathbf{x}_1; \hat{\theta})}{f_k(\mathbf{x}_1; \hat{\theta})} - \frac{b(\mathbf{x}_1; \hat{\theta}^{(i)})}{f_k(\mathbf{x}_1; \hat{\theta}^{(i)})} \right| \cdot \epsilon \\ &\stackrel{(a)}{\leq} \frac{|y_1| \cdot \epsilon}{|f_k(\mathbf{x}_1; \hat{\theta}) \cdot f_k(\mathbf{x}_1; \hat{\theta}^{(i)})|} \\ &\quad + \frac{b \cdot \epsilon}{|f_k(\mathbf{x}_1; \hat{\theta}) \cdot f_k(\mathbf{x}_1; \hat{\theta}^{(i)})|} \\ &\stackrel{(b)}{\leq} \frac{2(|y_1| + b) \epsilon}{(\gamma - \epsilon)_+^2 + \gamma^2 - \epsilon^2} \end{aligned}$$

where in step (a) we used the assumption that sub-network is ϵ -stable and Lemma 11.6, while step (b) amounts to lower-bounding the denominator, and deferred till the end of the proof.

Putting all together we have

$$\begin{aligned} |\mathcal{N}(\mathbf{x}; \hat{\theta}) - \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)})| &\leq 2 \cdot b \cdot \epsilon + \frac{1 + b \cdot \epsilon}{\gamma} \cdot \epsilon \\ &\quad + \mu \cdot \frac{2(1 + b) \epsilon}{(\gamma - \epsilon)_+^2 + \gamma^2 - \epsilon^2}. \end{aligned}$$

Now, let's require $\epsilon \leq \gamma/2$: By assumption we have that $\epsilon = (M/n)^\alpha$ and so our requirement is equivalent to $n \geq ((2M)/\gamma)^{\frac{1}{\alpha}}$. In overall, this gives

$$\begin{aligned} |\mathcal{N}(\mathbf{x}; \hat{\theta}) - \mathcal{N}(\mathbf{x}; \hat{\theta}^{(i)})| &\leq (2b + 1)\epsilon + \frac{b}{\gamma} \epsilon^2 \\ &\quad + 2(1 + b) \frac{\mu}{\gamma^2} \epsilon. \end{aligned}$$

We conclude by using the bounds on γ and μ in Lemma 11.7, and the one on b which comes by Lemma 11.6.

Proof of step (b) By Lemma 11.7 we have that $|f_k(\mathbf{x}; \hat{\theta}^{(i)})| \geq \gamma$. Now, given the above, the fact that the sub-network is ϵ -stable, and triangle inequality we have

$$\begin{aligned} \epsilon &\geq |f_k(\mathbf{x}; \hat{\theta}) - f_k(\mathbf{x}; \hat{\theta}^{(i)})| \\ &\geq |f_k(\mathbf{x}; \hat{\theta}^{(i)})| - |f_k(\mathbf{x}; \hat{\theta})| \geq \gamma - |f_k(\mathbf{x}; \hat{\theta})|. \end{aligned}$$

This gives

$$\begin{aligned}
 & \left(f_k(\mathbf{x}; \hat{\theta}) - f_k(\mathbf{x}; \hat{\theta}^{(i)}) \right)^2 \leq \epsilon^2 \\
 \iff & f_k(\mathbf{x}; \hat{\theta})^2 + f_k(\mathbf{x}; \hat{\theta}^{(i)})^2 - \epsilon^2 \\
 & \leq 2 f_k(\mathbf{x}; \hat{\theta}) \cdot f_k(\mathbf{x}; \hat{\theta}^{(i)}) \\
 \implies & \frac{(\gamma - \epsilon)_+^2 + \gamma^2 - \epsilon^2}{2} \leq f_k(\mathbf{x}; \hat{\theta}) \cdot f_k(\mathbf{x}; \hat{\theta}^{(i)}) .
 \end{aligned}$$

□

11.2. Technical proofs

In this section, we detail the proofs of our technical lemmas.

11.2.1. Minimum-norm interpolants and low-rank bias

Proof of Lemma 11.3 Invoking Theorem 4 in (Timor et al. 2023), we have that $\hat{\theta}$ obtained by algorithm $\mathcal{T}$ for architecture $\mathbb{A} = \langle L, d, d_0 \rangle$ verifies:

$$\frac{L}{\sum_{k=1}^L (\mathbf{S}(\hat{\mathbf{W}}_k))}^{-1} \leq B^{\frac{L^*}{L}}$$

thus

$$\frac{1}{L} \sum_{k=1}^L (\mathbf{S}(\hat{\mathbf{W}}_k))^{-1} \geq \frac{1}{B^{\frac{L^*}{L}}}$$

Since the quantity on the left is an average, there must exist $1 \leq k \leq L$ such that

$$(\mathbf{S}(\hat{\mathbf{W}}_k))^{-1} \geq \frac{1}{B^{\frac{L^*}{L}}} \tag{11.7}$$

thus

$$\mathbf{S}(\hat{\mathbf{W}}_k) \leq B^{\frac{L^*}{L}} .$$

By setting $L \geq L^* \frac{\log(B)}{\log(1+a \cdot \epsilon)}$, we ensure that:

$$\mathbf{S}(\hat{\mathbf{W}}_k) \leq 1 + a \cdot \epsilon .$$

We note $\hat{\mathbf{W}}_k = \lambda_k \mathbf{u}_k \mathbf{v}_k^T + \hat{\mathbf{W}}_k^\epsilon$ its decomposition following Definition 11.2.

11.2.2. Low-rank layers preserve learned signals

Proof of Lemma 11.6 Throughout the proof we drop dependence on θ , e.g. $\mathcal{N}(\mathbf{x}) \equiv \mathcal{N}(\mathbf{x}; \theta)$.

Introduce

$$b(\mathbf{x}) := \frac{1}{\epsilon} \cdot \left(\mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) - \mathcal{N}(\mathbf{x}) \right)$$

and we can thus write:

$$\mathcal{N}(\mathbf{x}) + b(\mathbf{x}) \cdot \epsilon = \mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) .$$

Showing statement 1. First consider the case when $\mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) > 0$. Now, using the fact that ReLU is positively-homogeneous,

$$\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) = \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \cdot \text{ReLU}(\lambda_k \mathbf{u}_k)$$

Therefore, using positive-homogeneity once again,

$$\begin{aligned} \mathcal{N}(\mathbf{x}) &= \mathcal{N}^{k+1:L} \left(\mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \cdot \text{ReLU}(\lambda_k \mathbf{u}_k) \right) - b(\mathbf{x}) \cdot \epsilon \\ &= \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \cdot \underbrace{\mathcal{N}^{k+1:L}(\text{ReLU}(\lambda_k \mathbf{u}_k))}_{C^+} - b(\mathbf{x}) \cdot \epsilon \end{aligned}$$

where we note that C^+ can take a different sign since $C^+ = \mathbf{W}_L^\top \mathcal{N}^{k+1:L-1}(\text{ReLU}(\lambda_k \mathbf{u}_k))$.

Showing statement 2. Now, considering an alternative case $\mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \leq 0$, positive-homogeneity once again gives

$$\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) = -\mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \cdot \text{ReLU}(-\lambda_k \mathbf{u}_k)$$

and so

$$\begin{aligned} \mathcal{N}(\mathbf{x}) &= \mathcal{N}^{k+1:L} \left(-\mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \cdot \text{ReLU}(-\lambda_k \mathbf{u}_k) \right) - b(\mathbf{x}) \cdot \epsilon \\ &= \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \cdot \underbrace{\left(-\mathcal{N}^{k+1:L}(\text{ReLU}(-\lambda_k \mathbf{u}_k)) \right)}_{C^-} - b(\mathbf{x}) \cdot \epsilon . \end{aligned}$$

Showing statement 3 and 4. It suffices to prove that for any input $\mathbf{x} \in \mathbb{R}^{d_0}$,

$$\left| \mathcal{N}(\mathbf{x}) - \mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) \right| \leq a \epsilon B^L .$$

By 1-Lipschitzness of ReLU, we have:

$$\begin{aligned} &\left\| \text{ReLU} \left(\mathbf{W}_k \mathcal{N}^{1:k-1}(\mathbf{x}) \right) - \text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right\| \\ &\leq \left\| \mathbf{W}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}) \right\| \\ &\leq \left\| \mathbf{W}_k^\epsilon \right\|_F \left\| \mathcal{N}^{1:k-1}(\mathbf{x}) \right\| \\ &\leq a \epsilon \left\| \mathbf{W}_k \right\|_F \left\| \mathcal{N}^{1:k-1}(\mathbf{x}) \right\| . \end{aligned}$$

11. Algorithmic Stability and Low-Rank Structure

where the last inequality comes by the following observation:

$$S(\mathbf{W}_k) \leq 1 + a\epsilon \implies \|\mathbf{W}_k^\epsilon\|_F \leq a\epsilon \|\mathbf{W}_k\|_F.$$

At the same time, note that by 1-Lipschitzness of ReLU, and Cauchy-Schwartz inequality we have that for any $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^{d_k}$,

$$|\mathcal{N}^{k+1:L}(\mathbf{z}) - \mathcal{N}^{k+1:L}(\mathbf{z}')| \leq \|\mathbf{z} - \mathbf{z}'\| \prod_{j=k+1}^L \|\mathbf{W}_j\|_F.$$

Combining the above we have

$$\begin{aligned} & \left| \mathcal{N}(\mathbf{x}) - \mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) \right| \\ &= \left| \mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\mathbf{W}_k \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) - \mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\lambda_k \mathbf{u}_k \mathbf{v}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) \right| \\ &\leq \left(\prod_{j=k}^L \|\mathbf{W}_j\|_F \right) \cdot a \cdot \epsilon \|\mathcal{N}^{1:k-1}(\mathbf{x})\| \\ &\leq \left(\prod_{j=1}^L \|\mathbf{W}_j\|_F \right) \cdot a \cdot \epsilon \end{aligned}$$

where the last inequality comes by Cauchy-Schwartz inequality, realizing that $\text{ReLU}(|x|) = |x|$, and the fact that $\|\mathbf{x}\| \leq 1$. $\square$

11.2.3. Minimum-norm interpolants and margins

Proof of Lemma 11.7 We will require the following:

Lemma 11.8 (Timor et al. (2023), Lemma 14). *Let $\hat{\theta} = \mathcal{T}(\mathbf{X}, \mathbf{y})$. Then, for every $1 \leq i < j \leq L$ we have $\|\hat{\mathbf{W}}_i\|_F = \|\hat{\mathbf{W}}_j\|_F$.*

First, notice that $\mathcal{N}(\cdot, \hat{\theta})$ is a minimum-norm interpolant and so Lemma 11.8, the weight matrices $\hat{\mathbf{W}}_\ell$ have the same norm and thus verify:

$$\|\hat{\mathbf{W}}_\ell\|_F^2 = \frac{1}{L} \sum_{\ell=1}^L \|\hat{\mathbf{W}}_\ell\|_F^2 \leq \frac{1}{L} \sum_{\ell=1}^L \|\mathbf{W}_\ell^*\|_F^2 \leq \frac{1}{L} \cdot L \cdot B^2$$

which gives us $\|\hat{\mathbf{W}}_\ell\|_F \leq B$.

