Analyzing Training Dynamics of Deep Neural Networks: Insights and Limitations of the Neural Tangent Kernel Regime

Mariia Seleznova

Mai, 2024

Analyzing Training Dynamics of Deep Neural Networks: Insights and Limitations of the Neural Tangent Kernel Regime

Mariia Seleznova

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

Mai, 2024

Erstgutachter/in: Prof. Dr. Gitta Kutyniok
Zweitgutachter/in: Prof. Dr. Mikhail Belkin
Drittgutachter/in: Prof. Dr. Stefanie Jegelka

Tag der Einreichung: 27.05.2024
Tag der mündlichen Prüfung: 07.11.2024

Acknowledgements

I would like to thank my supervisor, Prof. Dr. Gitta Kutyniok, for the continuous support, guidance, and freedom to pursue research in the topics that interested me the most. Your kindness and understanding meant a lot to me during the challenging times that I faced after the start of the full-scale Russo-Ukrainian war.

I am also grateful to Prof. Dr. Mikhail Belkin and Prof. Dr. Stefanie Jegelka for agreeing to review this thesis, and to Prof. Dr. Eyke Hüllermeier and Prof. Dr. Björn Ommer for serving in the doctoral committee.

Furthermore, I am thankful to all my colleagues, many of whom have not only accompanied me throughout this PhD journey but have also become cherished friends. In particular, I would like to thank Edward (Dr. Hung-Hsu Chou) for the inspiring collaboration, which has contributed to some of the results presented in this thesis, and for his wonderful sense of humor. I would also like to thank Raffaele for all the meals that he cooked for me while I was writing this thesis, and to Stefan for his help in navigating the complexities of German bureaucracy.

I am grateful to my family and friends, who may not be familiar with the content of this thesis, yet have provided me with unwavering support and encouragement.

Last but not least, I am immeasurably grateful to the Armed Forces of Ukraine for protecting lives of my family and friends back home.

Zusammenfassung

Tiefe neuronale Netze (TNN) erzielen in unterschiedlichsten Anwendungsbereichen beeindruckende Resultate und dennoch bleibt ihre zugrundeliegende Funktionsweise größtenteils unverstanden. Empirische Beobachtungen wie die gute Generalisierbarkeit stark überparametrisierter Netze stehen im Widerspruch zur klassischen statistischen Lerntheorie. Das Training der Netzwerke kann aufgrund ihrer stark nicht-konvexen Verlustlandschaft nicht mit klassischer konvexer Optimierung erklärt werden. Das Verständnis dieser Phänomene erfordert die Entwicklung ganz neuer theoretischer Ansätze, und die Einführung des Neural Tangent Kernels (NTK) in Jacot et al. (2018) erwies sich als ein signifikanter erster Schritt in diese Richtung. Diese Methode analysiert die Netzdynamiken im sogenannten *"infinite-width limit"* (Netzwerke mit unendlicher Breite). Unter bestimmten Annahmen befinden sich diese Netzwerke dann im sogenannten *"NTK-Regime"*, das eine wichtige Rolle bei der theoretischen Analyse der Netzwerk-Generalisierbarkeit und Konvergenz spielt.

Zwar ermöglicht das NTK-Regime eine komplette Charakterisierung von Netzwerken mit unendlicher Breite, die Analyse lässt sich jedoch nicht direkt auf Netzwerke mit endlicher Breite übertragen. Ziel dieser Doktorarbeit ist es, die Möglichkeiten und Grenzen des NTK-Regimes für die Weiterentwicklung der Theorie des Deep Learning genauer zu beleuchten. Der erste Teil der Arbeit zeigt basierend auf zwei Artikeln, dass es von den Hyperparametern der zufälligen *Initialisierung* und dem *Tiefe-zu-Breite*-Verhältnis abhängt, ob sich ein vollverbundenes Netzwerk im NTK-Regime befindet. Hierbei wird die Bedeutung der Drei-Phasen-Initialisierung (erstmals erkannt von Poole et al. (2016)) – *"ordered"*, *"chaotic"*, und *"edge of chaos (EOC)"* – genauer analysiert. Eine konkrete Charakterisierung der NTK Streuung im *"infinite-depth-and-width limit"* (Netzwerke mit unendlicher Tiefe und Breite) wird in allen drei Phasen aufgezeigt. Die Ergebnisse belegen ein exponentielles Wachstum der NTK-Streuung mit der Netzwerktiefe in der EOC und der chaotischen Phase, jedoch nicht in der geordneten Phase. Zusätzlich zeigen wir, dass der NTK der tiefen Netzwerke während des Trainings nur in der geordneten Phase konstant bleibt. Die theoretisch erzielten Ergebnisse werden mit einer umfangreichen Simulationsstudie belegt.

Der zweite Teil der Arbeit beschäftigt sich mit einem neuen Ansatz zur Analyse der Netzwerkdynamiken, der auf der *NTK-Blockstruktur*-Annahme beruht. Diese Annahme motiviert sich aus dem *"NTK-Alignment"*-Phänomen, in dem sich der NTK von Netzwerken mit endlicher Breite während des Trainings an die Zielfunktion anpasst. In der Klassifikation mit TNN führt dies zu einer Blockstruktur in der Kernel-Matrix, bei der die Korrelationen zwischen Datenpunkten derselben Klasse stärker sind als zwischen Datenpunkten verschiedener Klassen. Durch die Annahme der NTK-Blockstruktur analysieren wir am Ende des Trainings die Dynamik von TNN, die mit dem mittleren quadratischen Fehler trainiert werden. Wir leiten die Dynamikgleichungen her, zerlegen sie in interpretierbare Phasen und identifizieren eine Dynamikinvariante. Unsere Analysen zeigen, dass das aus empirischen Studien bekannte Phänomen des *"Neural Collapse (NC)"* an speziellen Punkten in der Dynamik auftritt. Zudem beleuchten wir die erforderlichen Annahmen für die Konvergenz zum NC. Eine große Simulationsstudie belegt unsere theoretischen Erkenntnisse.

Summary

The widespread use of Deep Neural Networks (DNNs) in various application has underscored their effectiveness, yet the fundamental principles behind their success largely remain elusive. Despite being highly overparametrized, DNNs often exhibit effective generalization, defying predictions of classical statistical learning theory. Moreover, theoretical analysis of DNNs' training falls outside of the scope of classical convex optimization theory, since DNNs' loss landscapes are highly non-convex. Addressing these challenges requires novel approaches to studying DNNs' training dynamics. The introduction of the Neural Tangent Kernel (NTK) in Jacot et al. (2018) has been a significant step forward in this direction, as it greatly simplified the analysis of DNNs' dynamics in the infinite-width limit, where DNNs enter the so-called *NTK regime* under certain conditions. This regime has played a pivotal role in recent theoretical analyses of DNNs' generalization and convergence.

While the NTK regime allows to completely describe the infinite-width limit of DNNs, it cannot capture all the properties of realistic finite-width DNNs' training dynamics. Thus, the objective of this thesis is to determine possibilities and limitations of the NTK regime for advancing the theory of deep learning. The first part of the thesis, comprising two papers, focuses on the limitations of the NTK regime for the analysis of fully-connected DNNs. Namely, our contributions demonstrate that whether a network is in the NTK regime depends on the hyperparameters of random *initialization* and the network's *depth-to-width* ratio. Our results indicate the importance of the three phases of initialization, identified in Poole et al. (2016): *ordered*, *chaotic*, and the *edge of chaos (EOC)*. We derive exact expressions for the NTK dispersion in the *infinite-depth-and-width* limit in all three phases, and conclude that the NTK variability grows exponentially with depth at the EOC and in the chaotic phase but not in the ordered phase. Additionally, we show that the NTK of deep networks may stay constant during training only in the ordered phase. Our contributions also include large-scale numerical experiments, which fully support the theoretical findings.

The second part of the thesis introduces a novel approach to analyze DNNs' training dynamics based on the NTK *block-structure* assumption. This assumption is motivated by the *NTK alignment* phenomenon, where the NTK of finite-width DNNs aligns with the target function during training. For classification DNNs, this alignment gives rise to an approximate block-structure in the kernel matrix, where the correlations between samples from the same class are stronger than between samples from different classes. We employ the NTK block-structure assumption to analyze the dynamics of DNNs trained with mean squared (MSE) loss at the end of training. Namely, we derive the dynamics equations, break the dynamics into interpretable phases, and identify a dynamics invariant. Our analysis reveals that a prominent empirical phenomenon called *Neural Collapse (NC)* occurs in certain fixed points of this dynamics, and provides necessary conditions for convergence to NC. We provide large-scale numerical experiments on three common DNN architectures and three benchmark datasets to support our theory.

Contents

Chapter 1

Introduction

Over the past two decades, Deep Neural Networks (DNNs) have pushed forward the state of the art in a wide range of applications. A prime example is image recognition, where DNNs have long surpassed all other algorithms, and even human performance, by a large margin (Krizhevsky et al., 2012; Szegedy et al., 2015; He et al., 2015). Another example, especially relevant at the time of writing this thesis, is natural language processing, where large language models like BERT and GPT have recently come into the global spotlight (Brown et al., 2020; Vaswani et al., 2017). Other areas where DNNs have shown impressive success include drug discovery (Zhavoronkov et al., 2019), predicting folding behaviour of proteins (Jumper et al., 2021), playing board games (Silver et al., 2016) and computer games (Mnih et al., 2013), and many more. Embraced by scientists, companies, and governments, DNNs have rapidly gained influence across diverse aspects of life.

As with great power comes great responsibility, the pervasive adoption of DNNs also caused significant criticism: many authors voiced concerns about the use of deep learning for safety-critical fields, such as autonomous driving and robotics, and for high-stakes decision making, such as medicine or criminal justice (Rudin, 2019; Willers et al., 2020). The criticism primarily stems from the fact that modern DNNs operate as black box models, i.e., it is currently impossible to provide meaningful performance guarantees for these models and reliably explain their predictions to humans. This lack of trustworthiness is critical when a model's error can cause significant harm, which makes large-scale adoption of DNNs in safety-critical applications problematic. As a reflection on this criticism, the machine learning community has embraced the challenge of "opening the black box" of deep learning, driving progress in fields such as explainability, fairness, and the mathematical foundations of deep learning. This thesis is a contribution to the latter field.

1.1 Challenges of Deep Learning Theory

Let us now delve into the question of why DNNs are considered black box models. Namely, what makes it challenging to derive performance guarantees for DNNs? In fact, as we will see in Section 2.1.2, even simple DNNs form families of functions that are rich enough to approximate any continuous function with arbitrary accuracy. Therefore, in theory, there usually exists a DNN with an appropriate choice of parameters that is guaranteed to exhibit high performance on a given problem. However, in practice, reliably identifying such a DNN is usually impossible due to the following reasons:

1. **Generalization:** In practical scenarios, the target function that a DNN aims to approximate is unknown. Instead, the network is given a finite dataset comprising (potentially noisy) samples of the target function. Naturally, the choice of the underlying function based on a finite dataset is not unique. Therefore, even if a DNN accurately fits the provided dataset, called the *training set*, its performance on new data, called the *test set*, may vary. The field of machine learning theory addressing how well a model performs on unseen data is known as *generalization*. While classical machine learning models, like linear models or kernel models, have well-established generalization theory and performance guarantees, deriving similar guarantees for DNNs has proven to be highly challenging. We define fundamental concepts and review literature regarding generalization of DNNs in Section 2.3.

2. **Optimization:** Parameters of DNNs are chosen by optimizing a given *loss* function, which quantifies the network's error on the training set. This process is known as *training*. DNNs are typically trained using a variant of Gradient Descent (GD) algorithm, which is a first-order numerical optimization algorithm with strong performance guarantees for convex optimization problems. However, optimization problems associated with training of DNNs are highly non-convex, always have multiple global minima, and may include undesirable local minima and saddle points. Moreover, the convergence point of the GD algorithm may significantly depend on initialization, which is usually random in practice. While recent research suggests that modern DNNs may have properties favourable for GD's convergence to a global minimum under certain conditions, it remains challenging to verify such conditions and to characterize the solutions identified by GD in a given setting. We introduce basic concepts related to DNNs' training and discuss relevant literature in Section 2.2.

Hence, the current lack of a comprehensive theory for generalization and training of DNNs makes it impossible to provide meaningful performance guarantees for DNNs used in practical applications. This thesis focuses on one research direction that contributes to the development of such a theory: the study of DNNs' training dynamics. In this section, we briefly review the current advances towards a theory of generalization and training of DNNs, as well as open problems, focusing on two relevant concepts: *overparametrization* and *implicit bias*.

1.1.1 Overparametrization

An important characteristic of modern DNNs, particularly relevant for generalization and optimization, is their *overparametrization*. Overparametrized models possess sufficient complexity to exactly fit any dataset of a given size. Therefore, overparametrized DNNs typically achieve near-zero training loss values, even when trained on a dataset with completely random target labels (Zhang et al., 2021).

Optimization landscapes of overparametrized DNNs typically include entire manifolds of *interpolating* global minima (i.e., solutions that exactly fit the dataset) (Cooper, 2021), and are non-convex even locally around these global minima (Liu et al., 2022). While the loss landscapes of overparametrized DNNs do not have strict local minima under weak conditions, they often include non-strict local minima and saddle points (Nguyen et al., 2018; Li et al., 2018). Given these characteristics, it is evident that properties of overparametrized DNNs trained using variants of GD may significantly depend on various factors related to the optimization process, such as the choice of initialization or the training algorithm.

Overparametrized DNNs also pose new theoretical challenges in the field of generalization. Classical statistical learning theory predicts poor generalization guarantees for these models, as traditional bounds on generalization tend to degrade with increasing model complexity. However, empirical evidence contradicts this expectation, indicating that gradient descent training of overparametrized DNNs frequently results in models with robust generalization performance on real-world data. As we discuss in Section 2.3, the modern perspective on generalization of overparametrized models aligns more closely with the *double descent* phenomenon (Belkin et al., 2019). This phenomenon suggests that generalization performance deteriorates with model complexity only until the interpolation threshold, after which it improves again with further overparametrization. Empirical studies have demonstrated double descent across a wide range of models, including DNNs (Nakkiran et al., 2021). Theoretical works have also established the occurrence of double descent in different machine learning models, such as linear models (Hastie et al., 2022) or random features models (Belkin et al., 2020; Mei and Montanari, 2022). However, there is currently no established theoretical framework to prove double descent in the context of DNNs.

As we discuss in Section 2.3, theoretical results regarding generalization of overparametrized models usually rely on a predefined *data model* and a known convergence point of a given *training algorithm*. This is in contrast with the classical statistical learning theory bounds, which are independent of both the data distribution and the training procedure. Given that overparametrized models can perfectly fit datasets with completely random labels, it is clear that meaningful generalization guarantees may require certain data-related assumptions. Moreover, the presence of entire manifolds of global minimizers, not all of which generalize equally well, highlights the importance of algorithm-related assumptions. Indeed, these assumptions determine which of the training loss minimizers are relevant for generalization of the trained model. This observation provides the crucial connection between training dynamics and generalization, which we discuss in more detail in the next section.

1.1.2 Implicit Bias

The questions of generalization and optimization are intimately related in the overparametrized setting: the generalization performance of a trained model is determined by the solution, to which the training algorithm converges. Therefore, the literature uses the notion of *implicit bias*, which is often informally defined as the tendency of gradient-based training algorithms to favor solutions that exhibit good generalization. However, since the generalization performance ultimately depends on the unknown target function, implicit bias is more accurately defined as convergence of gradient-based training algorithms to solutions with specific mathematical properties.

Deriving implicit bias results for a given model usually involves explicit analysis of the model's training dynamics. Implicit bias of overparametrized linear models, such as the least squares and the logistic regression, has been extensively explored due to the relatively tractable training dynamics equations of such models (Gunasekar et al., 2018; Soudry et al., 2018). Remarkably, the results regarding implicit bias of linear models reveal important differences between two loss functions, popular in machine learning: *mean squared error* (MSE) loss and *cross-entropy* (CE) loss (defined in Section 2.2). In case of MSE loss, it is well-known that GD training of overparametrized models converges to the global minimum closest to the initialization in terms of ℓ_2 norm. Conversely, CE loss has no attainable global minima, and tends to zero as the parameters vector tends to infinity. Therefore, GD training with CE loss diverges. However, the direction of the diverging parameters vector tends to the maximal-margin vector and is independent of initialization.

Implicit bias has also been extensively studied for linear DNNs, i.e., DNNs without a non-linear activation function. In this context, the picture is somewhat similar to linear models: CE loss results in implicit bias towards maximal-margin solutions, independently of initialization (Ji and Telgarsky, 2020). On the other hand, the implicit bias of GD with MSE loss significantly depends on the initialization and is generally more challenging to derive (Yun et al., 2020; Azulay et al., 2021). A related research direction studies the implicit bias of deep matrix factorization, which is analogous to the training of linear DNNs with no bias terms. The results in this area typically show implicit bias towards low-rank solutions (Chou et al., 2024; Bah et al., 2022; Li et al., 2020).

Despite the aforementioned results, the current understanding of implicit bias in non-linear DNNs is very limited. There exist some results on implicit bias of homogeneous DNNs, such as fully-connected ReLU DNNs with no bias terms, trained using CE loss (Lyu and Li, 2019; Ji and Telgarsky, 2020). However, implicit bias of non-linear DNNs with MSE loss poses theoretical challenges even in the simplest case of single-neuron ReLU networks (Vardi and Shamir, 2021). This field of study is particularly challenging because the training dynamics of DNNs are governed by complex non-linear equations, which generally defy analytical treatment. Nevertheless, recent studies have demonstrated that, under certain conditions, non-linear DNNs enter the so-called *lazy training* or *kernel regime*, where the training dynamics are linearized around the initialization (Chizat et al., 2019; Woodworth et al.,

2020). In the kernel regime, it is possible to derive implicit bias and generalization results for non-linear DNNs. However, these results may not necessarily describe the behavior of realistic DNNs. As a preview of the upcoming sections, we announce here that the main contributions of this thesis largely concern the limitations of the kernel regime for the study of DNNs' training dynamics.

1.2 Current Approaches to Study DNNs

As we outlined in the previous section, developing a theory for training and generalization of DNNs is highly challenging. Since the complex non-linear training dynamics of DNNs cannot be studied analytically in the general case, the current theory relies on various simplifications and special cases. Common simplified settings, used in the literature as proxies to study DNNs, can be divided into the following broad categories:

1. **Linear:** Numerous papers study linear DNNs, i.e., networks with identity activation function (Laurent and Brecht, 2018; Ji and Telgarsky, 2018). While mathematical analysis of such networks is much more accessible, properties of linear DNNs do not generally transfer to non-linear DNNs, as we saw in the discussion on implicit bias. It is also clear that linear DNNs can only approximate linear functions, hence, they are not used in deep learning practice.

2. **Homogeneous:** Another research direction considers homogeneous DNNs, i.e., networks with homogeneous activation functions, no skip connections, and no bias terms. Turns out that homogeneity significantly simplifies the training dynamics both in case of CE loss (Lyu and Li, 2019; Ji and Telgarsky, 2018) and MSE loss (Poggio and Liao, 2019). While homogeneous activation functions, such as ReLU, are common in deep learning practice, the complete absence of both bias terms and skip connections is uncommon. Indeed, homogeneous DNNs can only approximate homogeneous functions, which limits possible applications for such networks.

3. **Shallow:** Training dynamics of non-linear networks are often considered in the special case of shallow networks, i.e., networks with one or two hidden layers (Mei and Montanari, 2022; Arora et al., 2019a). While such results may point towards new insights concerning neural networks' dynamics, they do not provide any understanding of the effects of depth in deep learning, and usually cannot be easily generalized to arbitrary depth.

4. **Very wide:** Another recent line of research considers DNNs in the *infinite-width limit*, where the number of neurons in each layer tends to infinity (Lee et al., 2019, 2018). Remarkably, it is possible to derive explicit infinite-width limits for the training dynamics of DNNs in a wide range of settings (Yang and Hu, 2022). Some of the most notable results in this direction pertain to the *kernel regime* of DNNs, where the dynamics are linearized around the initialization and are governed by the so-called *Neural Tangent Kernel* (NTK) (Jacot et al., 2018; Yang, 2020b). While modern

overparameterized DNNs are often "very wide", the question of whether a given DNN can be approximated by a certain infinite-width limit is still largely open.

While each of the aforementioned approaches to study DNNs has its limitations, the constraints of the last approach are perhaps the least clear. Indeed, modern neural networks are usually *not* linear, *not* homogeneous, and *not* shallow. However, they often have very large width. In this thesis, we focus of insights and limitations of the last approach and, in particular, the kernel regime of DNNs.

1.2.1 Neural Tangent Kernel

In this section, we define the kernel regime of infinitely-wide DNNs and the associated kernel, called the Neural Tangent Kernel (NTK). To this end, let us denote the output function of a DNN $f_\theta : \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}^1$. Here $n_{\text{in}} \in \mathbb{N}$ is the input dimension and $\theta \in \mathbb{R}^P$ are the network's parameters. Then the NTK of this network is given by

$$\Theta(x_i, x_j) := \left\langle \nabla_\theta f_\theta(x_i), \nabla_\theta f_\theta(x_j) \right\rangle, \quad x_i, x_j \in \mathbb{R}^{n_{\text{in}}}, \tag{1.1}$$

where $\nabla_\theta f_\theta$ is the gradient of the output function with respect to the parameters θ. Assume further that the network is trained on a dataset $S = \{(x_i, y_i)\}_{i=1}^N$, with loss function $\mathcal{L}$, and learning rate $\eta > 0$. Then the following holds for a GD step of the output function:

$$\Delta f_\theta(x) = -\eta \sum_{(x_i, y_i) \in \mathcal{S}} \Theta(x, x_i) \frac{\partial \mathcal{L}}{\partial f_\theta}(x_i, y_i) + O(\eta^2), \tag{1.2}$$

where $\Delta f_\theta(x)$ represents the change in the value of $f_\theta(x)$ during the GD step. In other words, the NTK controls the first order approximation of the DNNs' GD dynamics.

Since the NTK depends on the DNN's parameters, it changes during training, and inherits randomness from the random initialization and any other stochastic elements of the training process. Consequently, the dynamics described in equation (1.2) generally defy analytical solutions. However, a famous result by Jacot et al. (2018) states that, in the infinite-width limit, the NTK is deterministic under proper random initialization and remains constant during training. Therefore, the dynamics in (1.2) reduce to kernel regression in this limit, and have an analytical solution expressed in terms of the NTK. We will call this setting the *kernel regime* or the *NTK regime* of DNNs. We discuss the NTK regime in detail in Section 2.4.

Within the kernel regime, it becomes feasible to study convergence and generalization of DNNs theoretically by analyzing the properties of the NTK at initialization. Therefore, many recent works proposed to study the NTK regime to gain new insights into DNNs' behavior (Huang et al., 2020; Adlam and Pennington, 2020; Wang et al., 2022; Bietti and Mairal, 2019; Tirer et al., 2022; Geiger et al., 2020). Numerous contributions also derived

[1] We consider NNs with scalar output here for simplicity of notation. However, the definition of the NTK and our discussion is naturally generalized to NNs with multidimensional output.

expressions for the infinite-width NTK of popular DNN architectures (Yang, 2020b; Du et al., 2019; Alemohammad et al., 2020). Other papers established bounds on the DNNs' width that ensure sufficient concentration of the NTK at initialization (Arora et al., 2019b; Buchanan et al., 2021) and stability of the NTK during training (Huang and Yau, 2020; Lee et al., 2019).

However, multiple authors have argued that the NTK regime and, in general, the infinite-width limit cannot explain the success of DNNs (Chizat et al., 2019; Hanin and Nica, 2019; Aitchison, 2020; Li et al., 2021; Huang and Yau, 2020). The primary argument in this context is that a constant NTK implies that DNNs do not learn new features during the training process within the kernel regime. Moreover, the NTK at initialization is *label-agnostic*, i.e., the value of $\Theta(x_i, x_j)$ does not depend on the target outputs (y_i, y_j). This property renders the NTK regime inadequate for explaining DNNs' capability to perform well on various tasks using the same dataset (Chen et al., 2020). Finally, numerous empirical results also demonstrated that there is often a performance gap between trained DNNs and their kernel regimes (Fort et al., 2020; Lee et al., 2020). Therefore, understanding the nature of the NTK regime of DNNs and its applicability is an important question of deep learning theory.

1.3 Contributions

The objective of this thesis is to determine possibilities and limitations of the NTK regime for advancing the theory of deep learning. The contributions of the thesis are published as the following papers:

1. *Analyzing Finite Neural Networks: Can We Trust Neural Tangent Kernel Theory?* (Seleznova and Kutyniok, 2022a) — included in Section 3.1.
2. *Neural Tangent Kernel Beyond the Infinite-Width Limit: Effects of Depth and Initialization* (Seleznova and Kutyniok, 2022b) — included in Section 3.2.
3. *Neural (Tangent Kernel) Collapse* (Seleznova et al., 2023) — included in Section 3.3.

The first two papers mainly focus on the limitations of the NTK regime for the analysis of realistic DNNs. The last paper proposes a new approach to analyze DNNs' dynamics using the kernel regime at the end of training, and employs this approach to explain a prominent empirical phenomenon observed in well-trained DNNs.

1.3.1 Limitations of the NTK Regime

The following two observations were the starting point for our research into the limitations of the NTK regime for the analysis of realistic DNNs:

- **Depth**: The infinite-width limit setting, where the NTK becomes constant and deterministic, assumes that the depth of a DNN is fixed while the width tends to

infinity. However, several papers demonstrated that infinite-width approximations of DNN's statistics at initialization often get worse as the network's depth increases (Li et al., 2021; Hanin and Nica, 2019; Hu and Huang, 2021). Such results typically consider the *infinite-depth-and-width limit*, where both depth and width of a DNN tend to infinity simultaneously with a given depth-to-width ratio. In particular, Hanin and Nica (2019) were the first to show that the NTK is random and changes during training in the infinite-depth-and-width limit for fully-connected ReLU DNNs under a certain initialization setting.

- **Initialization:** The initialization may significantly change the behavior of DNNs in the infinite-width limit, and even determine whether a DNN is in the kernel regime (Yang and Hu, 2022). However, previous works on the infinite-depth-and-width limit of the NTK do not clarify the effects of initialization in this setting. According to Poole et al. (2016), there are three phases with distinct properties in the initialization hyperparameters space: *ordered*, *chaotic* and the *edge of chaos* (EOC). In the infinite-width limit, the chaotic phase roughly corresponds to initialization settings where the gradients grow with the DNN's depth. Conversely, gradients decrease with growing depth in the ordered phase. EOC is the initialization at the border between these phases. Given this interpretation, the three phases of initialization should be significant for the statistical properties of the NTK of deep networks.

Hence, our research focused on exploring the *combined* effects of depth and initialization on the NTK of fully-connected DNNs. Our contributions in this research direction are summarized as follows:

- **Variability of the NTK at initialization:** We precisely characterized the dispersion of the NTK in the infinite-depth-and-width limit for fully-connected ReLU DNNs in Seleznova and Kutyniok (2022b). Our results indicate that the variability of the NTK grows exponentially with the depth-to-width ratio at the EOC and in the chaotic phase. On the other hand, the variance of the NTK tends to zero in the ordered phase independently of depth. Therefore, the NTK of deep networks is approximately deterministic at initialization only in the ordered phase. While these theoretical results are derived for ReLU DNNs, we empirically showed analogous results for sigmoid DNNs in Seleznova and Kutyniok (2022a). In addition, we provided non-asymptotic expressions for the first two moments of the NTK, and discussed the *finite-width effects* that follow. Our results significantly improve on the previous works, which could only provide non-tight bounds for the NTK dispersion, and did not consider different initialization settings. Furthermore, in contrast to the prior studies, we carried out extensive numerical experiments that thoroughly validate the correctness of our theoretical expressions.

- **Training dynamics of the NTK:** We proved that the expected relative change of the NTK value *during the first GD step* tends to infinity with depth in the chaotic phase, to zero in the ordered phase, and grows exponentially with the depth-to-width ratio at the EOC (Seleznova and Kutyniok, 2022b). Therefore, the NTK of deep

networks can stay constant during training only in the ordered phase. While our theoretical results focus on ReLU DNNs, we provide similar empirical results for sigmoid DNNs in Seleznova and Kutyniok (2022a). Namely, we show that the relative change in the NTK matrix norm is generally much larger in the chaotic phase than in the ordered phase. However, our experiments also indicate that the NTK *structure* changes non-trivially during training even in the ordered phase. Indeed, as we discuss in Seleznova and Kutyniok (2022b) and Seleznova et al. (2023), the NTK typically aligns with the target function during training. In the context of classification, this alignment manifests as an emergence of a block structure in the NTK matrix.

- **Generalization in the NTK regime:** We discuss the limitations of the NTK regime for the analysis of DNNs' generalization in Seleznova and Kutyniok (2022a). Our main observation in this discussion is that properties of the infinite-width NTK become unnatural as the depth increases. In particular, data-dependence of the infinite-width NTK matrix vanishes with depth, as also demonstrated in Xiao et al. (2020). Moreover, the infinite-width NTK matrix tends to a rank one matrix with depth in case of initialization in the ordered phase, which makes deep infinite-width networks untrainable in the ordered phase. Generalization analysis based on the NTK with such properties suggests poor generalization performance for deep networks, which does not agree with the empirical evidence.

1.3.2 Kernel Regime with Block-Structured NTK

The starting point for the second part of this thesis were the following observations regarding the *empirical* NTK of realistic DNNs:

- **NTK alignment:** While the infinite-width NTK does not change during training and does not depend on the target function, the empirical NTK aligns with the target function during training (Atanasov et al., 2021; Baratin et al., 2021; Shan and Bordelon, 2022; Seleznova and Kutyniok, 2022b). In other words, values of $\Theta(x_i, x_j)$ become aligned with $\langle y_i, y_j \rangle$, where $y_{i,j}$ are the target outputs for inputs $x_{i,j}$. The kernel-target alignment has long been seen as favourable for generalization of kernel methods in the literature (Cristianini et al., 2001). Therefore, NTK alignment could also provide insights regarding performance of trained DNNs
- **Rapid kernel learning:** Empirical evidence shows that the NTK aligns with the target function most rapidly during the early stages of training (Fort et al., 2020; Atanasov et al., 2021; Baratin et al., 2021). According to Fort et al. (2020), the performance of the empirical NTK after the initial rapid kernel learning phase essentially matches the performance of the fully-trained DNN. Theoretical study of simplified models in Atanasov et al. (2021) supports this conclusion. Therefore, it could be more appropriate to describe DNNs dynamics using the kernel regime *in the end of training*, where the NTK has developed its final structure.

Based on these observations, we proposed an approach to study the end-of-training dynamics

of DNN classifiers with NTK alignment in Seleznova et al. (2023). In classification problems, NTK alignment corresponds to the emergence of an approximate *block structure* in the NTK matrix, where the correlations between samples from the same class are stronger than between samples from different classes. In our work, we considered a simplified model of NTK alignment, where the kernel takes only three distinct values: an inter-class value, an intra-class value, and a diagonal value. Our contributions in this work are summarized as follows:

- **Gradient flow with block-structured NTK:** We derived and analyzed Gradient Flow (GF) dynamics of the last two layers of a DNN trained under MSE loss, assuming that the NTK in these layers is block-structured. In particular, we identified three distinct convergence rates in the dynamics, which correspond to three components of the training error: error of the global mean, of the class means, and of each individual sample. Moreover, we derived an *invariant* of the dynamics, which determines the properties of the convergence point.
- **Neural Collapse:** We proved that, under certain conditions, the GF dynamics of DNNs with block-structured NTK exhibits a prominent end-of-training phenomenon of modern DNNs, called *Neural Collapse* (NC) (Papyan et al., 2020). During NC, the class means of the DNN's last-layer features form a symmetric structure with maximal separation angle, and the features of each individual sample collapse to their class means. While the effects of NC for generalization of DNNs are not entirely clear (Kothapalli, 2023), maximal separation between classes is usually considered favourable for generalization in the literature (Jiang et al., 2018; Cisse et al., 2017). Our results provide the first theoretical connection between NTK alignment and NC. Moreover, they demonstrate the effectiveness of the kernel regime in the end of training for predicting behaviour of realistic DNNs.

1.4 Outline

Chapter 2 serves as the technical foundation for the contributions of this thesis. It begins with Section 2.1, which defines fully-connected neural networks, explores other prevalent architectures, and discusses the approximation capabilities of DNNs. Section 2.2 delves into the essential aspects of DNN training, including gradient descent and backpropagation, and discusses the effects of random initialization on DNNs' statistical properties. In Section 2.3, fundamental principles of generalization theory are discussed, encompassing classical results and modern perspectives. Section 2.4 defines the NTK, provides the fundamental results regarding the NTK regime of DNNs, and discusses the NTK alignment phenomenon.

Chapter 3 includes the individual publications forming this thesis. Section 3.1 corresponds to Seleznova and Kutyniok (2022a). Section 3.2 is published as Seleznova and Kutyniok (2022b). Section 3.3 corresponds to Seleznova et al. (2023).

Chapter 4 provides concluding remarks and explores potential directions for future work.

Chapter 2

Background and Foundations

This chapter furnishes the technical background relevant to the thesis. In Section 2.1, we provide a definition of neural networks. Section 2.2 explores common methods employed in the training of DNNs and discusses the effects of initialization on the properties of DNNs at initialization. The concept of generalization is formalized and examined in Section 2.3, followed by the definition of the Neural Tangent Kernel (NTK) and its significance in the theory of DNNs' training dynamics in Section 2.4. While the notation is generally introduced in the main text, a concise summary is also provided in Section 2.5.

2.1 (Deep) Neural Networks

Artificial Neural Networks (NNs), which are discussed in this thesis, find their roots in mathematical models of *biological* neural networks. These models were initially conceived as a representation of interconnected neurons within animal brains. Due to this historical connection, NNs are often visualized as computational graphs, composed of interconnected nodes called *neurons*. In this framework, each neuron receives inputs from neurons in its neighbourhood, processes the inputs, and transmits the output to other neurons. Figure 2.1 gives an example of a NN represented as a computational graph. The particular structure and size of the computational graph define the NN's *architecture*. In many common NN architectures, neurons are organized in *layers*, such that neurons of a given layer can receive input only from the previous layers, and can pass the output only to the following layers. *Depth* of such NNs is defined as the number of layers. There is no universal answer to the question of how many layers a NN should have to be classified as "deep". One of the contributions of this thesis demonstrates that the answer, in fact, may depend not only on the layer count but also on factors such as the depth-to-width ratio. Nevertheless, it is worth noting that depths of modern NNs are typically in the order of 10^1 to 10^2, which certainly qualifies them as Deep NNs (DNNs).

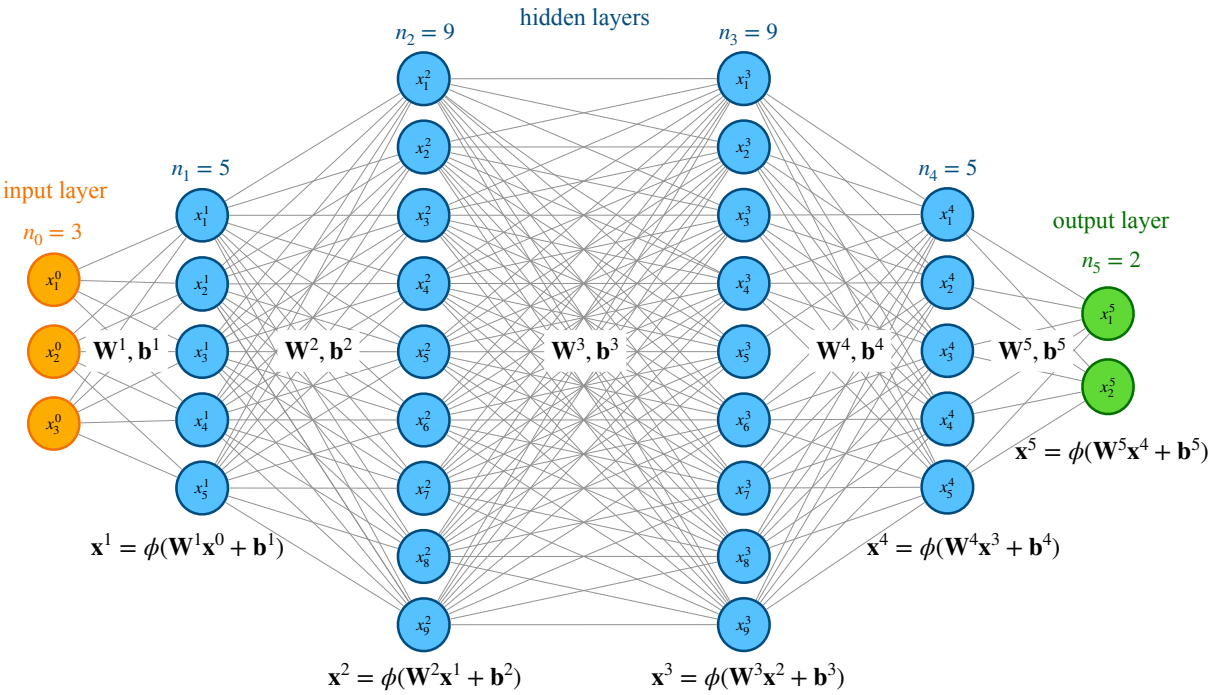


Figure 2.1: An example of a fully-connected feedforward NN with depth $L = 5$, input dimension $n_0 = 3$, hidden layers widths $\{n_\ell\}_{\ell=1}^4 = (5, 9, 9, 5)$, and output dimension $n_5 = 2$. Each layer $\ell, 1 \leq \ell \leq 5$ performs a composition of an affine-linear function, parametrized by weights matrix $\mathbf{W}^\ell$ and biases vector $\mathbf{b}^\ell$, and an activation function ϕ.