Proof of a lower bound. Using the fact that $\mathcal{N}(\mathbf{x}_i; \hat{\theta}) = y_i$ and assumption that $|y_i| \geq 1$ for each i , we have:

$$\begin{aligned} 1 \leq |y_i| &= |\mathcal{N}(\mathbf{x}_i; \hat{\theta})| \\ &= \left| \mathcal{N}^{k+1:L} \left(\text{ReLU} \left(\hat{\lambda}_k \hat{\mathbf{u}}_k \hat{\mathbf{v}}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) + \hat{\mathbf{W}}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right) \right| \\ &\stackrel{(a)}{\leq} \left| \text{ReLU} \left(\hat{\lambda}_k \hat{\mathbf{u}}_k \hat{\mathbf{v}}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}) + \hat{\mathbf{W}}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}) \right) \right| \prod_{\ell=k+1}^L \|\hat{\mathbf{W}}_\ell\|_2 \\ &\stackrel{(b)}{\leq} \left(\left| \hat{\mathbf{v}}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right| \|\hat{\mathbf{u}}_k\| |\hat{\lambda}_k| + \left\| \hat{\mathbf{W}}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right\| \right) \prod_{\ell=k+1}^L \|\hat{\mathbf{W}}_\ell\|_F \end{aligned}$$

11. Algorithmic Stability and Low-Rank Structure

where steps (a), (b) comes by the Cauchy-Schwartz inequality and the fact that $\text{ReLU}(|x|) = |x|$.

Thus,

$$\frac{1}{B^{L-k}} \leq \left| \hat{\mathbf{v}}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right| \|\hat{\mathbf{u}}_k\| |\hat{\lambda}_k| + \left\| \hat{\mathbf{W}}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right\| \quad (11.8)$$

Furthermore by Lemma 11.6,

$$\left\| \hat{\mathbf{W}}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right\| \leq a \epsilon \prod_{j=1}^k \|\mathbf{W}_k^\epsilon\|_2 \leq a \epsilon B^k.$$

By assumption of the Lemma we have $n \geq (2 M a B^L)^{\frac{1}{\alpha}}$, and therefore $\epsilon = \frac{M}{n^\alpha} \leq \frac{1}{2 \cdot a \cdot B^L}$, which gives us that:

$$\left\| \hat{\mathbf{W}}_k^\epsilon \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right\| \leq \frac{1}{2 B^{L-k}}.$$

Plugged into Equation (11.8), and using the fact that $|\hat{\lambda}_k| \leq \|\hat{\mathbf{W}}_k\|_2 \leq \|\hat{\mathbf{W}}_k\|_F \leq B$ and using that $\|\hat{\mathbf{u}}_k\| = 1$, we have:

$$\left| \hat{\mathbf{v}}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right| \geq \frac{1}{2 B^{L-k+1}}.$$

Proof of an upper bound. Similarly, using the Cauchy-Schwartz inequality,

$$|f_k(\mathbf{x}; \hat{\theta})| = \left| \hat{\mathbf{v}}_k^\top \mathcal{N}^{1:k-1}(\mathbf{x}_i; \hat{\theta}) \right| \leq \|\hat{\mathbf{v}}_k\| \|\mathbf{x}_i\| \prod_{\ell=1}^{k-1} \|\hat{\mathbf{W}}_\ell\|_F \leq B^{k-1}.$$

□

11.3. Concluding remarks of the chapter

Algorithmic stability The stability of interpolating kernel least-squares has been studied by (Rangamani et al. 2023) who established a connection to stability of the pseudo-inverse, which is indeed a minimum-norm interpolation, and which is in turn controlled by the smallest non-zero eigenvalue of the kernel matrix. In this chapter we looked into a rather different setting of interpolation with neural networks rather than kernel machines, and provided sufficient conditions for such stability.

Recently, (Schliserman & Koren 2022) analyzed the performance of gradient descent type methods on convex learning problems given linearly separable data. The setting of the chapter can be interpreted as noise-free labels and so in case of classification, we have separability as well, however the decision boundary can be non-linear (as we can conclude from Assumption 11.1).

Neural collapse Recently, a phenomenon called neural collapse (NC) (Papayan et al. 2020; Han et al. 2021) has been observed where the post-activations (and weights of a final layer) of a well-trained neural network, appear to be clustered in a low-dimensional subspace. Theoretical studies on the cause of NC mostly rely on unconstrained feature models (Fang et al. 2021; Mixon et al. 2022) where the product of two matrices is trained under gradient flow: the two matrices train to a low-rank structure where their ranks equal the dimension of the outputs.

Lottery ticket hypothesis The existence of a small good neural network within a large deep neural network has been proposed before as a prominent hypothesis in neural network learning. In particular the *lottery ticket hypothesis* (Frankle & Carbin 2018) posits that deep neural networks contain small sub-networks which could be trained in isolation and lead to comparable performance, and that these sub-networks happen to be favored by standard initializations. The *lottery ticket hypothesis* has however only been analyzed in some restricted settings (Frankle et al. 2020; Malach et al. 2020; Orseau et al. 2020; Sakamoto & Sato 2022). It is still an active area of theoretical research. In this chapter we explored a related concept, where the quality of the sub-network is captured by its stability, which establishes a link to analysis of the generalisation error.

Benign overfitting. One prominent line of recent work that tries to explain the success of deep learning is *benign overfitting* (Bartlett et al. 2020; Koehler et al. 2021). The idea behind benign overfitting is that even in the presence of low-to-moderate noise, interpolating predictors are able to achieve performance that is close to the noise rate (in other words, some inductive bias is present in the interpolation procedure). In particular, these works study this phenomenon through analysis of the excess risk of interpolants (such as linear interpolants in high dimension). The key idea behind these analyses is that *minimum-norm* interpolation is a sufficient condition for benign overfitting. On the other hand, it is also well-known that uniform stability is sufficient for uniform convergence and so one can show that the excess risk has correct asymptotic behavior (convergence to the noise rate) (Shalev-Shwartz et al. 2010).

These theories appear to be at odds in a general sense. However, minimal-norm interpolation can still be uniformly stable in a restricted sense, for instance, when the problem is well-conditioned (on the subspace) (Rangamani et al. 2023).

In this work, we argue that the existence of a uniformly stable subnetwork is sufficient for the stability of the entire neural network. However, such a subnetwork might not be uniformly stable everywhere, but only on some ‘nice’ problems (such as separable classification problems). A deeper understanding of this is left to determining whether and when the subnetwork is stable.

Limitations and outlook. In this work, we have studied sufficient conditions for algorithmic stability in minimum-norm interpolating deep ReLU neural networks. This study opens up several interesting avenues for future research. One of the sufficient conditions we examined is the existence of a stable sub-network, which we did not prove theoretically and which remains an open question. In related areas, such as the lottery ticket hypothesis, the existence of a well-performing sub-network has been

shown theoretically, which might inspire the use of similar techniques to prove the existence of a stable sub-network. Finally, an interesting open question is bridging the gap between the stability of minimum-norm interpolation in neural networks under GF and actual optimisation algorithms, such as SGD. While there are several promising directions, a complete picture incorporating stability analysis might require additional arguments (Poggio et al. 2020; Elkabetz & Cohen 2021).

One possible limitation of our analysis is that the stability bound involves a B -dependent factor which scales as B^{2L-k+1} in the best case. For the case $k = L - 1$, it might be possible to achieve a better factor B^L which would be in line with Rademacher complexity analysis (Golowich et al. 2018).

Even though we empirically validate that our assumptions hold for bias-free FCNs trained on some datasets, whether the condition holds in larger architectures trained on more complex datasets requires more empirical exploration. In the future, we aim to extend our results to more complex scenarios and empirically explore the limit in which deep neural networks satisfy our assumptions.

12. Conclusion

This thesis has explored two fundamental and deeply intertwined questions in the theory of neural networks: *explainability* and *generalisation*. While modern deep learning systems achieve remarkable empirical success across a wide range of domains, their predictions remain difficult to interpret and their ability to generalise continues to defy classical statistical intuition. By combining tools from logic, complexity theory, kernel methods, Bayesian statistics, random matrix theory, and algorithmic learning theory, this work has advanced the state-of-the-art on both challenges¹.

12.1. Summary of Contributions

On *explainability*, the thesis has contributed to rigorous foundations for cardinality-minimal explanations in neural networks, and has developed a novel logic-based formalism to study the expressiveness of neural architectures:

- We studied the *Minimum Change Required* and *Minimum Sufficient Reason* problems and strengthened intractability results, clarifying the computational limits of exact explanations.
- For practically relevant regimes, monotonic networks with (almost everywhere) smooth activations, we developed efficient algorithms based on greedy search that yield tractable and provably correct explanations.
- We connected minimal explanations to attribution methods via Aumann–Shapley values, situating them within a broader landscape of interpretability tools.
- Beyond individual decisions, we introduced a framework based on the *fibring of modal logics* to combine neural networks, demonstrating non-uniform expressivity correspondences for graph neural networks and transformers, and laying the foundation for developing methodologies to reverse-engineer logical theories from trained architectures.

On *generalisation*, the thesis developed new insights by analysing the spectral properties of the empirical kernel, and by investigating algorithmic stability in connection with training-induced structure:

- We proposed the Mercer spectrum of the empirical kernel as a key mathematical object to study generalisation and we characterised it using random matrix theory.
- We explained the *double-descent phenomenon* using Gaussian process limits, showing how discrepancies between finite-width random kernels and their infinite-

¹Reproducibility statement: For theoretical contributions, all results are accompanied by complete proofs which are included in the main text of this thesis; for experimental validations, the code developed to perform all experiments included in this thesis has been open-sourced as part of the corresponding peer-review publications.

12. Conclusion

width limits produce the interpolation peak and the different regimes.

- We extended the *renormalisation theory* of deep neural networks to new regimes using a spectral description of Bayesian training.
- We identified adversarial examples in interpolating regression settings with the presence of *near-zero eigenvalues* in the empirical kernel, helping reconcile strong generalisation performance with adversarial vulnerability.
- We proved sufficient conditions for the algorithmic stability of minimum-norm interpolating deep ReLU networks, linking generalisation to emergent phenomena such as low-rank structure.

Taken together, these results contribute to different formalisms for interpreting and understanding generalisation of modern neural networks, and combine them in new ways to yield novel insights. Moreover, the thesis highlights the deep connections between these two fronts: understanding minimal explanations requires reasoning about robustness, which is intimately tied to generalisation; conversely, explaining generalisation often relies on structures (low-rank, stable, interpretable features) that can serve explainability.

12.2. Future Research Directions

The results presented here open several avenues for further research:

- **Scalability of explanations.** While tractable for restricted classes, minimal explanations remain intractable in general. Bridging the gap between theory and practice requires approximation schemes and heuristics with provable guarantees that scale to foundation models.
- **From local to global interpretability.** Extending the logical fibering framework to produce global theories of model behaviour for large-scale architectures remains a challenging but promising direction.
- **Unifying perspectives.** This thesis has shown that explainability and generalisation are not orthogonal but deeply connected. Developing unified frameworks that jointly account for both could yield principled methods for building safer and more interpretable AI systems.
- **Applications to foundation models.** The safety concerns of large language and diffusion models (e.g. jailbreaks, prompt injection) demand theoretical insights at scale. Extending the methods of this thesis to these architectures is an urgent and impactful research direction.

Closing Remarks

Much remains to be understood about modern neural-network-based systems: universal approximators trained in high-dimensional, nonconvex landscapes, yet consistently capable of generalising and producing useful outputs for the world. This thesis has taken several steps toward demystifying these capabilities by providing new theoretical

12. Conclusion

tools for both explainability and generalisation. The interplay between these two questions—why neural networks generalise and how to explain their predictions—offers a fertile ground for future advances in the theory and practice of deep learning.