Artificial neuron From a mathematical perspective, a NN is a parametric function defined via a computational graph. In this context, the NN's *parameters* characterize the functions associated with each neuron. The function realized by a single artificial neuron, which is a building block of NNs, is given in the following definition:

Definition 2.1 (Artificial Neuron, Activation, Pre-Activation)**.** *Artificial neuron is a function* $a_\theta : \mathbb{R}^n \to \mathbb{R}^1$*, formed through the composition of an affine-linear function* $h_\theta : \mathbb{R}^n \to \mathbb{R}$ *and a non-linear function* $\phi : \mathbb{R} \to \mathbb{R}$*, called the activation function:*

$$a_\theta(\mathbf{x}) := \phi(h_\theta(\mathbf{x})), \quad h_\theta(\mathbf{x}) := \langle \mathbf{w}, \mathbf{x} \rangle + b, \quad \mathbf{x} \in \mathbb{R}^n. \tag{2.1}$$

Here the affine-linear function is parametrized by $\mathbf{w} \in \mathbb{R}^n$*, called the weights vector, and* $\mathbf{b} \in \mathbb{R}$*, called the bias. The set of parameters is given by* $\theta := (\mathbf{w}, \mathbf{b}) \in \mathbb{R}^n \times \mathbb{R} \simeq \mathbb{R}^{n+1}$*. The value of* $h(\mathbf{x})$ *is called pre-activation, and the value of* $a(\mathbf{x})$ *is called activation of the artificial neuron.*

[1]In Definition 2.1 and throughout the thesis, we consider NNs with *real* inputs, outputs and parameters, since the contributions of this thesis concern only real NNs. However, there exists literature on NNs that employ different number systems, such as complex numbers or quaternions (Lee et al., 2022; Parcollet et al., 2020).

Fully-connected NNs Given the definition of a neuron, it is possible to construct definitions of NNs with various architectures. Suppose a NN is defined by a computational graph, where neurons are organized into layers, such that neurons of each layer receive inputs from all the neurons of the previous layer, and transmit outputs to all the neurons of the next layer. Here the first layer, which receives the input of the NN, is called the *input layer*, and the last layer, which represents the model's output, is called the *output layer*. The remaining layers are called the *hidden layers*. An example of such a network is given in Figure 2.1. Additionally, suppose that all the weights and biases associated with neurons of this network are independent parameters. Then NNs with this architecture are called *fully-connected feedforward* NNs, and can be formally defined as follows:

Definition 2.2 (Fully-Connected Neural Network)**.** *A fully-connected feedforward NN with depth $L \in \mathbb{N}$, input dimension $n_0 \in \mathbb{N}$, hidden layers' widths $\{n_\ell\}_{\ell=1}^{L-1} \in \mathbb{N}^{L-1}$, output dimension $n_L \in \mathbb{N}$, and activation functions $\phi_\ell : \mathbb{R} \to \mathbb{R}, 1 \leq \ell \leq L$, is given by the following function:*

$$f_\theta(\mathbf{x}^0) := \big(\phi_L \odot \mathcal{A}_L \circ \phi_{L-1} \odot \mathcal{A}_{L-1} \circ \cdots \circ \phi_1 \odot \mathcal{A}_1\big)(\mathbf{x}^0) \in \mathbb{R}^L, \quad \mathbf{x}^0 \in \mathbb{R}^{n_0}, \tag{2.2}$$

where $\mathcal{A}_\ell : \mathbb{R}^{n_{\ell-1}} \to \mathbb{R}^\ell$ are affine-linear functions parametrized by weights $\mathbf{W}^\ell \in \mathbb{R}^{n_{\ell-1} \times n_\ell}$ and biases $\mathbf{b}^\ell \in \mathbb{R}^{n_\ell}$, i.e,

$$\mathcal{A}_\ell(\mathbf{x}) := \mathbf{W}^\ell \mathbf{x} + \mathbf{b}^\ell, \quad \mathbf{x} \in \mathbb{R}^{n_{\ell-1}}, \quad 1 \leq \ell \leq L, \tag{2.3}$$

and activation functions $\phi_\ell, 1 \leq \ell \leq L$, are applied to vectors element-wise. The parameters of the fully-connected feedforward NN are given by

$$\theta := \Big\{\big(\mathbf{W}^\ell, \mathbf{b}^\ell\big)\Big\}_{1 \leq \ell \leq L} \in \bigtimes_{\ell=1}^{L} \big(\mathbb{R}^{n_\ell \times n_{\ell-1}} \times \mathbb{R}^{n_\ell}\big) \simeq \mathbb{R}^P, \tag{2.4}$$

where $P := \sum_{\ell=1}^{L} n_\ell(n_{\ell-1} + 1) \in \mathbb{N}$ is the total number of parameters.

The term "fully-connected" in the above definition indicates that all the pairs of neurons from adjacent layers are connected, while the term "feedforward" implies that the information flows through the NN in a single direction – from input to output, meaning that there are no cycles in the computational graph. In the following discussion, we will refer to this architecture simply as "fully-connected NNs" when it does not lead to confusion. Definition 2.2 accurately characterizes such a network, since each vector-valued affine-linear function $\mathcal{A}_\ell$ is applied to the output of layer $\ell - 1$, and represents all the n_ℓ real-valued functions associated with neurons in layer $\ell \in [1, L]$. The corresponding weights matrices $\mathbf{W}^\ell \in \mathbb{R}^{n_{\ell-1} \times n_\ell}$ contain weights vectors of all the neuron in layer $\ell \in [1, L]$ as rows, and the bias vectors $\mathbf{b}^\ell \in \mathbb{R}^{n_\ell}$ contain scalar biases of these neurons as entries.

Fully-connected NNs can be considered the simplest and the most universal NN architecture, as they do not exploit any inherent structure in the input data. Despite their simplicity, these architectures pose numerous open theoretical challenges, as outlined in Chapter 1.

Given that the theoretical contributions of this thesis concentrate on fully-connected NNs, the subsequent sections offer formal definitions and results exclusively for this architecture. However, we provide a brief overview of popular NN architectures in the Section 2.1.1.

Activation functions Fully-connected NNs often use the same activation function in all the hidden layers, and a different activation function in the output layer. The common activation function choices include *sigmoid function*, *Rectified Linear Unit* (ReLU), and their modifications.

Definition 2.3 (Sigmoid Activation Function)**.** *Sigmoid activation function* $\phi : \mathbb{R} \to \mathbb{R}$ *is given by*

$$\phi(x) = \frac{1}{1 + e^{-x}}. \tag{2.5}$$

Definition 2.4 (ReLU Activation Function)**.** *ReLU activation function* $\phi : \mathbb{R} \to \mathbb{R}$ *is given by*

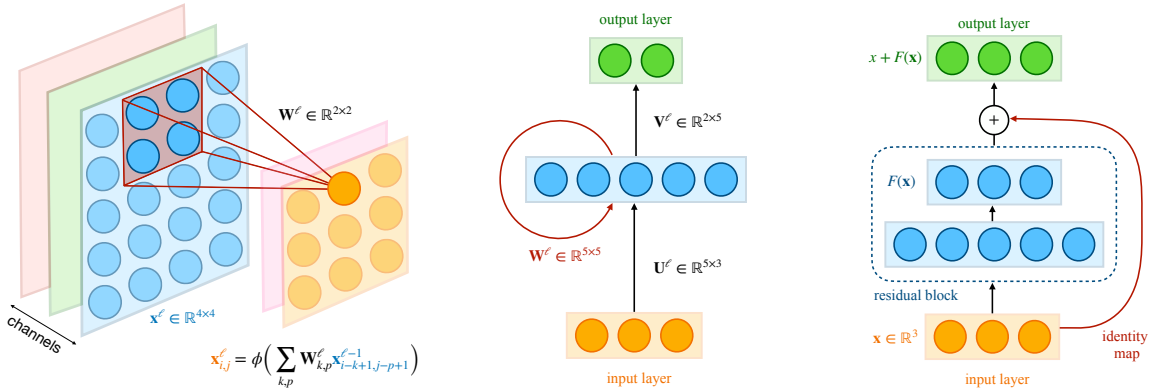
$$\phi(x) = \max\{0, x\}. \tag{2.6}$$

As we will see in Section 2.1.2, the activation function choice has minimal influence on the classical results regarding the approximation power of NNs. Therefore, modern DNNs often choose the ReLU activation function based on purely computational considerations. However, mathematical properties of the ReLU function, such as homogeneity and piecewise-linearity, open unique possibilities for mathematical analysis of ReLU NNs. Consequently, there is a growing body of theoretical research dedicated specifically to ReLU NNs. In this thesis, the primary focus is also on DNNs with ReLU activation.

2.1.1 Survey of NN Architectures

State-of-the-art DNNs in many applications use more advanced architectures to achieve better performance in specific tasks. These architectures typically leverage properties of the input data, such as spatial translation-invariance of objects in images or the sequential nature of words in text. In this section, we briefly describe several common NN architectures.

Convolutional NNs (CNNs) CNNs are foundational models of modern computer vision. These architectures have been state-of-the-art models for image recognition since the last decade (Krizhevsky et al., 2012; Szegedy et al., 2015; He et al., 2015). The defining feature of CNNs is the so-called *convolutional layers*. These layers often assume that the input has two spatial dimensions (as images) and may have several channels (such as three color channels of RGB images). In contrast to neurons of a fully-connected layer, each neuron of a convolutional layer is connected only to a fraction of the input neurons, called the receptive field. The receptive fields of the output neurons are obtained by sliding a two-dimensional window over the spacial dimensions of the input. An illustration of a convolutional layer is given in Figure 2.2a. Usually, all the neurons in a given output channel share the same



(a) Convolutional layer. (b) Recurrent NN. (c) Residual NN

weights matrix. This property is motivated by translation-invariance of natural images: a shift of an image does not change the objects that can be recognized in the image. Moreover, this property greatly reduces the number of parameters associated with a convolutional layer, in comparison with a fully-connected layer. Stacking multiple convolutional layers together allows to increase the receptive field of the output neurons and, potentially, capture more complex features of the input.

Recurrent NNs (RNNs) RNNs are often used for applications with sequential input data, such as text translation or speech recognition (Graves et al., 2013; Sutskever et al., 2011). In contrast to feedforward NNs, RNNs allow output of some neurons to affect subsequent input to the same neurons. In other words, RNNs have cycles in their computational graphs. When sequential input is gradually passed into an RNN, the internal state of the network (represented by the hidden layers) changes at each step and impacts the processing of the next data chunk. Figuratively, this means that RNNs have memory about the previous input chunks. A simple example of an RNN with a single fully-connected recurrent layer is given in Figure 2.2b. State-of-the-art RNNs can include multiple recurrent subnetworks with various architectures, and often use more advanced structure for recurrent layers, such as Long Short-Term Memory (LSTM) units (Hochreiter and Schmidhuber, 1997).

Graph NNs (GNNs) GNNs are designed for input data represented as graphs, where nodes and/or edges of the graph are described by feature vectors. GNNs achieve state-of-the-art performance in applications such as proteins' interactions (Fout et al., 2017) or social networks analysis (Wu et al., 2020). The defining feature of GNNs are the so-called *message passing layers*, which update feature vectors for each node of the input graph. Each message passing layer aggregates the information from the neighbourhood of a given node to compute a new feature vector of the node. This idea can be seen as a generalization of

convolutional layers, which compute a feature vector of a given pixel by aggregating the information from all the neighboring pixels, to arbitrary graph structures.

Residual NNs (ResNets) ResNets are networks with so-called *residual blocks*. The output of a residual block is a sum of a function implemented by the layers within this block and the input of the block. In other words, there is a connection between the input and the output layers of a residual block. An example of a ResNet with a single residual block is given in Figure 2.2c. While "ResNet" usually refers to the architecture introduces in the original paper on residual networks (He et al., 2016), many modern DNN architectures, such as transformers and LSTM networks, include residual blocks.

Transformers The Transformer architecture was introduced in Vaswani et al. (2017), and since then revolutionized natural language processing (NLP) and beyond. The key innovation of the Transformers is the self-attention mechanism, which allows the model to weigh different parts of the input sequence differently when making predictions. This attention mechanism eliminates the need for recurrent or convolutional layers, making it highly parallelizable and efficient for processing long-range dependencies. Transformers have become the backbone of various state-of-the-art models for tasks like machine translation, language understanding, and image generation.

Although we categorized architectures into distinct groups for clarity in the preceding discussion, modern DNNs commonly incorporate diverse combinations of these approaches. Examples include convolutional RNNs or residual CNNs, convolutional Transformers, and many more. This flexibility allows modern DNNs to leverage the strengths of different architectures to achieve state-of-the-art performance in a wide range of tasks.

2.1.2 Approximation Power of NNs

One of the most well-known results of NNs theory is the universal approximation theorem, which shows that even shallow fully-connected NNs can approximate any continuous function on a compact set with arbitrary precision (Cybenko, 1989; Hornik, 1991; Hornik et al., 1989; Funahashi, 1989). While there is a series of such theorems in the literature, we adopt a version from Pinkus (1999):

Theorem 2.1 (Universal Approximation Theorem)**.** *Let $f : \mathbb{R}^{n_0} \to \mathbb{R}^{n_2}$ denote a fully-connected NN with depth $L = 2$, widths $\{n_\ell\}_{\ell=0}^{2} \in \mathbb{N}^3$, and activation functions $\phi_2(x) = x$, $\phi_1(x) = \sigma(x), x \in \mathbb{R}$, for continuous $\sigma : \mathbb{R} \to \mathbb{R}$, i.e.,*

$$f_\theta(\mathbf{x}) = \mathbf{W}^2\sigma(\mathbf{W}^1\mathbf{x} + \mathbf{b}^1) + \mathbf{b}^2 \tag{2.7}$$

with parameters $\theta = \Big(\big(\mathbf{W}^1, \mathbf{b}^1\big), \big(\mathbf{W}^2, \mathbf{b}^2\big)\Big)$. Then σ is not polynomial if and only if for every input dimension $n_0 \in \mathbb{N}$, output dimension $n_2 \in \mathbb{N}$, compact set $K \subseteq \mathbb{R}^{n_0}$, continuous function $g : K \to \mathbb{R}^{n_2}$, and $\epsilon > 0$, there exists hidden layer width $n_1 \in \mathbb{N}$ and a choice of

parameters $\theta = \theta^*$ *such that*

$$\sup_{\mathbf{x} \in K} \|f_{\theta^*}(\mathbf{x}) - g(\mathbf{x})\| < \epsilon. \tag{2.8}$$

There are also numerous results that quantify the approximation rates of NNs with one hidden layer for particular classes of functions, such as smooth functions or functions with certain symmetries. We refer to DeVore et al. (2021) for a review of such results. Moreover, there is a more recent line of research that aims to explain the effects of depth for the approximation power of NNs. Some of these results show that depth allows NNs with very few neurons in the hidden layers to still achieve universal approximation (Hanin and Sellke, 2018; Hanin, 2019; Kidger and Lyons, 2020). Other results show that deep NNs are more efficient at approximation of certain classes of functions, in a sense that they require fewer parameters than shallow NNs for the same approximation rates (Eldan and Shamir, 2016; Yarotsky, 2017). One can find a survey of these results in Berner et al. (2021).

Overall, the literature shows that even simple NN architectures are sufficiently powerful to efficiently represent continuous functions. Assuming that target functions in most applications are at least piecewise continuous, this means that there usually exists a fully-connected NN with an appropriate choice of parameters that is guaranteed to achieve high performance on a given task. However, approximation theory does not explain how to find parameters for such a NN. It also does not explain why more advanced architectures, such as those surveyed in the previous section, may improve NNs' performance in many applications. While approximation theory is the most well-established subfield of the mathematical foundations of NNs, and provides rigorous results for the best-case performance of NNs, there is abundant evidence in the literature that real-world NNs do not achieve the performance predicted by approximation theory (Adcock and Dexter, 2021; Fokina and Oseledets, 2020; Hanin and Rolnick, 2019). We discuss the causes for this discrepancy in the next two sections.

2.2 Training

Training, also called *learning*, refers to the process of algorithmically identifying optimal parameters of a NN with a given architecture for a given task. In this thesis, we will focus on the *supervised learning setting*, where a NN is given a finite dataset of input samples with correct output values. Then the goal of training is to reconstruct a function that generated the dataset. In statistical learning theory, the standard approach to this problem is *Empirical Risk Minimization* (ERM).

Definition 2.5 (Empirical Risk Minimization)**.** *Given a set of functions* $\mathcal{F}$ *with domain* $\mathcal{X}$ *and codomain* $\mathcal{Y}$*, a training dataset* $S = \{(x_i, y_i)\}_{i=1}^N$ *with* $x_i \in \mathcal{X}$*,* $y_i \in \mathcal{Y}$*,* $i \in [1, N]$*, and a loss function* $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$*, an empirical risk minimization algorithm chooses* $\widehat{f}_S \in \mathcal{F}$

such that

$$\widehat{f}_S \in \arg\min_{f\in\mathcal{F}} \widehat{\mathcal{L}}_S(f), \quad \widehat{\mathcal{L}}_S(f) := \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(f(x_i), y_i), \tag{2.9}$$

assuming such a choice exists, i.e., a minimum of $\widehat{\mathcal{L}}_S$ is attained on $\mathcal{F}$. Here the function $\widehat{\mathcal{L}}_S : \mathcal{F} \to \mathbb{R}$ is called the empirical risk.

In other words, the goal of ERM algorithm is to choose a function $\widehat{f}_S \in \mathcal{F}$ that offers the best approximation of the training set S according to a given loss function $\mathcal{L}$. Note that the choice of loss function may significantly impact the function selected by ERM. For example, ERM with loss function $\mathcal{L}(\hat{y}, y) = |\hat{y} - y|$ prefers functions such that $\widehat{f}_S(x_i) \approx y_i$ for all $i \in [1, N]$. On the other hand, ERM with loss function $\mathcal{L}(\hat{y}, y) = \text{sign}(\hat{y}y)$ only optimizes for $\text{sign}(f(x_i)) = \text{sign}(y_i)$, $i \in [1, N]$.

In the context of NNs, the set of functions in the ERM definition comprises all the functions that can be realized by a chosen NN architecture, given by

$$\mathcal{F} = \{f_\theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L} \mid \theta \in \mathbb{R}^P\}, \tag{2.10}$$

where P is the number of the NN's parameters. Common choices of loss function include Mean Squared Error (MSE) and Cross-Entropy (CE) losses.

Definition 2.6 (Mean Squared Error Loss)**.** *Mean squared error loss $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is given by*

$$\mathcal{L}(\hat{y}, y) = \|\hat{y} - y\|_2^2. \tag{2.11}$$

Definition 2.7 (Cross-Entropy Loss)**.** *Let the output space $\mathcal{Y}$ contain $(\mathbb{R}^n_+ \setminus \{0\})$-vectors that sum to 1, i.e.,*

$$\mathcal{Y} = \{y \in \mathbb{R}^n_+ \setminus \{0\} \mid \|y\|_1 = 1\}. \tag{2.12}$$

Then cross-entropy loss $\mathcal{L} : \mathcal{Y} \times \overline{\mathcal{Y}} \to \mathbb{R}$ is given by

$$\mathcal{L}(\hat{y}, y) = -\langle y, \log \hat{y}\rangle, \tag{2.13}$$

where logarithm is applied element-wise.

While MSE is a general-purpose loss function, CE loss is a common choice for classification problems, where the NN's output is interpreted as a vector of probabilities of a finite number of classes. However, there is a growing body of evidence that MSE loss performs at least on par with CE loss for classification (Hui and Belkin, 2021; Demirkaya et al., 2020; Poggio and Liao, 2021). From the optimization standpoint, MSE and CE losses have very different properties. Indeed, MSE attains its minimum when the DNN's output matches the target output on the whole dataset, i.e., $f_\theta(x_i) = y_i$ for all $i = [1, N]$. On the other hand, DNNs trained with CE loss usually normalize the output using the so-called *softmax* function, defined as follows

$$\hat{y} = \text{softmax}(\mathbf{x}^L) := \frac{\exp(\mathbf{x}^L)}{\|\exp(\mathbf{x}^L)\|_1}, \tag{2.14}$$

which makes the global minima of CE loss unattainable for any finite parameters. Since MSE loss results in simpler dynamics equations and has more intuitive convergence properties, we mostly focus on dynamics with MSE loss in this thesis.

2.2.1 Gradient Descent

While ERM sets the goal of training, it does not specify the algorithmic procedure to find the minimum of the empirical risk. Some optimization problems arising from ERM have analytical solutions. A prominent example is the ordinary least squares problem, which corresponds to training a fully-connected NN with $L = 1$ (i.e., with no hidden layers) and linear activation function using MSE loss. However, analytical solutions are very rare in non-linear optimization problems, and virtually do not exist for real-world NNs. Therefore, NNs are trained using numerical optimization methods, the most common of which is *Gradient Descent* (GD), described in Algorithm 1.

Algorithm 1: Gradient Descent

Input : Differentiable empirical risk function $\widehat{\mathcal{L}}_S : \mathbb{R}^P \to \mathbb{R}$,
a number of steps $K \in \mathbb{N}$, a sequence of step sizes $\{\eta_k\}_{k=1}^K$,
initial choice of parameters $\theta^{(0)} \in \mathbb{R}^P$.

Output : A sequence of parameters values $\{\theta^{(k)}\}_{k=0}^K$.

for $k = 1, \dots, K$ **do**

  $\theta^{(k)} \leftarrow \theta^{(k-1)} - \eta_k \nabla_\theta \widehat{\mathcal{L}}_S(\theta^{(k-1)})$

end

One can see that GD is a greedy algorithm, which makes a step towards the steepest descent of the empirical loss function in each iteration. Note that in Algorithm 1, we define the empirical loss function $\widehat{\mathcal{L}}_S : \mathbb{R}^P \to \mathbb{R}$ as a function of NN's parameters $\theta \in \mathbb{R}^P$, since the loss only depends on the NN's output function f_θ through the parameters.

2.2.2 Backpropagation

GD and its variants became the algorithms of choice for optimizing NNs' parameters mainly due to their computation efficiency. Modern DNNs typically have from hundreds of thousands to even billions of parameters, and are trained for hundreds of thousands of GD iterations. Therefore, it is essential to have a very efficient and universal procedure to compute gradients of the empirical loss function with respect to the parameters in order to make the optimization feasible. Such a procedure exists for NNs and is called *backpropagation*. We will describe the backpropagation algorithm for fully-connected NNs. However, the same algorithm can be easily generalized to any feedforward NN architecture, and can also be adapted for RNNs (Werbos, 1990).

The backpropagation algorithm is essentially an efficient application of the chain rule, which makes use of the observation that parameters of each layer of a fully-connected NN affect

the output function only through the subsequent layers. To outline the backpropagation algorithm, we first introduce two key concepts: the *forward pass* and the *backward pass*. The forward pass is the iterative computation of the NN's output for a given input layer-by-layer, progressing in the forward direction – from the input layer to the output layer. The backward pass is the layer-by-layer computation of the NN's "error" in the reverse direction – from the output layer to the input layer.

Definition 2.8 (Forward Pass, Activation, Pre-Activation)**.** *Consider a fully-connected NN with depth $L \in \mathbb{N}$, widths $\{n_\ell\}_{\ell=0}^{L}$, and activation functions $\{\phi_\ell\}_{\ell=1}^{L}$. The forward pass of such a network on input $x := \mathbf{x}^0 \in \mathbb{R}^{n_0}$ is defined as follows:*

$$\mathbf{x}^\ell(x) := \phi_\ell(\mathbf{h}^\ell(x)), \quad \mathbf{h}^\ell(x) := \mathbf{W}^\ell \mathbf{x}^{\ell-1}(x) + \mathbf{b}^\ell, \quad \ell = [1, L], \tag{2.15}$$

where $\mathbf{x}^\ell : \mathbb{R}^{n_0} \to \mathbb{R}^{n_\ell}$ is called the activation, and $\mathbf{h}^\ell : \mathbb{R}^{n_0} \to \mathbb{R}^{n_\ell}$ is called the pre-activation of layer $\ell = [1, L]$.

Note that in the above definition activation $\mathbf{x}^\ell$ and pre-activation $\mathbf{h}^\ell$ are functions of the NN's input and depend on the NN's parameters $\theta \in \mathbb{R}^P$. For ease of notation, we omit the dependence on the input and the parameters when it does not lead to confusion in the following discussion. One can see that $\mathbf{x}^L = f_\theta$, according to Definition 2.2 of fully-connected NNs. In the following, we will use $\mathbf{x}^L$ and f_θ interchangeably to denote the output function of a NN.

Definition 2.9 (Backward Pass, Backpropagated Error)**.** *Consider a fully-connected NN with depth $L \in \mathbb{N}$, widths $\{n_\ell\}_{\ell=0}^{L}$, and activation functions $\{\phi_\ell\}_{\ell=1}^{L}$. Let $\mathcal{L} : \mathbb{R}^{n_L} \times \mathbb{R}^{n_L} \to \mathbb{R}$ be the loss function associated with training the NN. The backward pass of such a network on input $x := \mathbf{x}^0 \in \mathbb{R}^{n_0}$ with target output $y \in \mathbb{R}^{n_L}$ is defined as follows:*

$$\boldsymbol{\delta}^L(x, y) := \frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^L} = \frac{\partial \mathcal{L}(\mathbf{x}^L(x), y)}{\partial \mathbf{x}^L} \odot \phi_L'(\mathbf{h}^L(x)), \tag{2.16}$$

$$\boldsymbol{\delta}^\ell(x, y) := \frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^\ell} = \left(\mathbf{W}^{\ell+1}\right)^\top \boldsymbol{\delta}^{\ell+1}(x, y) \odot \phi_\ell'(\mathbf{h}^\ell(x)), \quad \ell = [1, L-1], \tag{2.17}$$

where $\boldsymbol{\delta}^\ell : \mathbb{R}^{n_0} \times \mathbb{R}^{n_L} \to \mathbb{R}^{n_\ell}$ is called the backpropagated error in layer $\ell = [1, L]$

Note that the gradient of the loss $\mathcal{L}(f_\theta(x), y)$ w.r.t. $\mathbf{h}^\ell$ in the above definition is more formally defined as follows:

$$\frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^\ell} := \nabla g(\mathbf{h}^\ell(x)), \tag{2.18}$$

where $g : \mathbb{R}^{n_\ell} \to \mathbb{R}$ is a function such that $\mathcal{L}(f_\theta(x), y) = (g \circ \mathbf{h}^\ell)(x)$. However, we will use the same abuse of notation throughout the thesis. Similarly to the forward pass variables, $\boldsymbol{\delta}^\ell$ is a function of the training sample (x, y) and depends on the NN's parameters θ. However, we will omit the dependence of the backpropagated error on its arguments and the parameters for clarity of notation, as long as this does not lead to confusion.

Algorithm 2: Backpropagation

Input : Fully-connected NN f_θ with depth L, widths $\{n_\ell\}_{\ell=0}^{L}$, and differentiable activation functions $\{\phi_\ell\}_{\ell=1}^{L}$, parameters $\theta = \left\{(\mathbf{W}^\ell, \mathbf{b}^\ell)\right\}_{\ell=1}^{L}$, differentiable loss function $\mathcal{L}$, input dataset $S = \left\{(x_i, y_i)\right\}_{i=1}^{N}$.

Output : Gradient of the empirical loss w.r.t. the parameters $g = \nabla_\theta \widehat{\mathcal{L}}_S(\theta)$.

$g \leftarrow$ gradient container filled with zeros;
for $i = 1, \ldots, N$ **do**
  $\mathbf{x}^0 \leftarrow x_i$;
  $y \leftarrow y_i$;
  for $\ell = 1, \ldots, L$ **do**
    $\mathbf{h}^\ell \leftarrow \mathbf{W}^\ell \mathbf{x}^{\ell-1} + \mathbf{b}^\ell$;
    $\mathbf{x}^\ell \leftarrow \phi_\ell(\mathbf{h}^\ell)$;
  end
  $\boldsymbol{\delta}^L \leftarrow \nabla_{\mathbf{x}^L} \mathcal{L}(\mathbf{x}^L, y) \odot \phi_L'(\mathbf{h}^L)$;
  for $\ell = L-1, \ldots, 1$ **do**
    $\boldsymbol{\delta}^\ell \leftarrow \left(\mathbf{W}^{\ell+1}\right)^\top \boldsymbol{\delta}^{\ell+1} \odot \phi_\ell'(\mathbf{h}^\ell)$;
  end
  $g_{\text{new}} \leftarrow$ empty gradient container;
  for $\ell = 1, \ldots, L$ **do**
    $d\mathbf{W}^\ell \leftarrow \boldsymbol{\delta}^\ell \otimes \mathbf{x}^{\ell-1}$;
    $d\mathbf{b}^\ell \leftarrow \boldsymbol{\delta}^\ell$;
    Insert $(d\mathbf{W}^\ell, d\mathbf{b}^\ell)$ into g_{new};
  end
  $g \leftarrow g + \frac{g_{\text{new}}}{N}$;
end

The backpropagated error $\boldsymbol{\delta}^\ell$ in layer ℓ can be interpreted as the role of the pre-activation $\mathbf{h}^\ell$ in the final loss value $\mathcal{L}(f_\theta(x), y)$. The formula for the backpropagated error computation in equation (2.17) comes from the application of the chain rule as follows:

$$\frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^\ell}^\top = \frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^{\ell+1}}^\top \cdot \frac{\partial \mathbf{h}^{\ell+1}}{\partial \mathbf{x}^\ell} \cdot \frac{\partial \mathbf{x}^\ell}{\partial \mathbf{h}^\ell} = (\boldsymbol{\delta}^{\ell+1})^\top \mathbf{W}^{\ell+1} \operatorname{diag}(\phi_\ell'(\mathbf{h}^\ell)), \tag{2.19}$$

where $\operatorname{diag}(v) \in \mathbb{R}^n$ denotes a diagonal matrix with values of vector $v \in \mathbb{R}^n$ on the main diagonal. The above equation is easy to derive taking into account that $\mathbf{x}^\ell = \phi_\ell(\mathbf{h}^\ell)$ and $\mathbf{h}^{\ell+1} = \mathbf{W}^{\ell+1}\mathbf{x}^\ell + \mathbf{b}^{\ell+1}$ according to Definition 2.8 of the forward pass.

Given the forward pass and the backward pass variables, the gradients of a NN can be

computed as follows:

$$\frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{W}^\ell} = \frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^\ell} \cdot \frac{\partial \mathbf{h}^\ell}{\partial \mathbf{W}^\ell} = \boldsymbol{\delta}^\ell \otimes \mathbf{x}^{\ell-1}, \tag{2.20}$$

$$\frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{b}^\ell} = \frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{h}^\ell} \cdot \frac{\partial \mathbf{h}^\ell}{\partial \mathbf{b}^\ell} = \boldsymbol{\delta}^\ell, \quad \ell = [1, L], \tag{2.21}$$

where $\partial \mathcal{L}(f_\theta(x), y)/\partial \mathbf{W}^\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ is the matrix of the gradients w.r.t. the weights of layer ℓ, and $\partial \mathcal{L}(f_\theta(x), y)/\partial \mathbf{b}^\ell \in \mathbb{R}^{n_\ell}$ is the vector of gradients w.r.t. the biases of layer ℓ. Finally, we notice that the empirical loss function $\widehat{\mathcal{L}}_S(\theta)$ is the mean of the losses corresponding to individual samples in the dataset. Therefore, the gradient of the empirical risk function is simply the sum of gradients corresponding to the individual input samples. We summarize the backpropagation approach to gradient computation in Algorithm 2.

The efficiency of the backpropagation algorithm in comparison with the naive application of the chain rule comes from two observations: using $\boldsymbol{\delta}^{\ell+1}$ to compute $\boldsymbol{\delta}^\ell$ reuses all the repeated computations; computing the derivatives in the backward direction from output to input only requires matrix-vector products. In fact, backpropagation is a special case of the reverse-mode automatic differentiation, which is usually more efficient when the number of variables is much larger than the output dimension (Griewank and Walther, 2008). While Algorithm 2 is specific for fully-connected NNs, modern frameworks for NN training, such as PyTorch (Paszke et al., 2019) or JAX (Bradbury et al., 2018), can perform reverse-mode automatic differentiation for a wide variety of NN architectures. These frameworks decompose a given function (implemented by a computer program) into a sequence of primitive operations with specified rules for computation of derivatives. This way, it is possible to perform chain rule efficiently in a completely mechanical way.

2.2.3 Gradient Flow

Many papers that consider training dynamics of DNNs rely on *Gradient Flow* (GF), which is a continuous-time approximation of GD. Recall the equation for $k-$th GD step with learning rate η from Algorithm 1:

$$\theta^{(k)} \leftarrow \theta^{(k-1)} - \eta \nabla_\theta \widehat{\mathcal{L}}_S(\theta^{(k-1)}). \tag{2.22}$$

We can now introduce a smooth function $\theta(t), t \geq 0$, such that $\theta^{(k)} := \theta(k\eta)$ for any $k \in \mathbb{N}$. Then, by taking the limit $\eta \to 0$, we obtain the corresponding GF equation:

$$\dot{\theta} = -\nabla_\theta \widehat{\mathcal{L}}_S(\theta). \tag{2.23}$$

In other words, GF can be seen as the limit of GD, where the learning rate tends to zero. This approximation makes the analysis of DNNs' dynamics considerably simpler, since it allows to remove the higher order terms with respect to η. While GF cannot capture all the properties of GD dynamics, there is theoretical and empirical evidence that it approximates the performance of GD with small enough learning rate (Elkabetz and Cohen, 2021). In Section 3.3 of this thesis, our contributions focus on GF dynamics of DNNs, leaving the consideration of the discrete-time effects for the future work.

2.2.4 Effects of Initialization

The initial parameters $\theta^{(0)}$ in GD algorithm are typically chosen randomly according to a given distribution. Common initialization schemes for fully-connected DNNs satisfy the following conditions for all $\ell \in [1, L], i \in [1, n_\ell], j \in [1, n_{\ell-1}]$:

$$\sqrt{n_{\ell-1}} \cdot \mathbf{W}_{i,j}^{\ell} \sim \mu_w \text{ i.i.d.}, \quad \mathbf{b}_i^{\ell} \sim \mu_b \text{ i.i.d.}, \tag{2.24}$$

where μ_w and μ_b are given probability measures, such that

$$\int x d\mu_{w,b}(x) = 0, \quad \sigma_{w,b}^2 := \int x^2 d\mu_{w,b}(x) < \infty. \tag{2.25}$$

Note that the variance of the weights at initialization is usually scaled by the width of the network to avoid overflow for very wide DNNs.

Under such a random initialization, all the variables in the forward pass (Definition 2.8) and the backward pass (Definition 2.9) of DNNs also become random. Therefore, it is of interest to study the statistical properties of DNNs with various initialization settings. One relevant line of research focuses on *signal propagation* in DNNs, i.e., changes of the distribution as it propagates through consecutive layers of DNNs (Poole et al., 2016; Schoenholz et al., 2016; Karakida et al., 2019). These works mostly focused on the special case of Gaussian initialization with hyperparameters $\sigma_w \in \mathbb{R}_+$ and $\sigma_b \in \mathbb{R}_+$, given by

$$\mathbf{W}_{i,j}^{\ell} \sim \mathcal{N}\Big(0, \frac{\sigma_w^2}{n_{\ell-1}}\Big) \text{ i.i.d. }, \quad \mathbf{b}_i^{\ell} \sim \mathcal{N}(0, \sigma_b^2) \text{ i.i.d. }, \tag{2.26}$$

and relied on the so called *mean field approximation*. Another closely related line of research establishes the connection between the infinite-width limit of DNNs and Gaussian processes (Lee et al., 2018; Yang, 2020a; Matthews et al., 2018). Since a significant part of this thesis is devoted to analyzing the statistical properties of DNNs at initialization, this section will delve into the basic aspects of these research areas.

Mean field approximation Consider the forward pass of a fully-connected DNN (see Definition 2.8). Given the general form of initialization in (2.24), the following is immediate for the pre-activation vectors $\mathbf{h}_a^{\ell} := \mathbf{h}^{\ell}(a), \mathbf{h}_b^{\ell} := \mathbf{h}^{\ell}(b)$ in layers $\ell \in [1, L]$ computed for inputs $a, b \in \mathbb{R}^{n_0}$:

$$\mathbb{E}[\mathbf{h}_a^{\ell}] = \mathbb{E}[\mathbf{h}_b^{\ell}] = 0, \quad \frac{1}{n_\ell}\mathbb{E}\big[\langle \mathbf{h}_a^{\ell}, \mathbf{h}_b^{\ell} \rangle\big] = \frac{\sigma_w^2}{n_{\ell-1}}\mathbb{E}\big[\langle \mathbf{x}_a^{\ell-1}, \mathbf{x}_b^{\ell-1} \rangle\big] + \sigma_b^2, \tag{2.27}$$

We can also notice that entries of $\mathbf{h}^{\ell}$, i.e., different neurons in the same layer, are identically distributed. Therefore, we can write the following:

$$\mathbb{E}\big[\langle \mathbf{h}_a^{\ell}, \mathbf{h}_b^{\ell} \rangle\big] = n_\ell \mathbb{E}_{u,v}[uv] = n_\ell Cov(u, v), \tag{2.28}$$

where (u, v) are random variables, jointly distributed as $(\mathbf{h}_{a,i}^{\ell}, \mathbf{h}_{b,i}^{\ell})$ for any neuron $i \in [n_\ell]$. Another general observation is that different neurons in the same layer are uncorrelated, i.e., $Cov(\mathbf{h}_{a,i}^{\ell}, \mathbf{h}_{b,j}^{\ell}) = 0$ if $i \neq j$. However, this does not imply independence, since different neurons still depend on the same parameters of previous layers.