Bibliography

- Adebayo, J., J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt & B. Kim (2018). “Sanity Checks for Saliency Maps”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Adlam, B. & J. Pennington (2020). “Understanding Double Descent Requires A Fine-Grained Bias-Variance Decomposition”. In: *Proc. of NeurIPS*.
- Advani, M., A. Saxe & H. Sompolinsky (2020). “High-dimensional dynamics of generalization error in neural networks”. In: *Neural networks : the official journal of the International Neural Network Society* 132, pp. 428–446. DOI: [10.1016/j.neunet.2020.08.022](https://doi.org/10.1016/j.neunet.2020.08.022).
- Aitchison, L. (2020). “Why Bigger Is Not Always Better: On Finite and Infinite Neural Networks”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by H. Daumé III & A. Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 156–164. URL: <https://proceedings.mlr.press/v119/aitchison20a.html>.
- Aitchison, L., A. Yang & S. W. Ober (2021). “Deep Kernel Processes”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by M. Meila & T. Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 130–140. URL: <https://proceedings.mlr.press/v139/aitchison21a.html>.
- Ameisen, E., J. Lindsey, A. Pearce, W. Gurnee, N. L. Turner, B. Chen, C. Citro, D. Abrahams, S. Carter, B. Hosmer, J. Marcus, M. Sklar, A. Templeton, T. Bricken, C. McDougall, H. Cunningham, T. Henighan, A. Jermyn, A. Jones, A. Persic, Z. Qi, T. B. Thompson, S. Zimmerman, K. Rivoire, T. Conerly, C. Olah & J. Batson (2025). *Circuit Tracing: Revealing Computational Graphs in Language Models*. Anthropic / Transformer-Circuits Blog preprint. URL: <https://transformer-circuits.pub/2025/attribution-graphs/methods.html>.
- Ancona, M., E. Ceolini, C. Öztireli & M. Gross (2018). “Towards Better Understanding of Gradient-Based Attribution Methods for Deep Neural Networks”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=Sy21R9JAW>.
- Angelino, E., N. Larus-Stone, D. Alabi, M. Seltzer & C. Rudin (2017). “Learning Certifiably Optimal Rule Lists”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 35–44. DOI: [10.1145/3097983.3098047](https://doi.org/10.1145/3097983.3098047).
- Anthropic (2023). “Towards Monosemanticity: Decomposing Language Models With Dictionary Learning”. In: *Transformer Circuits Thread*. URL: <https://transformer-circuits.pub/2023/monosemantic-features/index.html>.
- Anthropic (2024). “Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet”. In: *Transformer Circuits Thread*. URL: <https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html>.
- Aumann, R. J. & L. Shapley (1974). *Values of Non-Atomic Games*. Princeton University Press.

Bibliography

- Bach, S., A. Binder, G. Montavon, F. Klauschen, K.-R. Müller & W. Samek (2015). “On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation”. In: *PLOS ONE* 10.7, e0130140. DOI: [10.1371/journal.pone.0130140](https://doi.org/10.1371/journal.pone.0130140).
- Baker, C. T. H. (1977). *The Numerical Treatment of Integral Equations*. Oxford University Press.
- Banna, M., F. Merlevede & M. Peligrad (2015). “On the limiting spectral distribution for a large class of symmetric random matrices with correlated entries”. In: *Stochastic Processes and their Applications* 125.
- Barbier, J., F. Krzakala, N. Macris, L. Miolane & L. Zdeborova (2018). “Optimal Errors and Phase Transitions in High-Dimensional Generalized Linear Models”. In: *Proceedings of the 31st Conference On Learning Theory*. Ed. by S. Bubeck, V. Perchet & P. Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, pp. 728–731. URL: <https://proceedings.mlr.press/v75/barbier18a.html>.
- Barcelo, P., A. Kozachinskiy, A. W. Lin & V. Podolskii (2024). “Logical Languages Accepted by Transformer Encoders with Hard Attention”. In: *The Twelfth International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=gbrHZq07mq>.
- Barceló, P., E. V. Kostylev, M. Monet, J. Pérez, J. L. Reutter & J. P. Silva (2020a). “The Logical Expressiveness of Graph Neural Networks”. In: *8th International Conference on Learning Representations (ICLR)*.
- Barceló, P., M. Monet, J. Pérez & B. Subercaseaux (2020b). “Model Interpretability through the lens of Computational Complexity”. In: *Proc. of NeurIPS*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan & H. Lin.
- Bartlett, P. L. & S. Mendelson (2002). “Rademacher and Gaussian complexities: Risk bounds and structural results”. In: *JMLR* 3.Nov, pp. 463–482.
- Bartlett, P. L., D. J. Foster & M. Telgarsky (2017). “Spectrally-Normalized Margin Bounds for Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 30, pp. 6240–6249.
- Bartlett, P. L., P. M. Long, G. Lugosi & A. Tsigler (2020). “Benign Overfitting in Linear Regression”. In: *Proceedings of the National Academy of Sciences* 117.48, pp. 30063–30070. DOI: [10.1073/pnas.1907378117](https://doi.org/10.1073/pnas.1907378117).
- Bau, D., J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou & A. Torralba (2020). “Understanding the Role of Individual Units in a Deep Neural Network”. In: *Proceedings of the National Academy of Sciences* 117.48, pp. 30071–30078. DOI: [10.1073/pnas.1907375117](https://doi.org/10.1073/pnas.1907375117).
- Belkin, M., D. Hsu & J. Xu (2020). “Two Models of Double Descent for Weak Features”. In: *SIAM Journal on Mathematics of Data Science* 2.4, pp. 1167–1180.
- Belkin, M. (2021). “Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation”. In: *arXiv preprint arXiv:2105.14368*.
- Belkin, M., D. J. Hsu, S. Ma & S. Mandal (2019). “Reconciling modern machine-learning practice and the classical bias–variance tradeoff”. In: *Proceedings of the National Academy of Sciences* 116.32, pp. 15849–15854.
- Benedikt, M., C.-H. Lu, B. Motik & T. Tan (2024). “Decidability of Graph Neural Networks via Logical Characterizations”. In: *51st International Colloquium on Automata, Languages, and Programming (ICALP)*. Vol. 297. LIPIcs. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 127:1–127:20. DOI: [10.4230/LIPIcs.ICALP.2024.127](https://doi.org/10.4230/LIPIcs.ICALP.2024.127).
- Bhagoji, A. N., D. Cullina & P. Mittal (2019). “Lower Bounds on Adversarial Robustness from Optimal Transport”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox & R. Garnett.

Bibliography

- Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/02bf86214e264535e3412283e817deaa-Paper.pdf.
- Bingham, E., J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall & N. D. Goodman (2019). “Pyro: Deep Universal Probabilistic Programming”. In: *Journal of Machine Learning Research* 20.28, pp. 1–6. URL: <http://jmlr.org/papers/v20/18-403.html>.
- Blanc, G., J. Lange & L.-Y. Tan (2021). “Provably Efficient, Succinct, and Precise Explanations”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 34, pp. 6129–6141.
- Blei, D. M., A. Kucukelbir & J. D. McAuliffe (2017). “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518, pp. 859–877. DOI: [10.1080/01621459.2017.1285773](https://doi.org/10.1080/01621459.2017.1285773). URL: <https://doi.org/10.1080/2F01621459.2017.1285773>.
- Blum, A. & R. Rivest (1988). “Training a 3-Node Neural Network is NP-Complete”. In: *Advances in Neural Information Processing Systems*. Vol. 1. URL: <https://proceedings.neurips.cc/paper/1988/file/3def184ad8f4755ff269862ea77393dd-Paper.pdf>.
- Bordelon, B., A. Canatar & C. Pehlevan (2020). “Spectrum Dependent Learning Curves in Kernel Regression and Wide Neural Networks”. In: *Proc. PMLR*, pp. 1024–1034.
- Bosch, D., A. Panahi, A. Özcelikkale & D. Dubhash (2022). “Double Descent in Random Feature Models: Precise Asymptotic Analysis for General Convex Regularization”. In: *arXiv*. URL: <https://arxiv.org/abs/2204.02678>.
- Boucheron, S., G. Lugosi & P. Massart (2013). *Concentration inequalities: A nonasymptotic theory of independence*. Oxford University Press.
- Bousquet, O., S. Boucheron & G. Lugosi (2004). “Introduction to Statistical Learning Theory”. In: *Advanced Lectures on Machine Learning*. Vol. 3176. Lecture Notes in Computer Science. Springer, pp. 169–207.
- Bousquet, O. & A. Elisseeff (2002). “Stability and generalization”. In: *Journal of Machine Learning Research* 2, pp. 499–526.
- Bousquet, O., Y. Klochkov & N. Zhivotovskiy (2020). “Sharper bounds for uniformly stable algorithms”. In: *COLT*.
- Bronstein, M. M., J. Bruna, T. Cohen & P. Velickovic (2021). “Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges”. In: *CoRR* abs/2104.13478. arXiv: [2104.13478](https://arxiv.org/abs/2104.13478). URL: <https://arxiv.org/abs/2104.13478>.
- Brown, T. B., B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, et al. (2020). “Language Models are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33, pp. 1877–1901.
- Bubeck, S., V. Chandrasekaran, R. Eldan, et al. (2023). *Sparks of Artificial General Intelligence: Early experiments with GPT-4*. arXiv: [2303.12712](https://arxiv.org/abs/2303.12712) [cs.CL].
- Buza, K. (2014). “Feedback Prediction for Blogs”. In: *Data Analysis, Machine Learning and Knowledge Discovery*. Springer International Publishing, pp. 145–152.
- Canatar, A., B. Bordelon & C. Pehlevan (2021). “Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks”. In: *Nature Communications* 12.1. DOI: [10.1038/s41467-021-23103-1](https://doi.org/10.1038/s41467-021-23103-1). URL: <https://doi.org/10.1038/s41467-021-23103-1>.
- Cano, J. R., P. A. Gutiérrez, B. Krawczyk, M. Wozniak & S. García (2019). “Monotonic classification: An overview on algorithms, performance measures and data sets”. In: *Neurocomputing* 341, pp. 168–182. DOI: [10.1016/j.neucom.2019.02.024](https://doi.org/10.1016/j.neucom.2019.02.024). URL: <https://doi.org/10.1016/j.neucom.2019.02.024>.