Similarly, we can write the following for the scalar product of activations:

$$\mathbb{E}\big[\langle \mathbf{x}_a^{\ell}, \mathbf{x}_b^{\ell} \rangle\big] = \mathbb{E}\big[\langle \phi_\ell(\mathbf{h}_a^{\ell}), \phi_\ell(\mathbf{h}_b^{\ell}) \rangle\big] = n_\ell \mathbb{E}_{u,v}\big[\phi_\ell(u)\phi_\ell(v)\big], \tag{2.29}$$

Given the distribution of (u, v), the above equations can provide recursive relationships for the statistics of the forward pass of DNNs. However, even for Gaussian initialization (2.26), deriving the exact distribution of the pre-activations is challenging. Indeed, even though the distribution of $\mathbf{h}_{a,i}^{\ell}$ is Gaussian when conditioned on the parameters of all the previous layers, the marginal distribution does not have to be Gaussian. Therefore, most of the works on signal propagation in DNNs make the following assumption:

Assumption 2.1 (Mean Field Approximation (MFA))**.** *Assume that pre-activations of all the neurons of any layer $\ell \in [1, L]$ are mutually independent and normally distributed.*

Under this assumption, we can recursively calculate the expectations in (2.28) and (2.29). Denoting $q^{\ell}(a, b) := \mathbb{E}\big[\langle \mathbf{x}_a^{\ell}, \mathbf{x}_b^{\ell} \rangle\big]/n_\ell$, we have:

$$q^{\ell}(a,b) = \mathop{\mathbb{E}}_{(u,v)\sim\mathcal{N}(0,\Sigma^{\ell})}\big[\phi_\ell(u)\phi_\ell(v)\big], \quad \Sigma^{\ell}(a,b) := \sigma_w^2 \begin{bmatrix} q^{\ell-1}(a,a) & q^{\ell-1}(a,b) \\ q^{\ell-1}(a,b) & q^{\ell-1}(b,b) \end{bmatrix} + \sigma_b^2. \tag{2.30}$$

Therefore, the pre-activations in layer ℓ behave as a centered Gaussian process with covariance Σ^{ℓ}. In fact, several works have rigorously proved that this is indeed the case in the infinite-width limit of DNNs with Gaussian initialization (2.26) under weak assumptions (Lee et al., 2018; Yang, 2020a). In other words, MFA assumption leads to correct computations in the infinite-width limit. For some activation functions, the expectation in the above equation has closed-form expressions. Such expressions for ReLU and a certain sigmoid function are provided in the Appendix of Section 3.1.

To derive similar recursive expressions for the backward pass of a fully-connected DNN (Definition 2.9), previous works relied on one more assumption (Schoenholz et al., 2016):

Assumption 2.2 (Gradient Independence Assumption (GIA))**.** *Assume that matrix $(\mathbf{W}^{\ell})^T$ in the backward pass equations and matrix $\mathbf{W}^{\ell}$ in the forward pass equations are independent for all $\ell \in [1, L]$.*

Then the following recursive expression holds for the backpropagated errors $\boldsymbol{\delta}_a^{\ell} := \boldsymbol{\delta}^{\ell}(a)$, $\boldsymbol{\delta}_b^{\ell} := \boldsymbol{\delta}^{\ell}(b)$ in layer $\ell \in [1, L-1]$:

$$\frac{1}{n_\ell}\mathbb{E}\big[\langle \boldsymbol{\delta}_a^{\ell}, \boldsymbol{\delta}_b^{\ell} \rangle\big] =: p^{\ell}(a,b) = p^{\ell+1}(a,b)\frac{n_{\ell+1}}{n_\ell}\sigma_w^2 \mathop{\mathbb{E}}_{(u,v)\sim\mathcal{N}(0,\Sigma^{\ell})}[\phi_\ell'(u)\phi_\ell'(v)]. \tag{2.31}$$

Phases of initialization Given the mean field approximations for the forward and backward passes, it is possible to derive norms of the DNN's gradients at initialization, using the following expressions:

$$\left\|\frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{W}^\ell}\right\|_2^2 = \|\boldsymbol{\delta}^\ell\|_2^2 \|\mathbf{x}^{\ell-1}\|_2^2, \quad \left\|\frac{\partial \mathcal{L}(f_\theta(x), y)}{\partial \mathbf{b}^\ell}\right\|_2^2 = \|\boldsymbol{\delta}^\ell\|_2^2, \tag{2.32}$$

which directly follow from equations for gradients computation in the backpropagation algorithm (2.20). According to Schoenholz et al. (2016) and Poole et al. (2016), the following quantity

$$\chi := \sigma_w^2 \lim_{\ell \to \infty} \mathbb{E}_u\left[\left(\phi_\ell'(u)\right)^2\right], \quad u \sim \mathcal{N}(0, \sigma_w^2 q^{\ell-1}(a,a) + \sigma_b^2) \tag{2.33}$$

can help to identify three distinct *phases* in the space of hyperparameters (σ_w, σ_b):

- **Ordered phase:** If $\chi < 1$, the norms of the DNN's gradients asymptotically decrease at an exponential rate as the depth increases. At the same time, correlations $c^\ell(a,b) = \frac{q^\ell(a,b)}{\sqrt{q^\ell(a,a) q^\ell(b,b)}}$ between different input samples grow as the network gets deeper.
- **Chaotic phase:** If $\chi > 1$, the gradients asymptotically grow at an exponential rate with the depth, while the correlations $c^\ell(a,b)$ decrease.
- **Edge of chaos:** If $\chi \approx 1$, there is no exponential asymptotics of the gradients' norm with respect to the depth, which leads to better numerical stability and allows training deeper networks. For ReLU networks, this setting corresponds to He initialization, introduced in He et al. (2015).

Problems of mean field approximation While the contributions of this thesis in Section 3.2 study the differences of DNNs' behaviour in the initialization phases defined above, our theory does not rely on the mean field approximation. Namely, we do not use Assumption 2.1 and Assumption 2.2. While these assumptions often lead to correct computations in the infinite-width limit (see e.g. the discussion in Yang (2020b)), we find that they lead to dramatically incorrect results in the infinite-depth-and-width limit. The reason for this disparity is that the effects introduced by the dependence between forward and backward chains in layer ℓ are of the order $O(1/n_\ell)$. Therefore, these effects typically vanish in the infinite-width limit. Similarly, the effects introduced by the dependences between different neurons of the same layer vanish in the infinite-width limit. However, when the depth is comparable with width, multiplicative terms of order $O(1/n_\ell)$ in each layer result in non-trivial changes of the final expressions for the DNN's gradients.

2.3 Generalization

Generalization is a field of machine learning theory, which studies how well a given model performs on unseen data, i.e., data not used in the training process. The central quantity of interest in generalization theory is the *expected risk*.

Definition 2.10. *(Expected Risk) Let $\mathcal{F}$ be a set of functions with domain $\mathcal{X}$ and codomain $\mathcal{Y}$. Let μ be a probability distribution on $\mathcal{X} \times \mathcal{Y}$, and $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ be a loss function, such that $(x, y) \to \mathcal{L}(f(x), y)$ is measurable for any $f \in \mathcal{F}$. Then the generalization error, also called the expected risk, of function $f \in \mathcal{F}$ is defined as follows:*

$$\mathcal{L}_\mu(f) := \mathbb{E}_{(x,y)\sim\mu}[\mathcal{L}(f(x), y)]. \tag{2.34}$$

Recall that we introduced the empirical risk function $\widehat{\mathcal{L}}_S(f)$ in Definition 2.5 of ERM. Then, assuming that the dataset S is sampled i.i.d. from a distribution μ, i.e., $S \sim \mu^N$, we have the following relationship between expected and empirical risks:

$$\mathcal{L}_\mu(f) = \mathbb{E}_{S\sim\mu^N}\left[\widehat{\mathcal{L}}_S(f)\right]. \tag{2.35}$$

The goal of generalization theory is to derive bounds on the expected risk of a function $\widehat{f}_S \in \mathcal{F}$ chosen by a learning algorithm given a dataset S. Therefore, generalization theory results essentially study the concentration of the empirical risk as the dataset size N grows, with an additional difficulty that the function of interest depends on the dataset.

2.3.1 Classical Generalization Bounds

Classical results of generalization theory assume that, depending on the dataset, the learning algorithm may choose any function from the set $\mathcal{F}$, so the bounds must hold simultaneously for all $f \in \mathcal{F}$. Therefore, such results usually depend on a certain measure of *complexity* of $\mathcal{F}$. The simplest results of this kind concern finite sets of functions, where the natural measure of complexity is the size of $\mathcal{F}$. For instance, the following theorem provides a generalization bound for a binary classification problem, where the target outputs can only take two distinct values:

Theorem 2.2 (Adopted from Vapnik (2013))**.** *Let $\mathcal{F}$ be a set of functions from $\mathcal{X}$ to $\mathcal{Y}$, where $|\mathcal{Y}| = 2$. Let the associated loss function be given by $\mathcal{L}(\hat{y}, y) = \mathbb{1}_{y\neq\hat{y}}$. Let μ be an arbitrary distribution over $\mathcal{X} \times \mathcal{Y}$, and $S \sim \mu^N$ be a dataset comprising $N \in \mathbb{N}$ i.i.d. samples drawn from μ. Assume that $\mathcal{F}$ is finite, i.e., $|\mathcal{F}| < \infty$. Then for any $\delta \in (0, 1]$ the following holds with probability at least $1 - \delta$:*

$$\sup_{f\in\mathcal{F}} |\mathcal{L}_\mu(f) - \widehat{\mathcal{L}}_S(f)| \leq \sqrt{\frac{\log|\mathcal{F}| + \log(1/\delta)}{2N}}. \tag{2.36}$$

This result relies on Hoeffding's inequality, applied to the bounded random variables $\mathcal{L}(f(x_i), y_i)$ for $(x_i, y_i) \in S$, and a union bound over all the functions $f \in \mathcal{F}$. However, this approach cannot be extended to infinite sets of functions, which are common in practical machine learning problems. Therefore, much of classical generalization theory focuses on developing complexity measures for various learning settings. One prominent example of such measures for binary classification problems is the *Vapnik–Chervonenkis (VC) dimension*.

Definition 2.11 (Growth Function, VC dimension)**.** *Let $\mathcal{F}$ be a set of functions from $\mathcal{X}$ to $\mathcal{Y}$, where $|\mathcal{Y}| = 2$. The growth function of $\mathcal{F}$ is defined as follows:*

$$G_{\mathcal{F}}(m) := \max_{(x_1,\dots,x_m)\in\mathcal{X}^m} \left| \left\{ \big(f(x_1),\dots,f(x_m)\big) \in \mathcal{Y}^m \mid f \in \mathcal{F} \right\} \right|. \tag{2.37}$$

Then the VC dimension of $\mathcal{F}$ is given by:

$$\mathrm{VCdim}(\mathcal{F}) := \sup\{m \in \mathbb{N} \mid G_{\mathcal{F}}(m) = 2^m \}. \tag{2.38}$$

The growth function determines the maximal number of distinct classification patterns that functions in $\mathcal{F}$ can achieve on a set of m points. A set $(x_1,\dots,x_m)$ is considered "shattered" by $\mathcal{F}$ if $\mathcal{F}$ can realize all 2^m possible classification patterns on this set. The VC dimension of $\mathcal{F}$ is then defined as the maximal size of a dataset that can be shattered by $\mathcal{F}$. VC dimension is usually closely related to the number of parameters in machine learning methods. For instance, $\mathrm{VCdim}(\mathcal{F}) = n + 1$ for a set of functions realized by a linear classifier of dimension n, given by $\mathcal{F} = \{x \to \mathrm{sign}(\langle \mathbf{w}, x\rangle + b) \mid \mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}\}$. Similarly, VC dimension of fully-connected DNNs with binary output and piecewise-linear activation function in the hidden layers is bounded above by $O(PL\log P + PL^2)$, where L is the network's depth and P is the total number of parameters (Bartlett and Maass, 2003). The following bound holds for sets of functions with finite VC dimension:

Theorem 2.3 (Adopted from Vapnik (2013))**.** *Let $\mathcal{F}$ be a set of functions from $\mathcal{X}$ to $\mathcal{Y}$, where $|\mathcal{Y}| = 2$. Let the associated loss function be given by $\mathcal{L}(\hat{y}, y) = \mathbb{1}_{y\neq\hat{y}}$. Let μ be an arbitrary distribution over $\mathcal{X}\times\mathcal{Y}$, and $S \sim \mu^N$ be a dataset comprising $N \in \mathbb{N}$ i.i.d. samples drawn from μ. Assume that $\mathcal{F}$ has finite VC dimension, i.e., $D := \mathrm{VCdim}(\mathcal{F}) < \infty$. Then for any $\delta \in (0, 1]$ and any sample size $N > D/2$ the following holds with probability at least $1 - \delta$:*

$$\sup_{f\in\mathcal{F}} |\mathcal{L}_\mu(f) - \widehat{\mathcal{L}}_S(f)| \leq \sqrt{\frac{D(\log(2N/D) + 1) + \log(4/\delta)}{N}}. \tag{2.39}$$

The guarantees on the proximity between expected and empirical risks in Theorems 2.2 and 2.3 deteriorate as the model's complexity increases. However, some level of complexity is usually required in practice to ensure that a function with sufficiently low empirical risk exists in $\mathcal{F}$. This trade-off between the model's ability to fit the training data and the classical generalization guarantees is known as the *bias-variance trade-off*. In view of this trade-off, classical statistical learning theory prescribes to select models with limited complexity, ensuring the optimal balance between empirical risk and generalization. This approach to model selection is illustrated in Figure 2.3.

Notice that the bounds in Theorems 2.2 and 2.3 are completely *distribution-free*, i.e., they are valid for any data distribution. This property is characteristic for classical generalization theory results, which assume that the data distribution is unknown in practical machine learning scenarios. The same bounds are also independent of the dataset S and the choice of function $f \in \mathcal{F}$. While these properties ensure very general applicability, they also reveal the

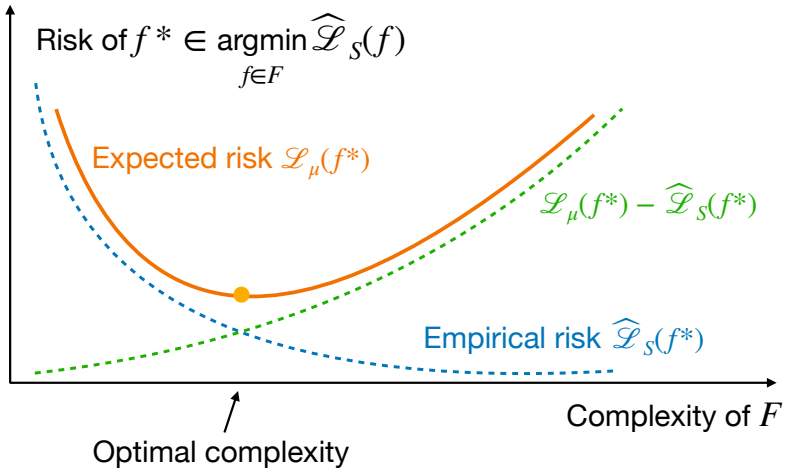


Figure 2.3: Classical generalization curve with bias-variance trade-off.

fundamental limitation of such results: they only capture the worst-case scenario. Therefore, even though the VC dimension bound is optimal in the class of distribution-, dataset-, and function-independent bounds, it does not capture the typical generalization performance of modern DNNs. This is particularly clear in case of overparametrized DNNs, for which $D > N$, and therefore the bound in Theorem 2.3 becomes completely vacuous.

Within the classical statistical learning theory, a number of approaches were developed to partially address the problems of VC bounds. For instance, bounds based on the *Rademacher complexity* depend on the dataset S, which may provide better results in certain scenarios. The notion of Rademacher complexity also allows to derive generalization bounds for regression problems, unlike the VC dimension. However, such bounds still consider the worst-case scenario with respect to the choice of $f \in \mathcal{F}$, and are still vacuous for modern overparametrized DNNs. Other approaches, such as *structural risk minimization* and *margin-based bounds*, additionally introduce the dependence on $f \in \mathcal{F}$ into the classical statistical learning theory bounds. However, all these approaches still lead to vacuous bounds in the overparametrized setting. We refer to Valle-Pérez and Louis (2020) for a comprehensive survey of different distribution-free generalization bounds and their drawback when applied to DNNs.

2.3.2 Modern Perspective on Generalization

One of the biggest challenges of modern machine learning theory is to explain the mechanisms behind the generalization of overparametrized DNNs. As we have seen in the previous section, classical generalization bounds are typically vacuous for overparametrized models. Nevertheless, modern heavily-overparametrized DNNs are known to generalize well in a variety of practical settings. This disparity implies that the classical bias-variance trade-off curve depicted in Figure 2.3 does not adequately describe generalization of overparametrized models. To account for this, the *double descent* generalization curve (illustrated in Figure 2.4)

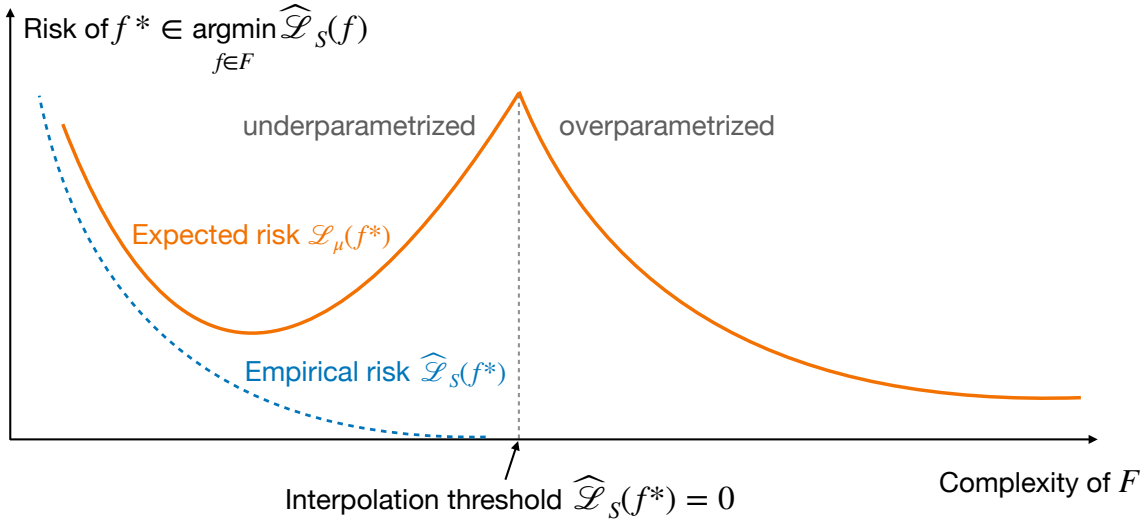


Figure 2.4: Double descent generalization curve.

was proposed in Belkin et al. (2019) as a novel view of generalization in overparametrized machine learning models.

The double descent curve aligns with the classical bias-variance trade-off curve for underparametrized models, which are not rich enough to interpolate the dataset and achieve zero empirical risk. However, increasing the model's capacity beyond the interpolation threshold results in improved generalization, contrary to the traditional perspective of classical statistical learning theory. The double descent curve has been observed empirically for a wide range of models, including DNNs (Nakkiran et al., 2021; Belkin et al., 2019). Theoretical works have also proved the emergence of double descent in a variety of machine learning models, such as linear models (Hastie et al., 2022), random features models (Belkin et al., 2020; Mei and Montanari, 2022), and kernel models (Liu et al., 2021). However, there is currently no theoretical framework that allows to rigorously prove the emergence of double descent in DNNs.

Example: least squares regression Let us now examine how theoretical results on double descent manage to avoid the problems of the classical generalization bounds from the previous section, using a simple linear regression problem as an example. Following Hastie et al. (2022), assume that the data samples $(\mathbf{x}_i, y_i) \in \mathbb{R}^P \times \mathbb{R}$ are i.i.d., and distributed according to the following *data model*:

$$(\mathbf{x}_i, \epsilon_i) \sim \mu_x \times \mu_\epsilon, \quad y_i = \langle \mathbf{w}^*, \mathbf{x}_i \rangle + \epsilon_i, \quad i \in [1, N], \tag{2.40}$$

where μ_x is a distribution on $\mathbb{R}^P$, such that $\mathbb{E}[\mathbf{x}_i] = 0$, $Cov(\mathbf{x}_i) = \Sigma$, and μ_ϵ is a distribution on $\mathbb{R}$, such that $\mathbb{E}[\epsilon_i] = 0$, $Var(\epsilon_i) = \sigma^2$. Let $\mathbf{X} \in \mathbb{R}^{N \times P}$ denote the features matrix, which contains samples $\mathbf{x}_i$ as rows, and $\mathbf{y} \in \mathbb{R}^N$ denote the vector of target outputs. Then the empirical risk minimization problem associated with the least squares (MSE loss) regression

is given by:

$$\hat{\mathbf{w}} \in \arg\min_{\mathbf{w}\in\mathbb{R}^P} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2, \tag{2.41}$$

where $\hat{\mathbf{w}}$ is the output of the ERM algorithm. Now we can notice that the minimizer of this problem is not unique for overparametrized models, which satisfy $P > N$. In fact, there is a linear subspace of minimizers with dimension $P - N$. Clearly, not all the minimizers of the empirical risk in this subspace generalize equally well for the given data model. However, double descent results additionally take into account that the training is carried out by GD algorithm. Then it is possible to use the following well-known result regarding the implicit bias of GD for the least squares regression:

Theorem 2.4. *Consider running GD algorithm for the minimization problem (2.41) with initialization $\mathbf{w}^{(0)} = 0$ and learning rate $0 < \eta \leq 1/\lambda_{\max}(\mathbf{X}^\top\mathbf{X})$, where $\lambda_{\max}(\mathbf{X}^\top\mathbf{X})$ is the largest eigenvalue of $\mathbf{X}^\top\mathbf{X}$. Then the iterates are given by*

$$\mathbf{w}^{(k)} = \mathbf{w}^{(k-1)} - \eta\mathbf{X}^\top(\mathbf{X}\mathbf{w}^{(k-1)} - \mathbf{y}), \quad k \in \mathbb{N}. \tag{2.42}$$

And the algorithm converges to the solution with minimal ℓ_2 norm:

$$\hat{\mathbf{w}} := \lim_{k\to\infty} \mathbf{w}^{(k)} = \arg\min\{\|\mathbf{v}^*\|_2^2 \mid \mathbf{v}^* \in \arg\min_{\mathbf{v}\in\mathbb{R}^P} \|\mathbf{X}\mathbf{v} - \mathbf{y}\|_2^2\}. \tag{2.43}$$

Moreover, the solution has the following closed-form expression:

$$\hat{\mathbf{w}} = (\mathbf{X}^\top\mathbf{X})^+\mathbf{X}^\top\mathbf{y}, \tag{2.44}$$

where $(\mathbf{X}^\top\mathbf{X})^+$ is the Moore-Penrose inverse of $\mathbf{X}^\top\mathbf{X}$.

Therefore, it is enough to consider the expected risk of a single function $\hat{f}(\mathbf{x}) := \langle\hat{\mathbf{w}}, \mathbf{x}\rangle$, to which GD converges for a given dataset:

$$\mathcal{L}_\mu(\hat{f}) = \mathbb{E}_{(\mathbf{x},\epsilon)\sim\mu_x\times\mu_\epsilon}\left[\left(\langle\hat{\mathbf{w}}, \mathbf{x}\rangle - y\right)^2\right]. \tag{2.45}$$

This is in stark contrast with the results of Theorems 2.2 and 2.3, which bound the risk simultaneously for all the functions that can be realized by a given model, independently of the dataset and the training algorithm. The following theorem gives an explicit expression for the expected risk $\mathcal{L}_\mu(\hat{f})$ of the linear regression model with additional assumptions on the input distribution:

Theorem 2.5 (Adopted from Hastie et al. (2022))**.** *Assume the data is distributed according to (2.40) with $\Sigma = \mathbb{I}$. Assume additionally that the distribution μ_x has finite moment of order $4 + k$ for some $k > 0$, and that $\|\mathbf{w}^*\| = r^2$ for all $N, P \in \mathbb{N}$. Then the following holds for the expected risk of the linear regression model trained using GD with MSE loss:*

$$\mathcal{L}_\mu(\hat{f}) \xrightarrow[\substack{N\to\infty, P\to\infty,\\ P/N\to\gamma\in\mathbb{R}}]{} \begin{cases} \sigma^2\dfrac{\gamma}{1-\gamma} & \text{for } \gamma < 1, \\ r^2\left(1 - \dfrac{1}{\gamma}\right) + \sigma^2\dfrac{1}{1-\gamma} & \text{for } \gamma > 1, \end{cases} \tag{2.46}$$

where $\gamma := P/N$ is the ratio between the number of parameters and the number of samples in the dataset.

The expressions for the expected risk in Theorem 2.5 follow the double descent pattern: the risk increases with the number of parameters in the underparametrized case, and decreases in the overparametrized case. Intuitively, the variance decreases in the overparametrized case because the space of the interpolating functions becomes larger with the parameters count. Therefore, the minimal ℓ_2-norm in this space can only decrease with more parameters. In other words, stronger overparametrization also implies stronger implicit regularization.

While we only considered the least squares regression here, double descent results for random features models or kernel models rely on the same principles. Namely, they compute the empirical risk only for a single data-dependent choice of the empirical risk minimizer, and exhibit double descent due to the growing regularity of this minimizer in the overparametrized setting.

Generalization and implicit bias Compared to the classical generalization theory bounds, current findings on double descent have sacrificed a lot of generality. Indeed, unlike the classical bounds, these results depend on the training procedure and the data distribution. Is this loss of generality necessary to derive non-vacuous generalization results for overparametrized models?

While there might be room to relax assumptions about the data distribution in existing results, the dependence on the training algorithm is a fundamental aspect of generalization theory for overparametrized models. Indeed, overparametrization implies that multiple functions in $\mathcal{F}$ achieve zero empirical risk, but usually not all of these functions generalize equally well. In the example of the overparametrized least squares regression that we considered above, the empirical risk minimizers can take the form $\tilde{\mathbf{w}} = \mathbf{w}^* + \alpha\Delta\mathbf{w}$, where $\alpha \in \mathbb{R}$ and $\Delta\mathbf{w} \in \{\mathbf{v} \in \mathbb{R}^P \mid \mathbf{X}\mathbf{v} = 0, \|\mathbf{v}\| = 1\}$. Therefore, it is easy to see that the expected risk for such a minimizer is given by

$$\mathcal{L}_\mu(\tilde{f}\,) = \mathbb{E}_{(\mathbf{x},\epsilon)\sim\mu_x\times\mu_\epsilon}[(\alpha\langle\Delta\mathbf{w},\mathbf{x}\rangle - \epsilon)^2] = \alpha^2\|\Delta\mathbf{w}\|_\Sigma^2 + \sigma^2, \tag{2.47}$$

where $\tilde{f}(\mathbf{x}) = \langle\tilde{\mathbf{w}},\mathbf{x}\rangle$ and $\|\Delta\mathbf{w}\|_\Sigma^2 = \Delta\mathbf{w}^\top\Sigma\Delta\mathbf{w}$. Then, provided that we can choose $\Delta\mathbf{w}$ such that $\|\Delta\mathbf{w}\|_\Sigma > 0$, there exist empirical risk minimizers with arbitrarily large expected risk. Therefore, relying on the knowledge about the exact minimizer chosen by GD is essential to derive any meaningful generalization guarantee.

The convergence of gradient-based algorithms to minimizers with certain properties is at the focus of the *implicit bias* literature, which we discussed in the introduction. Clearly, generalization performance of modern overparametrized models is deeply connected to the implicit bias of common optimization algorithms. This, in turn, highlights the importance of studying the training dynamics of machine learning models, as it allows to derive implicit bias results.

Towards generalization guarantees for DNNs Empirical evidence suggests that the double descent phenomenon can serve as a suitable framework for understanding

the generalization of modern DNNs (Nakkiran et al., 2021). However, a big obstacle for deriving theoretical generalization results for DNNs is the current lack of satisfactory results regarding the implicit bias of DNNs. Indeed, as we discussed in Section 1.2, existing results regarding training dynamics and implicit bias of DNNs rely on various simplifications, which do not accurately reflect the reality of modern deep learning. In the next section, we focus on one such simplification, called the *kernel regime* of DNNs, which is the central theme of this thesis. While the kernel regime of DNNs has been a breakthrough in deep learning theory, the contributions of this thesis and numerous relevant works have highlighted its limitations in capturing empirical properties of DNNs. Therefore, the perspective we adopt in this thesis is that new approaches, grounded in strong empirical evidence, are necessary to demystify the generalization of DNNs.

2.4 Neural Tangent Kernel

Let us consider the gradient flow dynamics of a NN $f_\theta : \mathbb{R}^{n_0} \to \mathbb{R}$ with trainable parameters $\theta \in \mathbb{R}^P$, which is trained on a dataset $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Here we consider the case of NNs with scalar output $n_L = 1$ for simplicity. The training dynamics of the NN's parameters is given by

$$\dot{\theta} = -\nabla \widehat{\mathcal{L}}_S(\theta) = -\frac{1}{N}\sum_{i=1}^{N} \nabla f_\theta(\mathbf{x}_i) \frac{\partial \mathcal{L}(f_\theta(\mathbf{x}_i), y_i)}{\partial f_\theta(\mathbf{x}_i)}. \tag{2.48}$$

Then, by chain rule, the corresponding dynamics of the DNN's output function for any input $\mathbf{x} \in \mathbb{R}^{n_0}$ is given by:

$$\dot{f}_\theta(\mathbf{x}) = \langle \nabla f_\theta(\mathbf{x}), \dot{\theta} \rangle = -\frac{1}{N}\sum_{i=1}^{N} \big\langle \nabla f_\theta(\mathbf{x}), \nabla f_\theta(\mathbf{x}_i) \big\rangle \frac{\partial \mathcal{L}(f_\theta(\mathbf{x}_i), y_i)}{\partial f_\theta(\mathbf{x}_i)}. \tag{2.49}$$

Therefore, the dynamics in the function space is controlled by an inner product kernel $\Theta(\mathbf{x}, \tilde{\mathbf{x}}) := \big\langle \nabla f_\theta(\mathbf{x}), \nabla f_\theta(\tilde{\mathbf{x}}) \big\rangle$, which is known as the Neural Tangent Kernel (NTK).

Definition 2.12 (Neural Tangent Kernel)**.** *Consider a NN $f_\theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ with trainable parameters $\theta \in \mathbb{R}^P$. Then the NTK of this network $\Theta : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_L \times n_L}$ is given by*

$$\Theta_{k,s}(\mathbf{x}, \tilde{\mathbf{x}}) := \big\langle \nabla f_{\theta,k}(\mathbf{x}), \nabla f_{\theta,s}(\tilde{\mathbf{x}}) \big\rangle, \quad \mathbf{x}, \tilde{\mathbf{x}} \in \mathbb{R}^{n_0}, \quad k, s \in [n_L], \tag{2.50}$$

where $f_{\theta,k} : \mathbb{R}^{n_0} \to \mathbb{R}$ is the k-th output neuron of the NN, and $\nabla f_{\theta,k} : \mathbb{R}^{n_0} \to \mathbb{R}^P$ denotes the gradient of f_k with respect to all the parameters of the NN.

Although here we specifically consider NNs, notice that equations (2.48), (2.49) hold for other machine learning models as well. In particular, for linear models we have $\Theta(\mathbf{x}, \tilde{\mathbf{x}}) := \langle \mathbf{x}, \tilde{\mathbf{x}} \rangle$, and for random features models $\Theta(\mathbf{x}, \tilde{\mathbf{x}}) := \langle \Phi(\mathbf{x}), \Phi(\tilde{\mathbf{x}}) \rangle$ with an appropriate feature map Φ. Finally, the role of the NTK in NNs' dynamics is analogous to the role of a kernel in kernel gradient flow. Therefore, the NTK generalizes the concepts of features matrices and kernels to NNs. However, unlike traditional kernels, the NTK depends on the NN's parameters, and therefore it changes during training and inherits randomness from the initialization.

2.4.1 Infinite-Width Limit

Since the NTK is random and changes during training, theoretical analysis of dynamics (2.49) is extremely challenging in the general case. However, a famous work by Jacot et al. (2018) showed that the NTK becomes deterministic and constant in the infinite-width-limit of NNs under certain conditions. This setting is called the *kernel regime* or the *NTK regime* of NNs. The dynamics of NNs in the NTK regime are equivalent to kernel gradient flow, enabling the derivation of theoretical results regarding implicit bias and generalization of NNs. In this section, we introduce the kernel regime of DNNs and its applications.

The infinite-width limit of the NTK is traditionally considered in the so-called *NTK parametrization*, where the NN's trainable parameters are variables $\{(\mathbf{w}^\ell, \beta^\ell)\}_{\ell=1}^L$, which are in the following relationship to the weights and biases of the NN:

$$\mathbf{W}_{i,j}^\ell = \frac{\sigma_w}{\sqrt{n_{\ell-1}}}\mathbf{w}_{i,j}^\ell, \quad \mathbf{b}_i^\ell = \sigma_b \beta_i^\ell. \tag{2.51}$$

The Gaussian initialization, equivalent to (2.26), is expressed as follows in the NTK parametrization:

$$\mathbf{w}_{i,j}^\ell \sim \mathcal{N}(0,1) \text{ i.i.d.}, \quad \beta_i^\ell \sim \mathcal{N}(0,1) \text{ i.i.d.} \tag{2.52}$$

Clearly, the reparametrization does not change the distribution of any variables of the NN's forward pass. However, it rescales the NN's Jacobians as follows:

$$\nabla_{\mathbf{w}^\ell} f_\theta(\mathbf{x}) = \frac{\sigma_w}{\sqrt{n_{\ell-1}}}\nabla_{\mathbf{W}^\ell} f_\theta(\mathbf{x}), \quad \nabla_{\beta^\ell} f_\theta(\mathbf{x}) = \sigma_b \nabla_{\mathbf{b}^\ell} f_\theta(\mathbf{x}). \tag{2.53}$$

Therefore, since the NTK is defined as the inner product of the NN's Jacobians with respect to the trainable parameters, this reparametrization introduces a width-dependent rescaling of the NTK summands. Note that during the training process, the NTK parametrization can also be interpreted as a suitable rescaling of the learning rates for individual parameters.

The NTK parametrization is convenient in the infinite-width limit, as the width-dependent rescaling ensures that the NTK does not diverge in this limit. Then it is possible to derive the following result regarding the concentration of the NTK at initialization:

Theorem 2.6 (Infinite-width NTK is deterministic)**.** *Consider a fully-connected NN with fixed depth $L \in \mathbb{N}$ and linear activation in the output layer, i.e., $\phi_L(x) = x$. Assume that the activation function in all the hidden layers is a Lipschitz continuous function ϕ. Assume further that the NN is parametrized according to (2.51) and initialized as in (2.52). Then, in the infinite-width limit $n_1, \ldots, n_{L-1} \to \infty$, the following holds for the NTK at initialization $\Theta^{(0)}$ computed on any inputs $\mathbf{x}, \tilde{\mathbf{x}} \in \mathbb{R}^{n_0}$:*

$$\Theta^{(0)}(\mathbf{x}, \tilde{\mathbf{x}}) \xrightarrow{p} \Theta^\infty(\mathbf{x}, \tilde{\mathbf{x}}) \mathbb{I}_{n_L}. \tag{2.54}$$

Moreover, Θ^∞ can be computed recursively using the following expression:

$$\Theta^\infty(\mathbf{x}, \tilde{\mathbf{x}}) = \sum_{\ell=1}^{L} \left(\Sigma^{\ell-1}(\mathbf{x}, \tilde{\mathbf{x}}) \prod_{\ell'=\ell}^{L} \dot{\Sigma}^{\ell'}(\mathbf{x}, \tilde{\mathbf{x}}) \right), \tag{2.55}$$

where $\Sigma^\ell(\mathbf{x}, \tilde{\mathbf{x}})$ *is the Gaussian process covariance defined in* (2.30)*, and*[2]

$$\dot{\Sigma}^\ell(\mathbf{x}, \tilde{\mathbf{x}}) = \mathop{\mathbb{E}}_{(u,v)\sim\mathcal{N}(0,\Sigma^\ell(\mathbf{x},\tilde{\mathbf{x}}))}\big[\phi'(u)\phi'(v)\big]. \tag{2.56}$$

This result was originally proved in the seminal work of Jacot et al. (2018) for the sequential setting, where limits with respect to the width of each layer are taken one by one. While the sequential limit is not a good model for DNNs that typically have comparable widths in different layers, this result was generalized to simultaneous limit in multiple following works (Yang, 2020a; Arora et al., 2019b). Some works have also derived convergence rates for this result (Arora et al., 2019b; Huang and Yau, 2020).