Bibliography

- Carlini, N., A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry & A. Kurakin (2019). “On Evaluating Adversarial Robustness”. In: *arXiv*. DOI: [arXiv:1902.06705](https://arxiv.org/abs/1902.06705).
- Charles, Z. & D. Papailiopoulos (2018). “Stability and generalization of learning algorithms that converge to global optima”. In: *ICML*.
- Chen, C., O. Li, C. Tao, A. J. Barnett, J. Su & C. Rudin (2019). “This Looks Like That: Deep Learning for Interpretable Image Recognition”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Chen, Z. & H. Schaeffer (2021). “Conditioning of Random Feature Matrices: Double Descent and Generalization Error”. In: *arXiv:2110.11477*.
- Cheng, X. & A. Singer (2013). “The Spectrum Of Random Inner-Product Kernel Matrices”. In: *Random Matrices: Theory and Applications* 02.04, p. 1350010. DOI: [10.1142/S201032631350010X](https://doi.org/10.1142/S201032631350010X). URL: <https://doi.org/10.1142/S201032631350010X>.
- Cho, K., B. van Merriënboer, C. Gulcehre, D. Bahdanau, et al. (2014). “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: *Proceedings of EMNLP*, pp. 1724–1734.
- Cho, Y. & L. Saul (2009). “Kernel methods for deep learning”. In: *Advances in neural information processing systems* 22, pp. 342–350.
- Cohen, J., S. Kaur, Y. Li, J. Z. Kolter & A. Talwalkar (2021). “Gradient Descent on Neural Networks Typically Occurs at the Edge of Stability”. In: *International Conference on Learning Representations (ICLR)*.
- Cohen, J. M., E. Rosenfeld & J. Z. Kolter (2019). “Certified Adversarial Robustness via Randomized Smoothing”. In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 1310–1320. URL: <https://proceedings.mlr.press/v97/cohen19c.html>.
- Craven, M. & J. W. Shavlik (1996). “Extracting Tree-Structured Representations of Trained Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 8, pp. 24–30.
- Criscitiello, C. & N. Boumal (2019). “Efficiently escaping saddle points on manifolds”. In: *NeurIPS* 32.
- Croce, F. & M. Hein (2020). “Minimally distorted Adversarial Examples with a Fast Adaptive Boundary Attack”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by H. D. III & A. Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 2196–2205. URL: <https://proceedings.mlr.press/v119/croce20a.html>.
- Cucala, D. J. T., B. Cuenca Grau, E. V. Kostylev & B. Motik (2022a). “Explainable GNN-Based Models over Knowledge Graphs”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=CrCvGNHAIrz>.
- Cucala, D. J. T., B. C. Grau & B. Motik (2022b). “Faithful Approaches to Rule Learning”. In: *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning (KR)*. URL: <https://proceedings.kr.org/2022/50/kr2022-0050-tena-cucala-et-al.pdf>.
- Cuenca Grau, B., E. Feng & P. A. Walega (2025). *The Correspondence Between Bounded Graph Neural Networks and Fragments of First-Order Logic*. arXiv: [2505.08021](https://arxiv.org/abs/2505.08021) [cs.AI]. URL: <https://arxiv.org/abs/2505.08021>.
- Cui, H., F. Krzakala & L. Zdeborova (2023). “Bayes-optimal Learning of Deep Random Networks of Extensive-width”. In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato &

Bibliography

- J. Scarlett. Vol. 202. *Proceedings of Machine Learning Research*. PMLR, pp. 6468–6521. URL: <https://proceedings.mlr.press/v202/cui23b.html>.
- Cui, H., B. Loureiro, F. Krzakala & L. Zdeborová (2022). “Generalization error rates in kernel regression: the crossover from the noiseless to noisy regime”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2022.11, p. 114004. DOI: [10.1088/1742-5468/ac9829](https://doi.org/10.1088/1742-5468/ac9829). URL: <https://doi.org/10.1088/1742-5468/ac9829>.
- D’Ascoli, S., M. Refinetti, G. Biroli & F. Krzakala (2020). “Double Trouble in Double Descent: Bias and Variance(s) in the Lazy Regime”. In: *ICML*. Vol. 119, pp. 2280–2290. URL: <https://proceedings.mlr.press/v119/d-ascoli20a.html>.
- Dandi, Y., L. Pesce, H. Cui, F. Krzakala, Y. M. Lu & B. Loureiro (2024). *A Random Matrix Theory Perspective on the Spectrum of Learned Features and Asymptotic Generalization Capabilities*. arXiv: [2410.18938](https://arxiv.org/abs/2410.18938) [stat.ML]. URL: <https://arxiv.org/abs/2410.18938>.
- Daniely, A., R. Frostig & Y. Singer (2016). “Toward Deeper Understanding of Neural Networks: The Power of Initialization and a Dual View on Expressivity”. In: *Advances in Neural Information Processing Systems* 29.
- Daniely, A. & H. Shacham (2020). “Most ReLU Networks Suffer from ℓ^2 Adversarial Perturbations”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan & H. Lin. Vol. 33. Curran Associates, Inc., pp. 6629–6636. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/497476fe61816251905e8baafdf54c23-Paper.pdf.
- Darwiche, A. (2020). “Three Modern Roles for Logic in AI”. In: *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*, pp. 229–243. DOI: [10.1145/3375395.3387641](https://doi.org/10.1145/3375395.3387641).
- Devroye, L. & T. Wagner (1979). “Distribution-free inequalities for the deleted and holdout error estimates”. In: *TIT* 25.2, pp. 202–207.
- Dhariwal, P. & A. Nichol (2021). “Diffusion Models Beat GANs on Image Synthesis”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Dhurandhar, A., P.-Y. Chen, R. Luss, C.-C. Tu, P. Ting, K. Shanmugam & P. Das (2018). “Explanations Based on the Missing: Towards Contrastive Explanations with Pertinent Negatives”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Dingle, K., C. Q. Camargo & A. A. Louis (2018). “Input–output maps are strongly biased towards simple outputs”. In: *Nature Communications* 9.1, pp. 1–7.
- Dinh, L., R. Pascanu, S. Bengio & Y. Bengio (2017). “Sharp Minima Can Generalize For Deep Nets”. In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 1019–1028.
- Doshi-Velez, F. & B. Kim (2017). *Towards a Rigorous Science of Interpretable Machine Learning*. arXiv: [1702.08608](https://arxiv.org/abs/1702.08608) [cs.LG]. URL: <https://arxiv.org/abs/1702.08608>.
- Duchi, J., E. Hazan & Y. Singer (2011). “Adaptive subgradient methods for online learning and stochastic optimization”. In: *Journal of machine learning research* 12, Jul, pp. 2121–2159.
- Dziugaite, G. K. & D. M. Roy (2017). “Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks via the PAC-Bayes Framework”. In: *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence (UAI)*.
- El Harzli, O., S. Badreddine & T. R. Besold (2023a). “What’s Wrong with Gradient-Based Complex Query Answering?” In: *Proceedings of the 2023 NeSy Workshop*, –. URL: <https://ceur-ws.org/Vol-3432/paper2.pdf>.

Bibliography

- El Harzli, O. & B. Cuenca Grau (2024). “Adversarial Attacks as Near-Zero Eigenvalues in the Empirical Kernel of Neural Networks”. In: *NeurIPS 2024 Workshop on Mathematics of Modern Machine Learning*. URL: <https://openreview.net/forum?id=QrZwRtwvzC>.
- El Harzli, O. & B. Cuenca Grau (2025). “Bayesian Treatment of the Spectrum of the Empirical Kernel in (Sub)Linear-Width Neural Networks”. In: *The Thirteenth International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=06znYvxClU>.
- El Harzli, O., B. Cuenca Grau, A. D’Avila Garcez, I. Horrocks & T. R. Besold (2026). “From Neural Networks to Logical Theories: The Correspondence between Fibring Modal Logics and Fibring Neural Networks”. In: *The Fourteenth International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=P1iAEhony>.
- El Harzli, O., B. Cuenca Grau & I. Horrocks (2023b). “Cardinality-Minimal Explanations for Monotonic Neural Networks”. In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*. Ed. by E. Elkind. Main Track. International Joint Conferences on Artificial Intelligence Organization, pp. 3677–3685. DOI: [10.24963/ijcai.2023/409](https://doi.org/10.24963/ijcai.2023/409). URL: <https://doi.org/10.24963/ijcai.2023/409>.
- El Harzli, O., B. Cuenca Grau, G. Valle-Pérez & A. A. Louis (2024a). “Double-Descent Curves in Neural Networks: A New Perspective Using Gaussian Processes”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.10, pp. 11856–11864. DOI: [10.1609/aaai.v38i10.29071](https://ojs.aaai.org/index.php/AAAI/article/view/29071). URL: <https://ojs.aaai.org/index.php/AAAI/article/view/29071>.
- El Harzli, O., Y. Nam, I. Kuzborskij, B. Cuenca Grau & A. A. Louis (2024b). “Algorithmic Stability of Minimum-Norm Interpolating Deep Neural Networks”. In: *NeurIPS 2024 Workshop on Mathematics of Modern Machine Learning*. URL: <https://openreview.net/forum?id=D0kCzV9k9h>.
- Elisseeff, A., T. Evgeniou & M. Pontil (2005). “Stability of Randomized Learning Algorithms.” In: *JMLR* 6.1.
- Elkabetz, O. & N. Cohen (2021). “Continuous vs. discrete optimization of deep neural networks”. In: *NeurIPS*.
- Emami, M., M. Sahraee-Ardakan, P. Pandit, S. Rangan & A. K. Fletcher (2020). “Generalization Error of Generalized Linear Models in High Dimensions”. In: *PMLR*.
- Erion, G., J. D. Janizek, P. Sturmfels, S. M. Lundberg & S.-I. Lee (2019). “Improving performance of deep learning models with axiomatic attribution priors and expected gradients”. In: *arXiv preprint arXiv:1906.10670*.
- Fan, Z. & Z. Wang (2020). “Spectra of the Conjugate Kernel and Neural Tangent Kernel for Linear-Width Neural Networks”. In: *Advances in Neural Information Processing Systems* 33.
- Fan, Z. & A. Montanari (2015). “The spectral norm of random inner-product kernel matrices”. In: *Probability Theory and Related Fields* 173, pp. 27–85.
- Fang, C., H. He, Q. Long & W. J. Su (2021). “Layer-Peeled Model: Toward Understanding Well-Trained Deep Neural Networks”. In: *arXiv preprint arXiv:2101.12699*.
- Farghly, T. & P. Rebeschini (2021). “Time-independent generalization bounds for SGLD in non-convex settings”. In: *NeurIPS*.
- Fawzi, A., M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatin, A. Novikov, F. J. R. Ruiz, J. Schrittwieser, G. Swirszcz, D. Silver, D. Hassabis & P. Kohli (2022). “Discovering Faster Matrix Multiplication Algorithms with Reinforcement Learning”. In: *Nature* 610, pp. 47–53. DOI: [10.1038/s41586-022-05172-4](https://doi.org/10.1038/s41586-022-05172-4).