The second important result states that the NTK does not change during training in the infinite-width limit:

Theorem 2.7 (Infinite-width NTK is constant)**.** *Consider a NN as described in Theorem 2.6. Additionally, assume that the activation function* ϕ *is differentiable, and its derivative* ϕ' *is Lipschitz continuous. The NN is trained using GD on a dataset* $S = \big\{(\mathbf{x}_i, \mathbf{y}_i)\big\}_{i=1}^N$*, contained in a compact set, and such that* $\mathbf{x}_i \neq \mathbf{x}_j$ *for all* $i \neq j$*. Assume the infinite-width NTK matrix* $\Theta^\infty(\mathbf{X}, \mathbf{X}) = \big\{\Theta^\infty(\mathbf{x}_i, \mathbf{x}_j)\big\}_{i,j=1}^N \in \mathbb{R}^{N\times N}$ *is full-rank, and the learning rate is set to* $\eta < \eta_{\max} := 2/(\lambda_{\max} + \lambda_{\min})$*, where* $\lambda_{\max,\min}$ *denote the largest and the smallest eigenvalues of* $\Theta^\infty(\mathbf{X}, \mathbf{X})$*. Then for any GD step* $t \in \mathbb{N}$*, the following holds in the infinite-width limit* $n_1, \ldots, n_{L-1} \to \infty$*:*

$$\Theta_{k,s}^{(t)}(\mathbf{X}, \mathbf{X}) \xrightarrow{p} \Theta^\infty(\mathbf{X}, \mathbf{X})\delta_{k,s}, \quad k, s \in [n_L]. \tag{2.57}$$

The first result regarding the limit of the NTK during training was proved in Jacot et al. (2018) for the sequential limit and gradient flow training, while following works generalized it to the simultaneous limit. The formulation that that we adopted here is an asymptotic version of the results in Lee et al. (2019), which showed that the above limit converges at a rate $O(1/\sqrt{n})$ uniformly over t. An analogous result was also proven for ReLU DNNs, which are not covered by Theorem 2.7, in Arora et al. (2019b). Finally, a stronger result regarding the convergence rate of the above limit was derived using the Neural Tangent Hierarchy (NTH) in Huang and Yau (2020).

The results of Theorems 2.6 and Theorem 2.7 together define the NTK regime of NNs, where the NTK is constant and deterministic during the whole training process.

2.4.2 Training Dynamics in the NTK Regime

The gradient flow dynamics of NNs (2.49) takes the following form in the NTK regime:

$$\dot{f}_\theta(\mathbf{x}) = -\frac{1}{N}\sum_{i=1}^N \Theta^\infty(\mathbf{x}, \mathbf{x}_i)\frac{\partial \mathcal{L}(f_\theta(\mathbf{x}_i), y_i)}{\partial f_\theta(\mathbf{x}_i)}. \tag{2.58}$$

[2] Lipschitz ϕ ensures that the derivative ϕ' exists almost everywhere, so the expectation is well-defined.

In the special case of MSE loss, this can be expressed as the following matrix ODE:

$$\dot{f}_\theta(\mathbf{X}) = -\frac{1}{N}\Theta^\infty(\mathbf{X},\mathbf{X})(f_\theta(\mathbf{X}) - \mathbf{Y}), \tag{2.59}$$

where $\mathbf{X} \in \mathbb{R}^{N\times n_0}$ is a matrix comprising all the inputs $\mathbf{x}_i$, $i \in [1, N]$ from the training dataset as rows, and $\mathbf{Y} \in \mathbb{R}^N$ is a vector of target outputs $y_i, i \in [1, N]$. Therefore, the dynamics in the NTK regime is governed by linear equations, and can be seen as the linearization of the NN's dynamics around its initialization.

The dynamics (2.59) has an analytical solution, expressed as follows:

$$f_\theta^{(t)}(\mathbf{X}) = \mathbf{Y} + (f_\theta^{(0)}(\mathbf{X}) - \mathbf{Y}) \exp\big(-t\Theta^\infty(\mathbf{X},\mathbf{X})\big), \tag{2.60}$$

where $\exp\big(-t\Theta^\infty(\mathbf{X},\mathbf{X})\big)$ is the matrix exponential. Therefore, it is possible to study convergence of gradient flow in the NTK regime. Indeed, we see that the NN's error on the training set converges to zero exponentially in the above equation, given that the NTK matrix is positive-definite.

For an arbitrary input $\mathbf{x} \in \mathbb{R}^{n_0}$, we can also give an explicit expression for the training dynamics, given that the infinite-width NTK matrix is invertible[3]:

$$f_\theta^{(t)}(\mathbf{x}) = f_\theta^{(0)}(\mathbf{x}) - \Theta^\infty(\mathbf{x},\mathbf{X})\Theta^\infty(\mathbf{X},\mathbf{X})^{-1}\big(\mathbb{I} - e^{-t\Theta^\infty(\mathbf{X},\mathbf{X})}\big)(f_\theta^{(0)}(\mathbf{X}) - \mathbf{Y}). \tag{2.61}$$

Therefore, it is possible to study generalization error of NNs in the kernel regime at any training time t using the above expression.

2.4.3 Generalization Bounds Based on the NTK

Several works derived generalization bounds for NNs in the NTK regime. The characteristic property of such bounds is their independence of width. I.e., the number of parameters of the NN can grow without worsening the generalization guarantee. The following bound was derived in Arora et al. (2019a) for sufficiently-wide NNs with one hidden layer trained for sufficiently many GD steps:

$$\mathcal{L}_\mu(f_\theta) \leq \sqrt{\frac{2\mathbf{Y}^\top(\Theta^\infty(\mathbf{X},\mathbf{X}))^{-1}\mathbf{Y}}{N}} + O\left(\sqrt{\frac{\log\frac{N}{\lambda_0\delta}}{N}}\right), \tag{2.62}$$

where the bound holds with probability at least $1-\delta$. A similar bound, generalized for NNs of arbitrary depth L initialized at the EOC, has been formulated in Cao and Gu (2019). This bound exhibits a linear growth with depth, indicating that the generalization guarantees deteriorate for deeper networks in the NTK regime. However, this observation appears inconsistent with empirical evidence. Our contributions in Section 3.1 partially concern generalization in the NTK regime and its dependence on depth. In particular, we discuss how the infinite-width NTK changes with depth, and how its properties lead to poor generalization guarantees.

[3] As we see e.g. in Section 3.2, the infinite-width NTK is indeed invertible if all the points in the training dataset are distinct.

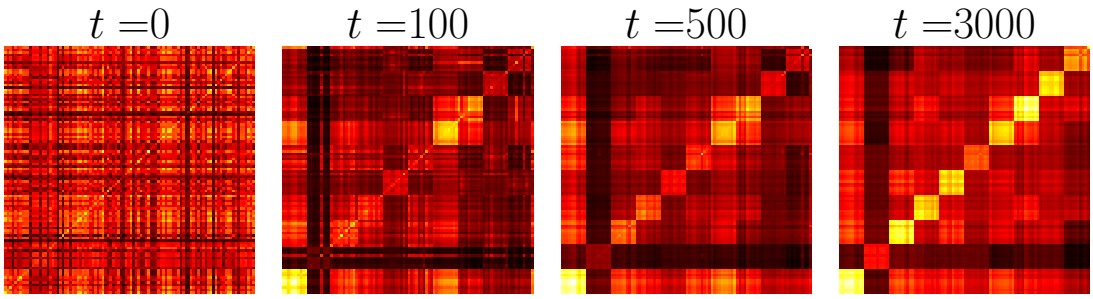


Figure 2.5: NTK alignment during training of a fully-connected ReLU DNNs with $L = 20$ and widths $n_\ell = 300$ for all $0 < \ell < L$ on MNIST. The NTK matrix develops an approximate block structure during training. The heatmaps show the NTK matrix on MNIST subsample of size 100 at epoch $t \in \{0, 100, 500, 3000\}$. The subsample is arranged so that diagonal blocks of size 10 contain pairwise NTK values on each class. Figure from Seleznova and Kutyniok (2022b).

2.4.4 NTK Alignment

The NTK at initialization is *label-agnostic*, meaning that its value for a pair $(x, \tilde{x})$ remains independent of whether the labels of x and $\tilde{x}$ are identical or not. Therefore, DNNs in the NTK regime do not learn and utilize any label-dependent features. Label-agnostic features, however, may not offer an optimal representation system for an arbitrary task. Indeed, since DNNs can perform equally well on various tasks using the same dataset, such as recognizing different objects in the same set of images, label-agnostic kernel is unlikely to explain the performance of trained DNNs.

Several studies have explored the advantages of incorporating label information into kernels (Cristianini et al., 2001; Gönen and Alpaydin, 2011). These studies consider the *alignment* between a given kernel matrix $\mathbf{K} := k(\mathbf{X}, \mathbf{X}) \in \mathbb{R}^{N \times N}$ and the "ideal kernel", which is proportional to the corresponding labels matrix $\mathbf{Y}\mathbf{Y}^\top$:

$$A(\mathbf{K}, \mathbf{Y}\mathbf{Y}^\top) := \frac{\langle \mathbf{K}, \mathbf{Y}\mathbf{Y}^\top \rangle_F}{\sqrt{\langle \mathbf{K}, \mathbf{K} \rangle_F \langle \mathbf{Y}\mathbf{Y}^\top, \mathbf{Y}\mathbf{Y}^\top \rangle_F}}, \tag{2.63}$$

where $\langle \mathbf{K}_1, \mathbf{K}_2 \rangle_F := \sum_{i,j=1}^N k_1(x_i, x_j) k_2(x_i, x_j)$. Then higher kernel alignment values are associated with better generalization performance of the corresponding kernel methods. Hence, kernel alignment can be interpreted as a metric indicating the compatibility between a kernel and a specific task. In the context of DNNs, Chen et al. (2020) argued that label-agnosticism of the NTK could account for the performance gap observed between trained DNNs and the NTK regime. They demonstrated that adding a label-dependent term to the infinite-width NTK enhances the performance of the kernel. Therefore, it is crucial to characterize the label-awareness of the empirical NTK to gain insights into the properties of trained DNNs.

Multiple recent papers have observed that the empirical NTK of finite-width DNNs aligns with the labels matrix $\mathbf{Y}\mathbf{Y}^\top$ during training (Baratin et al., 2021; Shan and Bordelon, 2022; Atanasov et al., 2021; Seleznova and Kutyniok, 2022b). This process characterizes feature learning in DNNs and is called *NTK alignment* in the literature. In agreement with the intuition from kernel methods, higher NTK alignment values are correlated with better performance of DNNs (Atanasov et al., 2021; Seleznova et al., 2023). Figure 2.5 gives an example of the NTK alignment arising during training a fully-connected DNN on MNIST. In classification problems, the labels matrix $\mathbf{Y}\mathbf{Y}^\top$ has a block structure, with diagonal blocks filled with ones and non-diagonal blocks filled with zeros. Therefore, the NTK alignment in these problems manifests as an emergence of a block structure in the NTK matrix. This observation forms the basis for the NTK block structure assumption, which we introduce in Section 3.3. Additionally, we present numerous visual examples of the NTK block structure in trained DNNs and conduct experiments to showcase the dynamics of NTK alignment during training in Section 3.3.

2.5 Notation

The set of natural numbers is denoted by $\mathbb{N}$. The set of real numbers is denoted by $\mathbb{R}$. The set of non-negative real numbers is denoted by $\mathbb{R}_+$. $[N_1, N_2]$ is a set of integers $\{N_1, \dots, N_2\}$. Operation $\circ$ denotes composition of functions. Operations $\odot$ denotes composition of functions, where the outer function in the composition is applied component-wise. Set product is denoted by $\bigtimes$ (or $\times$ for two sets). Set isomorphism is denoted by $\simeq$. Convergence in probability is denoted by $\xrightarrow{p}$. In the context of NNs, we generally denote by $f_\theta(\mathbf{x})$ the output function of a DNN with parameters θ computed on the input $\mathbf{x}$. We denote $\nabla_{\mathbf{v}} f_\theta(\mathbf{x})$ the gradient of the DNN's output function computed on input $\mathbf{x}$ with respect to the subset of parameters $\mathbf{v}$, where the parameters are set to their current values, given by θ. In the context of gradient flow, $\dot{f}$ denotes the derivative of a function f with respect to the time variable t. For a kernel function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, we denote by $k(\mathbf{X}, \tilde{\mathbf{X}}) \in \mathbb{R}^{N \times \tilde{N}}$ the matrix of the kernel values computed for all the pairs of rows in matrices $\mathbf{X} \in \mathbb{R}^{N \times n}$, $\tilde{\mathbf{X}} \in \mathbb{R}^{\tilde{N} \times n}$, i.e., $k(\mathbf{X}, \tilde{\mathbf{X}})[i, j] = k(\mathbf{X}[i], \tilde{\mathbf{X}}[j])$. For any probability distribution μ, we assume a suitable underlying probability space. $\mathbf{E}_{x \sim \mu}$ denotes the expectation with respect to random variable x distributed according to μ. When the random variables and the distribution are not specified, the expectation is taken with respect to all the relevant random variables. $Cov(x, y)$ denotes the covariance of random variables x and y.

Chapter 3

Contributing Papers

3.1 Analyzing Finite Neural Networks: Can We Trust Neural Tangent Kernel Theory?

Contributing article: Seleznova, M. and Kutyniok, G. (2022a). Analyzing Finite Neural Networks: Can We Trust Neural Tangent Kernel Theory? In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference*, pages 868–895. PMLR.

Author contributions: Mariia Seleznova developed the original research idea to analyze the NTK of finite-width DNNs as a function of initialization hyperparameters and the network's depth to determine when the NTK regime approximates realistic DNNs. Mariia Seleznova formulated all the theorems and derived all the proofs presented in the paper, designed and programmed all the numerical experiments, wrote the paper's main text and appendices, and designed all the figures. As the main author, Mariia Seleznova also managed the publication process: paper submission to the conference, writing a rebuttal after the initial reviews, addressing reviewers' concerns, and producing the camera-ready version of the paper. Gitta Kutyniok took part in the project discussions at all the stages, provided feedback, reviewed and proofread the paper.

Additional resources:

- Paper link: https://proceedings.mlr.press/v145/seleznova22a.html
- Slides: https://msml21.github.io/slides/id44.pdf
- Video presentation: Google Drive link

Proceedings of Machine Learning Research vol 145:1–28, 2021 2nd Annual Conference on Mathematical and Scientific Machine Learning

Analyzing Finite Neural Networks: Can We Trust Neural Tangent Kernel Theory?

Mariia Seleznova SELEZNOVA@MATH.LMU.DE and **Gitta Kutyniok** KUTYNIOK@MATH.LMU.DE
Department of Mathematics, Ludwig-Maximilians-Universität München, Munich, Germany

Editors: Joan Bruna, Jan S Hesthaven, Lenka Zdeborova

Abstract

Neural Tangent Kernel (NTK) theory is widely used to study the dynamics of infinitely-wide deep neural networks (DNNs) under gradient descent. But do the results for infinitely-wide networks give us hints about the behavior of real finite-width ones? In this paper, we study empirically when NTK theory is valid in practice for fully-connected ReLU and sigmoid DNNs. We find out that whether a network is in the NTK regime depends on the hyperparameters of random initialization and the network's depth. In particular, NTK theory does not explain the behavior of sufficiently deep networks initialized so that their gradients explode as they propagate through the network's layers: the kernel is random at initialization and changes significantly during training in this case, contrary to NTK theory. On the other hand, in the case of vanishing gradients, DNNs are in the the NTK regime but become untrainable rapidly with depth. We also describe a framework to study generalization properties of DNNs, in particular the variance of network's output function, by means of NTK theory and discuss its limits.

Keywords: Deep Neural Networks (DNN), Neural Tangent Kernel (the NTK)

1. Introduction

Deep neural networks (DNNs) have gained a lot of popularity in the last decades due to their success in a variety of domains, such as image classification (Krizhevsky et al., 2012), speech recognition (Hannun et al., 2014), playing games (Mnih et al., 2013), etc. Consequently, there has been a tremendous interest in the theoretical properties of DNNs: expressivity (Montufar et al., 2014), optimization (Goodfellow et al., 2014) and generalization (Hardt et al., 2016). However, many aspects of DNNs, in particular their surprising generalization properties, still remain unclear to the community (Zhang et al., 2016).

To study theoretical properties of DNNs, numerous recent papers have considered them in the infinite-width limit. In particular, there is a line of research which shows that untrained fully-connected networks of depth L and widths $M_1, \dots, M_L$ with weights and biases initialized randomly as

$$\mathbf{W}_{ij}^l \sim \mathcal{N}(0, \sigma_w^2/M_l), \mathbf{b}_i^l \sim \mathcal{N}(0, \sigma_b^2) \tag{1}$$

behave as Gaussian processes (GP) in the infinite-width limit (for any $l \in [1, L], M_l \to \infty$) (Lee et al., 2017; Matthews et al., 2018; Novak et al., 2018). These GPs are then fully described by a so-called Neural Network Gaussian Process (NNGP) kernel, and a number of publications have studied properties of this kernel depending on the network's depth and initialization hyperparameters (Poole et al., 2016; Schoenholz et al., 2016). These works developed a *mean field* theory formalism for NNs and identified that there exist two situations – depending on hyperparameters (σ_w^2, σ_b^2) – in which

signal propagation through the network differs substantially: *ordered* and *chaotic* phases, which correspond to vanishing and exploding gradients. However, these results only concern untrained randomly initialized networks.

There have also been recent successes in the theory of trained infinitely wide DNNs. In particular, it has been shown that the evolution of NN's output during gradient flow training can be captured by a so-called Neural Tangent Kernel (NTK) Θ^t (Jacot et al., 2018; Arora et al., 2019; Yang, 2020):

$$\begin{aligned} \frac{df^t(x)}{dt} &= -\frac{1}{S}\sum_{s=1,\dots S} \Theta^t(x, x_s)\cdot[f^t(x_s) - y_s], \\ \Theta^t(x_i, x_j) &= \nabla_w f^t(x_i)^T \nabla_w f^t(x_j), \quad w = \{\mathbf{W}^l, \mathbf{b}^l\}_{l=1,\dots L}, \end{aligned} \tag{2}$$

where $f^t(x)$ is the network's output on x at time t and $D = \{(x_s, y_s)\}_{s=1,\dots S}$ is the training set. In general, the NTK changes during training time t and the dynamics in (2) is complex. However, as layers' widths tend to infinity with fixed depth, it can be shown that the NTK stays constant during training and equal to its initial value:

$$\Theta^t(x_i, x_j) = \Theta^0(x_i, x_j). \tag{3}$$

Moreover, the NTK at initialization converges to a deterministic kernel Θ^* in the same limit:

$$\Theta^0(x_i, x_j) \xrightarrow[M_l\to\infty]{} \Theta^*(x_i, x_j). \tag{4}$$

These two results allow to dramatically simplify the analysis of DNNs behavior, as the dynamics in (2) becomes identical to kernel regression and the ODE has a closed-formed solution.