Bibliography

- Feldman, V. & J. Vondrak (2018). “Generalization Bounds for Uniformly Stable Algorithms”. In: *Advances in Neural Information Processing Systems*. Vol. 31.
- Fong, R. C. & A. Vedaldi (2017). “Interpretable Explanations of Black Boxes by Meaningful Perturbation”. In: *ICCV*.
- Foret, P., A. Kleiner, H. Mobahi & B. Neyshabur (2021). “Sharpness-Aware Minimization for Efficiently Improving Generalization”. In: *International Conference on Learning Representations (ICLR)*.
- Frankle, J. & M. Carbin (2018). “The lottery ticket hypothesis: Finding sparse, trainable neural networks”. In: *ICLR*.
- Frankle, J., G. K. Dziugaite, D. Roy & M. Carbin (2020). “Linear mode connectivity and the lottery ticket hypothesis”. In: *ICML*.
- Frei, S., G. Vardi, P. Bartlett & N. Srebro (2023). “Benign overfitting in linear classifiers and leaky relu networks from kkt conditions for margin maximization”. In: *COLT*. PMLR.
- Frei, S., G. Vardi, P. Bartlett, N. Srebro & W. Hu (2022). “Implicit Bias in Leaky ReLU Networks Trained on High-Dimensional Data”. In: *ICLR*.
- Friedman, E. (2004). “Paths and Consistency in Additive Cost Sharing”. In: *International Journal of Game Theory* 32.4, pp. 501–518. DOI: [10.1007/s001820400173](https://doi.org/10.1007/s001820400173).
- Friedman, J. H. & B. E. Popescu (2008). “Predictive Learning via Rule Ensembles”. In: *Annals of Applied Statistics* 2.3, pp. 916–954. DOI: [10.1214/07-A0AS148](https://doi.org/10.1214/07-A0AS148).
- Frosst, N. & G. E. Hinton (2017). “Distilling a Neural Network Into a Soft Decision Tree”. In: *Comprehensibility and Explanation in AI and ML (CEx@AI*IA 2017)*.
- Gabbay, D. M. (1999). *Fibiring Logics*. New York: Clarendon Press.
- Gal, Y. (2016). “Uncertainty in Deep Learning”. PhD thesis. University of Cambridge.
- Galanti, T., Z. S. Siegel, A. Gupte & T. Poggio (2023). *Characterizing the Implicit Bias of Regularized SGD in Rank Minimization*. arXiv: [2206.05794](https://arxiv.org/abs/2206.05794) [cs.LG].
- Garcez, A. d’Avila & L. C. Lamb (2023). “Neurosymbolic AI: the 3rd wave”. In: *Artificial Intelligence Review* 56.11, pp. 12387–12406. DOI: [10.1007/s10462-023-10448-w](https://doi.org/10.1007/s10462-023-10448-w). URL: <https://doi.org/10.1007/s10462-023-10448-w>.
- Garcez, A. S. d. & D. M. Gabbay (2004). “Fibiring neural networks”. In: *Proceedings of the 19th National Conference on Artificial Intelligence. AAAI’04*. San Jose, California: AAAI Press, pp. 342–347. ISBN: 0262511835.
- Gardner, E. (1988). “The Space of Interactions in Neural Network Models”. In: *Journal of Physics A: Mathematical and General* 21.1, pp. 257–270. DOI: [10.1088/0305-4470/21/1/030](https://doi.org/10.1088/0305-4470/21/1/030).
- Geiger, M., A. Jacot, S. Spigler, F. Gabriel, L. Sagun, S. d’Ascoli, G. Biroli, C. Hongler & M. Wyart (2020). “Scaling Description of Generalization with Number of Parameters in Deep Learning”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2020.2, p. 023401. DOI: [10.1088/1742-5468/ab633c](https://doi.org/10.1088/1742-5468/ab633c).
- Gerace, F., B. Loureiro, F. Krzakala, M. Mezard & L. Zdeborova (2020). “Generalisation error in learning with random features and the hidden manifold model”. In: *ICML*.
- Gerbelot, C., A. Abbata & F. Krzakala (2020). “Asymptotic errors for convex penalized linear regression beyond Gaussian matrices”. In: *ICML*.
- Geronimo, J. S. & T. P. Hill (2003). “Necessary and Sufficient Condition that the Limit of Stieltjes Transforms is a Stieltjes Transform”. In: *Journal of Approximation Theory* 121.1, pp. 54–60. DOI: [10.1016/S0021-9045\(02\)00042-4](https://doi.org/10.1016/S0021-9045(02)00042-4).
- Ghorbani, B., S. Mei, T. Misiakiewicz & A. Montanari (2020). “When Do Neural Networks Outperform Kernel Methods?” In: *NeurIPS* 33.

Bibliography

- Glashoff, K. & M. M. Bronstein (2013). “Almost-commuting matrices are almost jointly diagonalizable”. In: *arXiv preprint arXiv:1305.2135*.
- Golowich, N., A. Rakhlin & O. Shamir (2018). “Size-Independent Sample Complexity of Neural Networks”. In: *Proceedings of the 31st Conference on Learning Theory*. Ed. by S. Bubeck, V. Perchet & P. Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, pp. 297–299.
- Goodfellow, I., Y. Bengio & A. Courville (2016). *Deep Learning*. <http://www.deeplearningbook.org>. MIT Press.
- Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville & Y. Bengio (2014). “Generative Adversarial Nets”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Goodfellow, I., J. Shlens & C. Szegedy (2015). “Explaining and Harnessing Adversarial Examples”. In: *International Conference on Learning Representations*. URL: <http://arxiv.org/abs/1412.6572>.
- Goyal, Y., Z. Wu, J. Ernst, D. Batra, D. Parikh & S. Lee (2019). “Counterfactual Visual Explanations”. In: *Proceedings of ICML*. PMLR, pp. 2376–2384.
- Greshake, K., R. Chevalier, et al. (2023). *Not What You’ve Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection*. arXiv: [2302.12173](https://arxiv.org/abs/2302.12173) [cs.CR].
- Grohe, M. (2023). “The Descriptive Complexity of Graph Neural Networks”. In: *38th ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pp. 1–14. DOI: [10.1109/LICS56636.2023.10175735](https://doi.org/10.1109/LICS56636.2023.10175735).
- Gu, J. & V. Tresp (2019). “Saliency Methods for Explaining Adversarial Attacks”. In: *arXiv*. DOI: [arXiv:1908.08413](https://arxiv.org/abs/1908.08413).
- Guo, Y., C. Zhang, C. Zhang & Y. Chen (2018). “Sparse DNNs with Improved Adversarial Robustness”. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. NIPS’18. Montréal, Canada: Curran Associates Inc., pp. 240–249.
- Györfi, L., M. Kohler, A. Krzyzak & H. Walk (2002). *A Distribution-Free Theory of Non-parametric Regression*. Springer.
- Han, X., V. Pappas & D. L. Donoho (2021). “Neural Collapse Under MSE Loss: Proximity to and Dynamics on the Central Path”. In: *ICLR*.
- Hanin, B. & A. Zlokapa (2023). “Bayesian interpolation with deep linear networks”. In: *Proceedings of the National Academy of Sciences* 120.23, e2301345120. DOI: [10.1073/pnas.2301345120](https://doi.org/10.1073/pnas.2301345120). eprint: <https://www.pnas.org/doi/pdf/10.1073/pnas.2301345120>. URL: <https://www.pnas.org/doi/abs/10.1073/pnas.2301345120>.
- Hao, Y., D. Angluin & R. Frank (2022). “Formal Language Recognition by Hard Attention Transformers: Perspectives from Circuit Complexity”. In: *Transactions of the Association for Computational Linguistics* 10, pp. 800–810.
- Hardt, M., B. Recht & Y. Singer (2016). “Train faster, generalize better: Stability of stochastic gradient descent”. In: *ICML*.
- Hastie, T., A. Montanari, S. Rosset & R. J. Tibshirani (2019). “Surprises in high-dimensional ridgeless least squares interpolation”. In: *arXiv preprint arXiv:1903.08560*.
- Hayase, J., E. Borevkovic, N. Carlini, F. Tramèr & M. Nasr (2024). “Query-Based Adversarial Prompt Generation”. In: *Advances in Neural Information Processing Systems*. Vol. 37.
- He, Y., W. Zheng, Y. Dong, Y. Zhu, C. Chen & J. Li (2025). “Towards Global-level Mechanistic Interpretability: A Perspective of Modular Circuits of Large Language

Bibliography

- Models". In: *Forty-second International Conference on Machine Learning*. URL: <https://openreview.net/forum?id=do5vVfKEXZ>.
- Hendrycks, D. & T. Dietterich (2019). "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations". In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=HJz6tiCqYm>.
- Ho, J., W. Chan, C. Saharia, et al. (2022). "Imagen Video: High Definition Video Generation with Diffusion Models". In: *arXiv preprint arXiv:2210.02303*.
- Ho, J., A. Jain & P. Abbeel (2020). "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Hochreiter, S. & J. Schmidhuber (1997a). "Flat minima". In: *Neural Computation* 9.1, pp. 1–42.
- Hochreiter, S. & J. Schmidhuber (1997b). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780.
- Hornik, K. (1991). "Approximation capabilities of multilayer feedforward networks". In: *Neural networks* 4.2, pp. 251–257.
- Huben, R., H. Cunningham, L. R. Smith, A. Ewart & L. Sharkey (2024). "Sparse Autoencoders Find Highly Interpretable Features in Language Models". In: *International Conference on Learning Representations (ICLR)*. URL: <https://openreview.net/forum?id=8MKcD6zX2L>.
- Ignatiev, A., Y. Izza, P. J. Stuckey & J. Marques-Silva (2022). "Using MaxSAT for Efficient Explanations of Tree Ensembles". In: *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI)*. Vol. 36. 6, pp. 3776–3785. DOI: <https://doi.org/10.1609/aaai.v36i4.20292>.
- Ignatiev, A., N. Narodytska & J. Marques-Silva (2019). "Abduction-Based Explanations for Machine Learning Models". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 1511–1519. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/3964>.
- Ilyas, A., S. Santurkar, D. Tsipras, L. Engstrom, B. Tran & A. Madry (2019). "Adversarial Examples Are Not Bugs, They Are Features". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox & R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/e2c420d928d4bf8ce0ff2ec19b371514-Paper.pdf.
- Izza, Y. & J. Marques-Silva (2021). "On Explaining Random Forests with SAT". In: *Proc. of IJCAI*. Ed. by Z. Zhou. ijcai.org, pp. 2584–2591.
- Jacot, A., F. Gabriel & C. Hongler (2018). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)* 31, pp. 8571–8580.
- Jacot, A., B. Simsek, F. Spadaro, C. Hongler & F. Gabriel (2020). "Kernel Alignment Risk Estimator: Risk Prediction from Training Data". In: *NeurIPS*. Vol. 33.
- Jastrzębski, S., D. Arpit, N. Ballas, A. Fischer, Y. Bengio & A. Storkey (2017). "Three Factors Influencing Minima in SGD". In: *arXiv preprint arXiv:1711.04623*.
- Ji, S., S. Pan, E. Cambria, P. Marttinen & P. S. Yu (2022). "A Survey on Knowledge Graphs: Representation, Acquisition, and Applications". In: *IEEE Transactions on Neural Networks and Learning Systems* 33.2, pp. 494–514. DOI: [10.1109/TNNLS.2021.3070843](https://doi.org/10.1109/TNNLS.2021.3070843). URL: <https://doi.org/10.1109/TNNLS.2021.3070843>.
- Ji, Z. & M. Telgarsky (2019). "Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow ReLU networks". In: *ICLR*.

Bibliography

- Ji, Z. & M. Telgarsky (2020). "Directional convergence and alignment in deep learning". In: *NeurIPS*.
- Jiang, Y., B. Neyshabur, H. Mobahi, D. Krishnan & S. Bengio (2019). "Fantastic Generalization Measures and Where to Find Them". In: *International Conference on Learning Representations (ICLR)*.
- Jones, L. (1997). "The computational intractability of training sigmoidal neural networks". In: *IEEE Transactions on Information Theory* 43.1, pp. 167–173. DOI: [10.1109/18.567673](https://doi.org/10.1109/18.567673).
- Judd, S. (1988). "On the complexity of loading shallow neural networks". In: *Journal of Complexity* 4.3, pp. 177–192. DOI: [https://doi.org/10.1016/0885-064X\(88\)90019-2](https://doi.org/10.1016/0885-064X(88)90019-2).
- Jumper, J., R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes, S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli & D. Hassabis (2021). "Highly Accurate Protein Structure Prediction with AlphaFold". In: *Nature* 596, pp. 583–589. DOI: [10.1038/s41586-021-03819-2](https://doi.org/10.1038/s41586-021-03819-2).
- Karoui, N. E. (2010). "The spectrum of kernel random matrices". In: *The Annals of Statistics* 38.1. DOI: [10.1214/08-aos648](https://doi.org/10.1214/08-aos648). URL: <https://doi.org/10.1214%2F08-aos648>.
- Katz, G., C. Barrett, D. Dill, K. Julian & M. Kochenderfer (2017). "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks". In: *Computer Aided Verification*. Springer International Publishing, pp. 97–117.
- Kazim, E. & A. S. Koshiyama (2021). "A high-level overview of AI ethics". In: *Patterns* 2 (9). DOI: [10.1016/j.patter.2021.100314](https://doi.org/10.1016/j.patter.2021.100314).
- Keskar, N. S., D. Mudigere, J. Nocedal, M. Smelyanskiy & P. T. P. Tang (2017). "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima". In: *International Conference on Learning Representations (ICLR)*.
- Kim, B., R. Khanna & O. O. Koyejo (2016). "Examples are not enough, learn to criticize! Criticism for Interpretability". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee, M. Sugiyama, U. Luxburg, I. Guyon & R. Garnett. Vol. 29. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2016/file/5680522b8e2bb01943234bce7bf84534-Paper.pdf.
- Kindermans, P.-J., S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan & B. Kim (2019). "The (Un)reliability of Saliency Methods". In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, pp. 267–280.
- Kingma, D. P. & J. Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.
- Kipf, T. & M. Welling (2017). "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations (ICLR)*.
- Koehler, F., L. Zhou, D. J. Sutherland & N. Srebro (2021). "Uniform convergence of interpolators: Gaussian width, norm bounds and benign overfitting". In: *NeurIPS*.
- Koh, P. W. & P. Liang (2017). "Understanding Black-box Predictions via Influence Functions". In: *Proceedings of ICML*. PMLR.
- Krizhevsky, A., I. Sutskever & G. E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 25.

Bibliography

- Kumar, A., T. Serra & S. Ramalingam (2019). “Equivalent and Approximate Transformations of Deep Neural Networks”. In: *CoRR* abs/1905.11428. arXiv: 1905.11428. URL: <http://arxiv.org/abs/1905.11428>.
- Kutin, S. & P. Niyogi (2002). “Almost-everywhere algorithmic stability and generalization error”. In: *UAI*.
- Kuzborskij, I. & C. H. Lampert (2018). “Data-Dependent Stability of Stochastic Gradient Descent”. In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 2820–2829.
- Lamb, L. C., A. d’Avila Garcez, M. Gori, M. O. R. Prates, P. H. C. Avelar & M. Y. Vardi (2020). “Graph Neural Networks Meet Neural-Symbolic Computing: A Survey and Perspective”. In: *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI)*. Survey track, pp. 4877–4884. DOI: 10.24963/ijcai.2020/679.
- LeCun, Y., L. Bottou, Y. Bengio & P. Haffner (1998a). “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.
- LeCun, Y., C. Cortes & C. J. C. Burges (1998b). *The MNIST Database of Handwritten Digits*. <http://yann.lecun.com/exdb/mnist/>.
- Lee, J., S. Schoenholz, J. Pennington, B. Adlam, L. Xiao, R. Novak & J. Sohl-Dickstein (2020). “Finite versus infinite neural networks: an empirical study”. In: *NeurIPS* 33.
- Lee, J., J. Sohl-dickstein, J. Pennington, R. Novak, S. Schoenholz & Y. Bahri (2018). “Deep Neural Networks as Gaussian Processes”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=B1EA-M-0Z>.
- Lei, Y. & Y. Ying (2020). “Fine-grained analysis of stability and generalization for stochastic gradient descent”. In: *ICML*.
- Letham, B., C. Rudin, T. H. McCormick & D. Madigan (2015). “Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model”. In: *Annals of Applied Statistics* 9.3, pp. 1350–1371. DOI: 10.1214/15-AOAS848.
- Lewkowycz, A., Y. Bahri, E. Dyer, J. Sohl-Dickstein & G. Gur-Ari (2020). “The Large Learning Rate Phase of Deep Learning: The Catapult Mechanism”. In: *International Conference on Learning Representations*.
- Li, H., Z. Xu, G. Taylor, C. Studer & T. Goldstein (2018). “Visualizing the loss landscape of neural nets”. In: *Advances in Neural Information Processing Systems*, pp. 6389–6399.
- Li, Q. & H. Sompolinsky (2021). “Statistical mechanics of deep linear neural networks: The backpropagating kernel renormalization”. In: *Physical Review X* 11.3, p. 031059.
- Liao, Z., R. Couillet & M. W. Mahoney (2020). “A random matrix analysis of random Fourier features: beyond the Gaussian kernel, a precise phase transition, and the corresponding double descent”. In: *NeurIPS* 33.
- Lin, C., I. Covert & S.-I. Lee (2023). “On the Robustness of Removal-Based Feature Attributions”. In: *Advances in Neural Information Processing Systems*. Vol. 36.
- Lipton, Z. C. (2018). “The Mythos of Model Interpretability”. In: *Communications of the ACM* 61.10, pp. 36–43. DOI: 10.1145/3233231.
- Liu, F., Z. Liao & J. Suykens (2021). “Kernel regression in high dimensions: Refined analysis beyond double descent”. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by A. Banerjee & K. Fukumizu. Vol. 130. *Proceedings of Machine Learning Research*. PMLR, pp. 649–657. URL: <https://proceedings.mlr.press/v130/liu21b.html>.
- Liu, X., X. Han, N. Zhang & Q. Liu (2020). “Certified Monotonic Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 33, pp. 15427–15438. URL: <https://arxiv.org/abs/2006.04238>.

Bibliography

- [//proceedings.neurips.cc/paper/2020/file/b139aeda1c2914e3b579aafd3ceeb1bd-Paper.pdf](https://proceedings.neurips.cc/paper/2020/file/b139aeda1c2914e3b579aafd3ceeb1bd-Paper.pdf).
- Livan, G., M. Novaes & P. Vivo (2018). *Introduction to Random Matrices*. Springer International Publishing. DOI: [10.1007/978-3-319-70885-0](https://doi.org/10.1007/978-3-319-70885-0). URL: <https://doi.org/10.1007/978-3-319-70885-0>.
- Louart, C., Z. Liao & R. Couillet (2018). “A Random Matrix Approach to Neural Networks”. In: *The Annals of Applied Probability* 28.2, pp. 1190–1248. ISSN: 10505164, 21688737. URL: <https://www.jstor.org/stable/26542333> (visited on 09/25/2023).
- Lu, Y. M. & H.-T. Yau (2023). *An Equivalence Principle for the Spectrum of Random Inner-Product Kernel Matrices with Polynomial Scalings*. arXiv: [2205.06308](https://arxiv.org/abs/2205.06308) [math.PR].
- Lucic, A., M. A. ter Hoeve, G. Tolomei, M. de Rijke & F. Silvestri (2022). “CF-GNNExplainer: Counterfactual Explanations for Graph Neural Networks”. In: *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*. Vol. 151. Proceedings of Machine Learning Research. PMLR, pp. 4499–4511.
- Lundberg, S. M. & S.-I. Lee (2017). “A Unified Approach to Interpreting Model Predictions”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Lyu, K. & J. Li (2019). “Gradient Descent Maximizes the Margin of Homogeneous Neural Networks”. In: *ICLR*.
- MacKay, D. J. C. (1992). “A practical Bayesian framework for backpropagation networks”. In: *Neural Computation* 4.3, pp. 448–472.
- Madry, A., A. Makelov, L. Schmidt, D. Tsipras & A. Vladu (2018). “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=rJzIBfZAb>.
- Magris, M. & A. Iosifidis (2023). “Bayesian Learning for Neural Networks: An Algorithmic Survey”. In: *Artificial Intelligence Review* 56.10, pp. 11773–11823. DOI: [10.1007/s10462-023-10443-1](https://doi.org/10.1007/s10462-023-10443-1).
- Maillard, A., E. Troiani, S. Martin, F. Krzakala & L. Zdeborová (2024). *Bayes-optimal learning of an extensive-width neural network from quadratically many samples*. arXiv: [2408.03733](https://arxiv.org/abs/2408.03733) [stat.ML]. URL: <https://arxiv.org/abs/2408.03733>.
- Malach, E., G. Yehudai, S. Shalev-Schwartz & O. Shamir (2020). “Proving the lottery ticket hypothesis: Pruning is all you need”. In: *ICML*.
- Mallinar, N., J. B. Simon, A. Abedsoltan, P. Pandit, M. Belkin & P. Nakkiran (2022). “Benign, Tempered, or Catastrophic: A Taxonomy of Overfitting”. In: *arXiv*. URL: <https://arxiv.org/abs/2207.06569>.
- Marchenko, V. & L. Pastur (1967). “Distribution of eigenvalues for some sets of random matrices”. In: *Matematicheskii Sbornik* 72.
- Marques-Silva, J. (2022). “Logic-based Explainability in Machine Learning”. In: *Artificial Intelligence* 303, p. 103645. DOI: [10.1016/j.artint.2021.103645](https://doi.org/10.1016/j.artint.2021.103645).
- Marques-Silva, J., T. Gerspacher, M. C. Cooper, A. Ignatiev & N. Narodytska (2021). “Explanations for Monotonic Classifiers”. In: *Proceedings of ICML*. PMLR, pp. 7469–7479.
- Matthews, A. G. d. G., M. Rowland, J. Hron, R. E. Turner & Z. Ghahramani (2018). “Gaussian process behaviour in wide deep neural networks”. In: *arXiv preprint arXiv:1804.11271*.
- McAllester, D. (1999). “PAC-Bayesian model averaging”. In: *Proceedings of the 12th Annual Conference on Computational Learning Theory (COLT)*, pp. 164–170.

Bibliography

- Meegen, A. van & H. Sompolinsky (2024). *Coding schemes in neural networks learning classification tasks*. arXiv: 2406.16689 [cs.LG]. URL: <https://arxiv.org/abs/2406.16689>.
- Mei, S. & A. Montanari (2019). “The generalization error of random features regression: Precise asymptotics and double descent curve”. In: *arXiv preprint arXiv:1908.05355*.
- Meng, K., D. Bau, A. Andonian & Y. Belinkov (2022). “Locating and Editing Factual Associations in GPT”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. URL: https://papers.nips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.
- Mercer, J. (1909). “Functions of Positive and Negative Type, and Their Connection with the Theory of Integral Equations”. In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 209, pp. 415–446. DOI: [10.1098/rsta.1909.0016](https://doi.org/10.1098/rsta.1909.0016).
- Mingard, C., J. Skalse, G. Valle-Pérez, D. Martínez-Rubio, V. Mikulik & A. A. Louis (2019). “Neural networks are a priori biased towards Boolean functions with low entropy”. In: *arXiv preprint arXiv:1909.11522*.
- Mingard, C., G. Valle-Pérez, J. Skalse & A. A. Louis (2020). “Is SGD a Bayesian sampler? Well, almost”. In: *arXiv preprint arXiv:2006.15191*.
- Mingo, J. A. & R. Speicher (2017). *Free Probability and Random Matrices*. Springer. DOI: <https://doi.org/10.1007/978-1-4939-6942-5>.
- Minh, H. Q., P. Niyogi & Y. Yao (2006). “Mercer’s Theorem, Feature Maps, and Smoothing”. In: *Learning Theory*. Vol. 4005. Lecture Notes in Computer Science. Springer, pp. 154–168. DOI: [10.1007/11776420_14](https://doi.org/10.1007/11776420_14).
- Mixon, D. G., H. Parshall & J. Pi (2022). “Neural collapse with unconstrained features”. In: *Sampling Theory, Signal Processing, and Data Analysis* 20.2, p. 11.
- Mohri, M., A. Rostamizadeh & A. Talwalkar (2012). *Foundations of Machine Learning*. MIT Press.
- Moore, E. (1920). “On the reciprocal of the general algebraic matrix”. In: *Bulletin of the American Mathematical Society*.
- Moosavi-Dezfooli, S.-M., A. Fawzi & P. Frossard (2016). “DeepFool: a simple and accurate method to fool deep neural networks”. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016*.
- Naddeo, K., N. Bouaynaya & R. Shterenberg (2022). “An Information Geometric Perspective to Adversarial Attacks and Defenses”. In: *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. DOI: [10.1109/IJCNN55064.2022.9892170](https://doi.org/10.1109/IJCNN55064.2022.9892170).
- Nagarajan, V. & J. Z. Kolter (2019). “Uniform convergence may be unable to explain generalization in deep learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 32.
- Nakkiran, P., G. Kaplun, Y. Bansal, T. Yang, B. Barak & I. Sutskever (2021). “Deep Double Descent: Where Bigger Models and More Data Hurt”. In: *Journal of Statistical Mechanics: Theory and Experiment* 12, p. 124003.
- Nam, Y., N. Fonseca, S. H. Lee, C. Mingard, N. Goring, O. E. Harzli, A. H. Erturk, S. Hayou & A. A. Louis (2025). *Disentangling Rich Dynamics from Feature Learning: A Framework for Independent Measurements*. arXiv: 2410.04264 [stat.ML]. URL: <https://arxiv.org/abs/2410.04264>.
- Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Vol. 118. Lecture Notes in Statistics. Springer.

Bibliography

- Neyshabur, B., R. Tomioka & N. Srebro (2015). “Norm-Based Capacity Control in Neural Networks”. In: *Proceedings of the 28th Conference on Learning Theory (COLT)*, pp. 1376–1401.
- Nunn, P., M. Sälzer, F. Schwarzentruher & N. Troquard (2024). “A Logic for Reasoning about Aggregate-Combine Graph Neural Networks”. In: *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*. Ed. by K. Larson. Main Track. International Joint Conferences on Artificial Intelligence Organization, pp. 3532–3540. DOI: [10.24963/ijcai.2024/391](https://doi.org/10.24963/ijcai.2024/391). URL: <https://doi.org/10.24963/ijcai.2024/391>.
- Olah, C., A. Mordvintsev & L. Schubert (2017). “Feature Visualization”. In: *Distill* 2.11, e7. DOI: [10.23915/distill.00007](https://doi.org/10.23915/distill.00007).
- Olsson, C., N. Elhage, N. Nanda, et al. (2022). *In-context Learning and Induction Heads*. Transformer Circuits (online). URL: <https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html>.
- OpenAI (2023). *GPT-4 Technical Report*. arXiv: [2303.08774](https://arxiv.org/abs/2303.08774) [cs.CL]. URL: <https://arxiv.org/abs/2303.08774>.
- Orseau, L., M. Hutter & O. Rivasplata (2020). “Logarithmic pruning is all you need”. In: *NeurIPS*.
- Ouyang, L., J. Wu, X. Jiang, et al. (2022). “Training Language Models to Follow Instructions with Human Feedback”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Pacelli, R., S. Ariosto, M. Pastore, F. Ginelli, M. Gherardi & P. Rotondo (2023). “A Statistical Mechanics Framework for Bayesian Deep Neural Networks Beyond the Infinite-Width Limit”. In: *Nature Machine Intelligence* 5, pp. 1497–1507. DOI: [10.1038/s42256-023-00767-6](https://doi.org/10.1038/s42256-023-00767-6).
- Paknezhad, M., C. P. Ngo, A. A. Winarto, A. Cheong, C. Y. Beh, J. Wu & H. K. Lee (2022). “Explaining adversarial vulnerability with a data sparsity hypothesis”. In: *Neurocomputing* 495, pp. 178–193. ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2022.01.062>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231222000819>.
- Papayan, V., X. Han & D. L. Donoho (2020). “Prevalence of neural collapse during the terminal phase of deep learning training”. In: *Proceedings of the National Academy of Sciences of the United States of America* 117.40, pp. 24652–24663.
- Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai & S. Chintala (2019). “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems* 32, pp. 8024–8035.
- Perez, E., S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese & G. Irving (2022). “Red Teaming Language Models with Language Models”. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 3419–3448. DOI: [10.18653/v1/2022.emnlp-main.225](https://doi.org/10.18653/v1/2022.emnlp-main.225). URL: <https://aclanthology.org/2022.emnlp-main.225/>.
- Phuong, M. & C. H. Lampert (2020). “The inductive bias of ReLU networks on orthogonally separable data”. In: *ICLR*.
- Poggio, T., A. Banburski & Q. Liao (2020). “Theoretical issues in deep networks”. In: *Proceedings of the National Academy of Sciences of the United States of America* 117.48, pp. 30039–30045.

Bibliography

- Poole, B., S.Lahiri, M. Raghu, J. Sohl-Dickstein & S. Ganguli (2016). "Exponential expressivity in deep neural networks through transient chaos". In: *NeurIPS* 29.
- Pruthi, G., F. Liu, S. Kale & M. Sundararajan (2020). "Estimating Training Data Influence by Tracing Gradient Descent". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Qin, C., J. Martens, S. Gowal, D. Krishnan, K. Dvijotham, A. Fawzi, S. De, R. Stanforth & P. Kohli (2019). "Adversarial Robustness through Local Linearization". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox & R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/0defd533d51ed0a10c5c9dbf93ee78a5-Paper.pdf.
- Rahimi, A. & B. Recht (2007). "Random Features for Large-Scale Kernel Machines". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Rahimi, A. & B. Recht (2008). "Uniform Approximation of Functions with Random Bases". In: *Allerton Conference on Communication, Control, and Computing*.
- Rahimi, A. & B. Recht (2009). "Weighted Sums of Random Kitchen Sinks: Replacing Minimization with Randomization in Learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Rangamani, A., L. Rosasco & T. Poggio (2023). "For interpolating kernel machines, minimizing the norm of the ERM solution maximizes stability". In: *Analysis and Applications* 21.01, pp. 193–215.
- Rasmussen, C. & C. Williams (2006). *Gaussian Processes for Machine Learning*. MIT Press.
- Rasmussen, C. (1995). "A Practical Monte Carlo Implementation of Bayesian Learning". In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky, M. Mozer & M. Hasselmo. Vol. 8. MIT Press. URL: https://proceedings.neurips.cc/paper_files/paper/1995/file/84d2004bf28a2095230e8e14993d398d-Paper.pdf.
- Ribeiro, M. T., S. Singh & C. Guestrin (2016). "'Why Should I Trust You?': Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1135–1144. DOI: [10.1145/2939672.2939778](https://doi.org/10.1145/2939672.2939778).
- Ribeiro, M. T., S. Singh & C. Guestrin (2018). "Anchors: High-Precision Model-Agnostic Explanations". In: *Proceedings of AAAI*, pp. 1527–1535.
- Richards, D. & I. Kuzborskij (2021). "Stability & Generalisation of Gradient Descent for Shallow Neural Networks without the Neural Tangent Kernel". In: *NeurIPS*.
- Richards, D. & M. Rabbat (2021). "Learning with Gradient Descent and Weakly Convex Losses". In: *AISTATS*.
- Rombach, R., A. Blattmann, D. Lorenz, P. Esser & B. Ommer (2022). "High-Resolution Image Synthesis with Latent Diffusion Models". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695.
- Ronneberger, O., P. Fischer & T. Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241.
- Saitoh, S. & Y. Sawano (2016). *Theory of Reproducing Kernels and Applications*. Vol. 44. Developments in Mathematics. Springer Singapore. ISBN: 978-981-10-0530-5. DOI: [10.1007/978-981-10-0530-5](https://doi.org/10.1007/978-981-10-0530-5).
- Sakamoto, K. & I. Sato (2022). "Analyzing lottery ticket hypothesis from PAC-bayesian theory perspective". In: *NeurIPS*.

Bibliography

- Scarselli, F., M. Gori, A. C. Tsoi, M. Hagenbuchner & G. Monfardini (2009). “The Graph Neural Network Model”. In: *IEEE Transactions on Neural Networks* 20.1, pp. 61–80.
- Schaper, S. & A. A. Louis (2014). “The arrival of the frequent: how bias in genotype-phenotype maps can steer populations to local optima”. In: *PloS one* 9.2, e86635.
- Schilling, R. L. (2005). *Measures, Integrals and Martingales*. Cambridge University Press.
- Schliserman, M. & T. Koren (2022). “Stability vs implicit bias of gradient methods on separable data and beyond”. In: *Conference on Learning Theory*.
- Schröder, D., H. Cui, D. Dmitriev & B. Loureiro (2023). “Deterministic equivalent and error universality of deep random features learning”. In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato & J. Scarlett. Vol. 202. Proceedings of Machine Learning Research. PMLR, pp. 30285–30320. URL: <https://proceedings.mlr.press/v202/schroder23a.html>.
- Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh & D. Batra (2017). “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: *ICCV*.
- Seung, H. S., H. Sompolinsky & N. Tishby (1992). “Statistical Mechanics of Learning from Examples”. In: *Physical Review A* 45.8, pp. 6056–6091. DOI: [10.1103/PhysRevA.45.6056](https://doi.org/10.1103/PhysRevA.45.6056).
- Shafahi, A., W. R. Huang, C. Studer, S. Feizi & T. Goldstein (2019). “Are adversarial examples inevitable?” In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=r1lWUoA9FQ>.
- Shalev-Shwartz, S. & S. Ben-David (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.
- Shalev-Shwartz, S., O. Shamir, N. Srebro & K. Sridharan (2010). “Learnability, stability and uniform convergence”. In: *JMLR* 11, pp. 2635–2670.
- Shapley, L. (1953). *Contributions to the Theory of Games II*. Princeton University Press. Chap. A Value for n-Person Games. DOI: [10.1515/9781400881970-018](https://doi.org/10.1515/9781400881970-018).
- Sharkey, L., B. Chughtai, J. Batson, J. Lindsey, J. Wu, L. Bushnaq, N. Goldowsky-Dill, S. Heimersheim, A. Ortega, J. Bloom, S. Biderman, A. Garriga-Alonso, A. Conmy, N. Nanda, J. Rumbelow, M. Wattenberg, N. Schoots, J. Miller, E. J. Michaud, S. Casper, M. Tegmark, W. Saunders, D. Bau, E. Todd, A. Geiger, M. Geva, J. Hoogland, D. Murfet & T. McGrath (2025). *Open Problems in Mechanistic Interpretability*. arXiv: [2501.16496](https://arxiv.org/abs/2501.16496).
- Shi, W., A. Shih, A. Darwiche & A. Choi (2020). “On Tractable Representations of Binary Neural Networks”. In: *Proc. of KR*. Ed. by D. Calvanese, E. Erdem & M. Thielscher, pp. 882–892.
- Shih, A., A. Choi & A. Darwiche (2018). “A Symbolic Approach to Explaining Bayesian Network Classifiers”. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, pp. 5103–5111. DOI: [10.24963/ijcai.2018/708](https://doi.org/10.24963/ijcai.2018/708). URL: <https://doi.org/10.24963/ijcai.2018/708>.
- Shrikumar, A., P. Greenside & A. Kundaje (2017). “Learning Important Features through Propagating Activation Differences”. In: *Proceedings of ICML*. PMLR. URL: <https://proceedings.mlr.press/v70/shrikumar17a.html>.
- Shu, D., X. Wu, H. Zhao, D. Rai, Z. Yao, N. Liu & M. Du (2025). “A Survey on Sparse Autoencoders: Interpreting the Internal Mechanisms of Large Language Models”. In: *Findings of the Association for Computational Linguistics: EMNLP 2025*. Association

Bibliography

- for Computational Linguistics, pp. 1690–1712. DOI: [10.18653/v1/2025.findings-emnlp.89](https://doi.org/10.18653/v1/2025.findings-emnlp.89). URL: <https://aclanthology.org/2025.findings-emnlp.89/>.
- Silver, D., A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, et al. (2016). “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529, pp. 484–489. DOI: [10.1038/nature16961](https://doi.org/10.1038/nature16961).
- Simon, J. B., M. Dickens, D. Karkada & M. Deweese (2023). “The Eigenlearning Framework: A Conservation Law Perspective on Kernel Ridge Regression and Wide Neural Networks”. In: *Transactions on Machine Learning Research*. ISSN: 2835-8856. URL: <https://openreview.net/forum?id=FDbQGCAViI>.
- Simon-Gabriel, C.-J., Y. Ollivier, L. Bottou, B. Schölkopf & D. Lopez-Paz (2019). “First-Order Adversarial Vulnerability of Neural Networks and Input Dimension”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri & R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 5809–5817. URL: <https://proceedings.mlr.press/v97/simon-gabriel19a.html>.
- Simonyan, K. & A. Zisserman (2014). “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556*.
- Simonyan, K., A. Vedaldi & A. Zisserman (2014). “Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps”. In: *ICLR*.
- Smilkov, D., N. Thorat, B. Kim, F. Viégas & M. Wattenberg (2017). *SmoothGrad: Removing Noise by Adding Noise*. arXiv: [1706.03825 \[cs.LG\]](https://arxiv.org/abs/1706.03825).
- Smith, S. L. & Q. V. Le (2018). “A Bayesian Perspective on Generalization and Stochastic Gradient Descent”. In: *International Conference on Learning Representations (ICLR)*.
- Sollich, P. & A. Halees (2002). “Learning Curves for Gaussian Process Regression: Approximations and Bounds”. In: *Neural Computation* 14.6, pp. 1393–1428. DOI: [10.1162/089976602753712990](https://doi.org/10.1162/089976602753712990). URL: <https://doi.org/10.1162/089976602753712990>.
- Soudry, D., E. Hoffer, M. S. Nacson, S. Gunasekar & N. Srebro (2018). “The implicit bias of gradient descent on separable data”. In: *Journal of Machine Learning Research* 19.70, pp. 1–57.
- Spigler, S., M. Geiger, S. d’Ascoli, L. Sagun, G. Biroli & M. Wyart (2019). “A jamming transition from under- to over-parametrization affects loss landscape and generalization in deep learning”. In: *Journal of Physics A: Mathematical and Theoretical* 52.47, p. 474001.
- Srinivas, S. & F. Fleuret (2019). “Full-Gradient Representation for Neural Network Visualization”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Sundararajan, M. & A. Najmi (2020). “The Many Shapley Values for Model Explanation”. In: *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. PMLR, pp. 9269–9278. URL: <https://proceedings.mlr.press/v119/sundararajan20b.html>.
- Sundararajan, M., A. Taly & Q. Yan (2017). “Axiomatic Attribution for Deep Networks”. In: *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. PMLR, pp. 3319–3328. URL: <https://proceedings.mlr.press/v70/sundararajan17a.html>.
- Szegedy, C., W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow & R. Fergus (2014). “Intriguing properties of neural networks”. In: *International Conference on Learning Representations*.
- Tanay, T. & L. Griffin (2016). “A Boundary Tilting Perspective on the Phenomenon of Adversarial Examples”. In: *arXiv*. DOI: [arXiv:1608.07690](https://arxiv.org/abs/1608.07690).

Bibliography

- Telgarsky, M. (2022). “Feature selection and low test error in shallow low-rotation ReLU networks”. In: *ICLR*.
- Timor, N., G. Vardi & O. Shamir (2023). “Implicit regularization towards rank minimization in relu networks”. In: *ALT*.
- Tramer, F. (2022). “Detecting Adversarial Examples Is (Nearly) As Hard As Classifying Them”. In: *Proceedings of the 39th International Conference on Machine Learning*. Ed. by K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu & S. Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, pp. 21692–21702. URL: <https://proceedings.mlr.press/v162/tramer22a.html>.
- Tropp, J. A. (2015). “An introduction to matrix concentration inequalities”. In: *Foundations and Trends® in Machine Learning* 8.1-2, pp. 1–230.
- Trouillon, T., J. Welbl, S. Riedel, É. Gaussier & G. Bouchard (2016). “Complex Embeddings for Simple Link Prediction”. In: *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. Vol. 48. Proceedings of Machine Learning Research, pp. 2071–2080. URL: <https://proceedings.mlr.press/v48/trouillon16.html>.
- Turner, N. L., L. Thiergart, G. Leech, M. Udell, J. Vazquez, et al. (2023). “Steering Language Models with Activation Engineering”. In: *arXiv preprint arXiv:2308.10248*. URL: <https://arxiv.org/abs/2308.10248>.
- Ustun, B. & C. Rudin (2017). “Optimized Risk Scores”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pp. 1125–1134. DOI: [10.1145/3097983.3098161](https://doi.org/10.1145/3097983.3098161).
- Valle-Pérez, G., C. M. Campos Camargo & A. A. Louis (2019). “Deep Learning Generalizes Because the Parameter-Function Map Is Biased Towards Simple Functions”. In: *International Conference on Learning Representations (ICLR)*.
- Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer.
- Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser & I. Polosukhin (2017). “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Vol. 30. URL: <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.
- Villegas, R., D. Erhan, F. Mahmood, et al. (2023). “Phenaki: Variable length video generation from open domain textual descriptions”. In: *International Conference on Learning Representations (ICLR)*.
- Virmaux, A. & K. Scaman (2018). “Lipschitz regularity of deep neural networks: analysis and efficient estimation”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi & R. Garnett. Vol. 31. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2018/file/d54e99a6c03704e95e6965532dec148b-Paper.pdf.
- Wachter, S., B. Mittelstadt & C. Russell (2018). “Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR”. In: *Harvard Journal of Law & Technology* 31.2, pp. 841–887.
- Wäldchen, S., J. MacDonald, S. Hauch & G. Kutyniok (2021). “The Computational Complexity of Understanding Binary Classifier Decisions”. In: *J. Artif. Intell. Res.* 70, pp. 351–387.
- Wand, M. P. & M. C. Jones (1995). *Kernel Smoothing*. Vol. 60. Monographs on Statistics and Applied Probability. London: Chapman and Hall. ISBN: 9780412552700.
- Wang, B., J. Gao & Y. Qi (2017a). “A Theoretical Framework for Robustness of (Deep) Classifiers against Adversarial Examples”. In: *arXiv*. DOI: [arXiv:1612.00334](https://doi.org/10.48550/arXiv.1612.00334).

Bibliography

- Wang, M. & C. Ma (2023). “Generalization Error Bounds for Deep Neural Networks Trained by SGD”. In: *arXiv preprint arXiv:2206.03299*.
- Wang, P., Y. Lei, D. Wang, Y. Ying & D.-X. Zhou (2023a). “Generalization Guarantees of Gradient Descent for Multi-Layer Neural Networks”. In: *arXiv preprint arXiv:2305.16891*.
- Wang, T., C. Rudin, F. Doshi-Velez, Y. Liu, E. Klampfl & P. MacNeille (2017b). “A Bayesian Framework for Learning Rule Sets for Interpretable Classification”. In: *Journal of Machine Learning Research* 18.70, pp. 1–37.
- Wang, Y., B. Dong, K. Xu, H. Piao, Y. Ding, B. Yin & X. Yang (2023b). “A Geometrical Approach to Evaluate the Adversarial Robustness of Deep Neural Networks”. In: *ACM Trans. Multimedia Comput. Commun. Appl.* 19.5s. ISSN: 1551-6857. DOI: [10.1145/3587936](https://doi.org/10.1145/3587936). URL: <https://doi.org/10.1145/3587936>.
- Wang, Z., H. Wang, S. Ramkumar, P. Mardziel, M. Fredrikson & A. Datta (2020). “Smoothed Geometry for Robust Attribution”. In: *Advances in Neural Information Processing Systems*. Vol. 33.
- Weitzner, D. & R. Giryes (2023). “On the Relationship Between Universal Adversarial Attacks and Sparse Representations”. In: *IEEE Open Journal of Signal Processing* 4, pp. 99–107. DOI: [10.1109/OJSP.2023.3244486](https://doi.org/10.1109/OJSP.2023.3244486).
- Wigner, E. (1955). “Characteristic vectors of bordered matrices with infinite dimensions”. In: *Annals of Mathematics* 62.
- Wilson, A. G., Z. Hu, R. Salakhutdinov & E. P. Xing (2016). “Deep Kernel Learning”. In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 51. Proceedings of Machine Learning Research. PMLR, pp. 370–378. URL: <https://proceedings.mlr.press/v51/wilson16.html>.
- Wilson, A. G. & P. Izmailov (2020). “Bayesian Deep Learning and a Probabilistic Perspective of Generalization”. In: *Advances in Neural Information Processing Systems* 33.
- Wu, Y., M. Jiang, Z. Huang, F. Chao & C. Zhou (2015). “An NP-complete fragment of fibring logic”. In: *Annals of Mathematics and Artificial Intelligence* 75.3-4, pp. 391–417. DOI: [10.1007/s10472-015-9468-4](https://doi.org/10.1007/s10472-015-9468-4).
- Xu, S., S. Venugopalan & M. Sundararajan (2020). “Attribution in Scale and Space”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9680–9689.
- Yang, A. X., M. Robeyns, E. Milsom, B. Anson, N. Schoots & L. Aitchison (2023). “A theory of representation learning gives a deep generalisation of kernel methods”. In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato & J. Scarlett. Vol. 202. Proceedings of Machine Learning Research. PMLR, pp. 39380–39415. URL: <https://proceedings.mlr.press/v202/yang23k.html>.
- Yeh, C.-K., J. S. Kim, I. E.-H. Yen & P. Ravikumar (2018). “Representer Point Selection for Explaining Deep Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*.
- Yu, F., Z. Qin, C. Liu, L. Zhao, Y. Wang & X. Chen (2019). “Interpreting and Evaluating Neural Network Robustness”. In: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, pp. 4199–4205. DOI: [10.24963/ijcai.2019/583](https://doi.org/10.24963/ijcai.2019/583). URL: <https://doi.org/10.24963/ijcai.2019/583>.
- Zavatone-Veth, J. A., W. L. Tong & C. Pehlevan (2022). “Contrasting random and learned features in deep Bayesian linear regression”. In: *Phys. Rev. E* 105 (6), p. 064118.

Bibliography

- DOI: [10.1103/PhysRevE.105.064118](https://doi.org/10.1103/PhysRevE.105.064118). URL: <https://link.aps.org/doi/10.1103/PhysRevE.105.064118>.
- Zeiler, M. D. & R. Fergus (2014). “Visualizing and Understanding Convolutional Networks”. In: *ECCV*. Springer, pp. 818–833.
- Zhang, C., S. Bengio, M. Hardt, B. Recht & O. Vinyals (2016). “Understanding deep learning requires rethinking generalization”. In: *arXiv preprint arXiv:1611.03530*.
- Zhang, C., S. Bengio, M. Hardt, B. Recht & O. Vinyals (2021). “Understanding deep learning (still) requires rethinking generalization”. In: *Communications of the ACM* 64.3, pp. 107–115.
- Zhao, C., P. T. Fletcher, M. Yu, Y. Peng, G. Zhang & C. Shen (2019). “The Adversarial Attack and Detection under the Fisher Information Metric”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01, pp. 5869–5876. DOI: [10.1609/aaai.v33i01.33015869](https://doi.org/10.1609/aaai.v33i01.33015869). URL: <https://ojs.aaai.org/index.php/AAAI/article/view/4536>.
- Zou, A., Z. Wang, A. Pan, et al. (2023). *Universal and Transferable Adversarial Attacks on Aligned Language Models*. arXiv: [2307.15043](https://arxiv.org/abs/2307.15043) [cs.CL].