However, some recent papers argue that the success of DNNs cannot be explained by their behavior in the infinite-width limit (Chizat et al., 2019; Hanin and Nica, 2019). One justification for this view is that no feature learning occurs when (3) and (4) hold, as the NTK stays constant during training and depends only on the parameters at initialization. Moreover, the NTK becomes completely data-independent in the infinite-depth limit, which suggests poor generalization performance (Xiao et al., 2019). That is why, to study properties of real DNNs, it is important to understand when and if NTK theory can be applied to finite-width NNs.

1.1. Contribution

Our aim in this work is to understand when the inferences of NTK theory (3) and (4) hold for real NNs depending on hyperparameters $(\sigma_w^2, \sigma_b^2, L, M)$ and what this implies for the existing theoretical results about DNNs based on NTK theory. The contributions of our work are as follows:

- **NTK variance at initialization.** We study empirically when the NTK is approximately deterministic at initialization for finite-width fully-connected ReLU and `tanh` networks with different hyperparameters $(\sigma_w^2, \sigma_b^2, L, M)$. Our results suggest that, depending on the initialization hyperparameters (σ_w^2, σ_b^2), there is a phase in the hyperparameter space where the NTK is close to deterministic for any depth L, so (4) holds. However, there is also a phase where the NTK variance grows with L/M, so (4) does not hold for deep networks. Following the terminology from Poole et al. (2016), we will call these phases *ordered* and *chaotic*, respectively.

- **NTK change during training.** We also empirically study changes in the NTK matrix during gradient descent training for ReLU and `tanh` networks. Our results show that, in the ordered phase, the relative change in the NTK matrix norm caused by training is small and does not increase with L, so (3) holds. However, in the chaotic phase the NTK matrix change during training is large and grows with depth L. This implies that (3) does not hold, i.e. DNNs initialized in the chaotic phase do not behave as NTK theory suggests.

- **NTK theory approach for generalization.** Some recent publications analyze properties of the NTK and draw conclusions about DNNs' generalization thereof (Xiao et al., 2019; Geiger et al., 2020). Other authors argue that the behavior of networks in the NTK regime is trivial and does not yield good generalization properties, that are however observed for DNNs in practice (Chizat et al., 2019). We show how to compute data-independent variance of the network's output when it evolves according to NTK theory. However, given our empirical results for when NTK theory is applicable, we discover that these findings do not explain the behavior of finite-width networks in most of the hyperparameters space $(\sigma_w^2, \sigma_b^2, L, M)$.

1.2. Related work

This work adds to the line of research that studies the correspondence between finite- and infinite-width DNNs. In particular, the difference between theoretical (infinite-width) and empirical (finite-width) NTK. In this section, we survey the prior results in this direction and position our contribution within them.

A number of papers have studied the convergence of the empirical NTK at initialization to the theoretical NTK. The first fundamental result of NTK theory is that the NTK converges to a deterministic limit as M goes to infinity (Jacot et al., 2018). The following work proved a non-asymptotic bound on minimal M required to guarantee this convergence in case of ReLU networks (Arora et al., 2019). This bound on M depends on the depth as $O(L^6 log(L))$, therefore L/M is always small for deep networks when the bound holds. Then, a recent theoretical work improved this result in a special case of ReLU networks with initialization $(\sigma_w = 2, \sigma_b = 0)$ by showing the precise exponential dependence of the NTK variance at initialization on L/M (Hanin and Nica, 2019). That is, (4) does not hold for such networks when L/M is bounded away from zero. However, the proofs given in the paper are not immediately generalizable for different activation functions and different initialization parameters. Thus, there is still no solid understanding of the NTK randomness depending on the choice of a network. Therefore, in Section 3, we empirically study the randomness of the NTK at initialization for ReLU and `tanh` networks with a variety of hyperparameters $(M, L, \sigma_w, \sigma_b)$ and observe the precise dependence on 1) the position of initialization (σ_w, σ_b) in either ordered or chaotic phase, 2) depth-to-width ratio L/M in the chaotic phase.

Changes of the NTK matrix during gradient descent training have also been analyzed in the literature mostly as a function of M. In particular, it has been proven (Huang and Yau, 2020) and shown experimentally (Lee et al., 2019) that the change of the NTK matrix during gradient descent training is bounded by $O(1/M)$ when the depth L is fixed. For ReLU networks with initialization $(\sigma_w = 2, \sigma_b = 0)$ it has also been proven that the change of the NTK in a gradient descent step depends exponentially on L/M (Hanin and Nica, 2019). We add to these results in Section 4 by investigating the NTK changes during training for two activation functions and hyperparameters (σ_w, σ_b, L).

A different line of research has also studied the theoretical (infinite-width) NTK as a function of depth and initialization parameters (Xiao et al., 2019; Hayou et al., 2019). These contributions found that the spectrum of infinite-width NTK behaves differently in ordered and chaotic phases. The authors also showed that the infinite-*depth* limit of the theoretical NTK (when first the limit $M \to \infty$ is taken with fixed L and then $L \to \infty$) yields trivial performance and cannot explain properties of finite DNNs. These papers showed that both in ordered and chaotic phases the NTK approaches its trivial limit exponentially in L, and only in the border between phases (EOC) this convergence is sub-exponential. However, the setting of these contributions requires L/M values to be small, therefore they do not explain how the randomness of NTK and its changes during training impact the results. Our work shows that in the chaotic phase and at the EOC the NTK does not behave as its theoretical limit when L/M is bounded away from zero, therefore we cannot draw conclusions about such DNNs based on the theoretical NTK.

In generalization research, the recent trend is double descent – the phenomenon that highly overparametrized models, including DNNs, tend to generalize surprisingly well (Belkin et al., 2018; Nakkiran et al., 2019; Belkin et al., 2019; Hastie et al., 2019). The recent developments in the theory of double descent showed that overparametrized linear models reach low generalization error because, counterintuitively, their variance decreases when the number of parameters increases beyond the number of samples (Hastie et al., 2019). However, there is still no double descent theory for DNNs, which are significantly more theoretically complex than linear models. In Section 5, we studied the variance of DNNs' output with the simplifications of NTK theory, which can be seen as the first step into this direction.

2. Mean field approach for wide neural networks

A number of recent papers used the *mean field* formalism to study forward- and backpropagation of signal through randomly initialized DNNs (Poole et al., 2016; Schoenholz et al., 2016; Karakida et al., 2018; Yang and Schoenholz, 2017). We first describe this approach and show how ordered and chaotic phases, which correspond to vanishing and exploding gradients, arise from it.

Suppose there is a fully-connected feed-forward neural network initialized randomly as in (1) with hidden layers' widths $M_1, \dots M_L$. Forward propagation through the network is given by

$$\mathbf{x}^l(x_s) = \phi(\mathbf{h}^l(x_s)), \quad \mathbf{h}^l(x_s) = \mathbf{W}^l \mathbf{x}^{l-1}(x_s) + \mathbf{b}^l, \quad l = 1, \dots L,$$
$$\mathbf{x}^0(x_s) = x_s, \quad s = 1, \dots S,$$

where ϕ is the activation function, $\mathbf{x}^l$ are activations, $\mathbf{h}^l$ are pre-activations in each layer l, and $D = (X, Y) = \{(x_s, y_s)\}_{s=1,\dots S}$ is a dataset.

Consider variances $q^l(x_s) \coloneqq \mathbb{E}[(\mathbf{h}^l_i(x_s))^2]$ of the pre-activations in each layer for a given input vector x_s. The mean field theory approach assumes that $\mathbf{h}^l_i(x_s),\ i = 1, \dots M_l$ are i.i.d Gaussian, so by central limit theorem in the limit of $M \to \infty$, the variance can be seen as a sum over different neurons in the same layer $q^l(x_s) = \frac{1}{M_l} \sum_{i=1}^{M_l} (\mathbf{h}^l_i(x_s))^2$. Then it can be computed through a recursive relation:

$$q^l(x_s) = \sigma_w^2 \int Dz \cdot \phi(\sqrt{q^{l-1}(x_s)} z)^2 + \sigma_b^2, \tag{5}$$

where the average over numerous neurons in layer $l-1$ is replaced by an integral over a Gaussian distribution $Dz = \frac{dz}{\sqrt{2\pi}} e^{-z^2/2}$. Then the variance of activations $\hat{q}^l(x_s) \coloneqq \mathbb{E}[(\mathbf{x}^l_i(x_s))^2]$ is given by

$$\hat{q}^l(x_s) = \int Dz \cdot \phi(\sqrt{q^l(x_s)}z)^2. \tag{6}$$

In the same fashion, Poole et al. (2016) derive a recursive map for the correlation between pre-activations of two different inputs and the correlation between activations of two different inputs, denoted correspondigly $q^l(x_s, x_r) \coloneqq \mathbb{E}[\mathbf{h}_i^l(x_s)\mathbf{h}_i^l(x_r)]$ and $\hat{q}^l(x_s, x_r)] \coloneqq \mathbb{E}[\mathbf{x}_i^l(x_s)\mathbf{x}_i^l(x_r)]$:

$$\begin{aligned} q_{sr}^l(x_s, x_r) &= \sigma_w^2 \int Dz_1 Dz_2 \cdot \phi(u_1)\phi(u_2) + \sigma_b^2, \\ \hat{q}_{sr}^{l-1}(x_s, x_r) &= \int Dz_1 Dz_2 \cdot \phi(u_1)\phi(u_2), \\ u_1 = \sqrt{q^{l-1}(x_s)}z_1, \quad u_2 &= \sqrt{q^{l-1}(x_r)}[c_{sr}^{l-1}z_1 + \sqrt{1-(c_{sr}^{l-1})^2}z_2], \\ c_{sr}^{l-1} &= \frac{q^{l-1}(x_s, x_r)}{\sqrt{q^{l-1}(x_s)q^{l-1}(x_r)}}. \end{aligned} \tag{7}$$

The gradients of the network are given by the backpropagation chain:

$$\begin{aligned} \frac{\partial f}{\partial \mathbf{W}_{ij}^l} &= \delta_i^l \phi(\mathbf{h}_j^{l-1}), \quad \frac{\partial f}{\partial \mathbf{b}_i^l} = \delta_i^l, \\ \delta_i^l &= \frac{\partial f}{\partial \mathbf{h}_i^l} = \phi^{'}(\mathbf{h}_i^l) \sum_j \delta_j^{l+1} \mathbf{W}_{ji}^{l+1}, \end{aligned}$$

where we omitted the dependence on input x_s for simplicity. With an additional assumption that weights in forward- and backpropagation are drawn independently, i.e. $\phi(\mathbf{h}_j^l)$ and δ_i^l are independent, Schoenholz et al. (2016) derived a recursive relation for the variance of the backpropagated errors $p^l(x_s) \coloneqq \mathbb{E}[\sum_i (\delta_i^l(x_s))^2]$:

$$p^l(x_s) = \sigma_w^2 p^{l+1}(x_s) \frac{M_{l+1}}{M_{l+2}} \int Dz [\phi^{'}(\sqrt{q^l(x_s)}z)]^2. \tag{8}$$

And for the corresponding correlation between backpropagated errors of two different input vectors $p_{sr}^l(x_s, x_r) \coloneqq \mathbb{E}[\sum_i (\delta_i^l(x_s)\delta_i^l(x_r))]$:

$$\begin{aligned} p_{sr}^l(x_s, x_r) &= \sigma_w^2 p_{sr}^{l+1}(x_s, x_r) \frac{M_{l+1}}{M_{l+2}} \int Dz_1 Dz_2 \cdot \phi^{'}(u_1)\phi^{'}(u_2), \\ u_1 = \sqrt{q^l(x_s)}z_1, \quad u_2 &= \sqrt{q^l(x_r)}[c_{sr}^l z_1 + \sqrt{1-(c_{sr}^l)^2}z_2], \\ c_{sr}^l &= \frac{q_{sr}^l(x_s, x_r)}{\sqrt{q^l(x_s)q^l(x_r)}}. \end{aligned} \tag{9}$$

Note that for certain activation functions, e.g. ReLU and `erf`, the integrals in (5), (6), (7), (8) and (9) can be taken analytically. One can refer to Appendix E for these analytical expressions.

We can now introduce, following the notation from Poole et al. (2016) and Schoenholz et al. (2016), a quantity that controls the backpropagation of variance $p^l(x_s)$:

$$\begin{aligned} \chi_1^l &= \sigma_w^2 \int Dz [\phi^{'}(\sqrt{q^l}z)]^2, \\ p^l &= p^{l+1} \cdot \chi_1^l, \end{aligned}$$

where we assumed that the network's width is constant, i.e. $M_{l+1}/M_{l+2} = 1$. Then χ_1 also controls the propagation of the gradients at initialization:

$$\mathbb{E}[(\frac{\partial f^0(x_s)}{\partial \mathbf{W}^l_{ij}})^2] = \mathbb{E}[(\delta^l_i)^2]\mathbb{E}[(\phi(\mathbf{h}^{l-1}_j))^2] \propto p^l(x_s).$$

In particular, when the initialization parameters are such that $\chi^l_1 < 1$ in all the layers, the gradients vanish, and when $\chi^l_1 > 1$ the gradients explode. These two situations are referred to as *ordered* and *chaotic* phases correspondingly, and the border between these phases defined by $\chi^l_1 = 1$ is called *edge of chaos* (EOC) initialization. Several authors suggest that networks should be initialized near EOC to allow deeper signal propagation (Hayou et al., 2018; Schoenholz et al., 2016).

In the next two sections of the paper, we test empirically how different parameters of random initialization (σ^2_w, σ^2_b), as well as network's architecture (M, L), impact the behavior of the empirical NTK Θ^t. Our observation is that for finite-width networks chaotic and ordered phases give rise to very different behavior of the empirical NTK as compared to the theoretical NTK, which has not been considered in the community before to the best of our knowledge.

3. NTK variance at initialization

First we aim to verify empirically when the theoretical result (4) that the NTK is deterministic at initialization in the infinite-width limit holds for finite-width NNs. Following Hanin and Nica (2019), we computed the ratio $\mathbb{E}[\Theta^0(x,x)^2]/\mathbb{E}^2[\Theta^0(x,x)] \in [1, \infty)$ to study the distribution of the NTK. When the NTK at initialization is close to deterministic, its distribution is similar to a delta function around its mean and the value of the ratio is close to one. On the other hand, when this ratio is bounded away from one, the NTK's variance is comparable to its mean value and therefore cannot be disregarded.

One can see the results of our experiments for fully-connected ReLU and `tanh` networks with constant width M in Figure 1. We observe that when σ^2_w is small enough (ordered phase), the NTK variance is small and does not increase with depth L, implying that (4) holds for any depth and NTK theory can be used to study NNs initialized in this way. However, for large σ^2_w (chaotic phase) the variance grows significantly with L, hence for very deep networks in this phase (4) does not hold. At the EOC, the variance of the NTK is a fraction of its mean even for very deep networks, so NTK theory can approximate the average behavior of networks initialized near EOC, but the random effects may still be significant. One can also see that as M grows, the vertical red region gets narrower, i.e. the transition becomes sharper. This is consistent with the fact that the theoretical border between vanishing and exploding gradients is sharp and computed in mean field theory (Section 2) by taking the limit $M \to \infty$. These results are similar for ReLU and `tanh` networks, taking into account that the theoretical boundary between phases — given by $\chi^l_1 = 1$ and indicated by the dashed line in the figures — is located at larger σ^2_w values for sigmoid networks. One also observes that the NTK variance is small for sufficiently shallow NNs with any σ^2_w value. Such shallow networks were mostly considered in recent empirical studies on behavior of wide NNs under gradient descent (Lee et al., 2019). It is thus important to note, that such empirical results may be invalid for much deeper networks, depending on the initialization parameters.

Moreover, when depth L is fixed and width M increases, the the NTK variance decreases in the chaotic phase, which supports the hypothesis that the variance depends on the ratio L/M. To examine this dependence on L/M in more detail, we present Figure 2. It shows the ratio

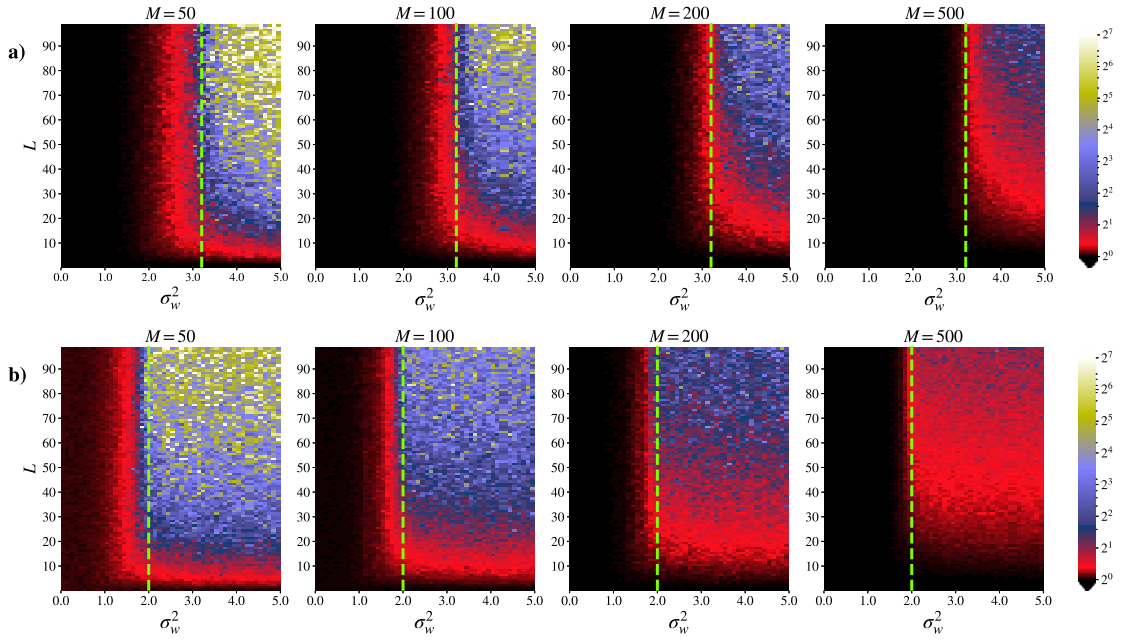


Figure 1: Ratio $\frac{\mathbb{E}[\Theta^0(x,x)^2]}{\mathbb{E}^2[\Theta^0(x,x)]}$ for fully-connected a) `tanh`, b) ReLU networks of constant widths $M = 50, 100, 200, 500$, in all the experiments $\sigma_b^2 = 1$. The expected values for each set of parameters are calculated by sampling 200 random initializations of the network. The NTK is computed using TensorFlow automatic differentiation. The dashed line shows the theoretical border between ordered and chaotic phases ($\chi_1^l = 1$) for the given hyperparameters. In the black zone, the ratio is close to one, i.e. the NTK at initialization Θ^0 has low variance and can be considered a deterministic variable. In the red zone, the NTK standard deviation is comparable with its mean. In the blue zone, the NTK standard deviation is greater than its mean, so the NTK is not deterministic and cannot be replaced by its mean.

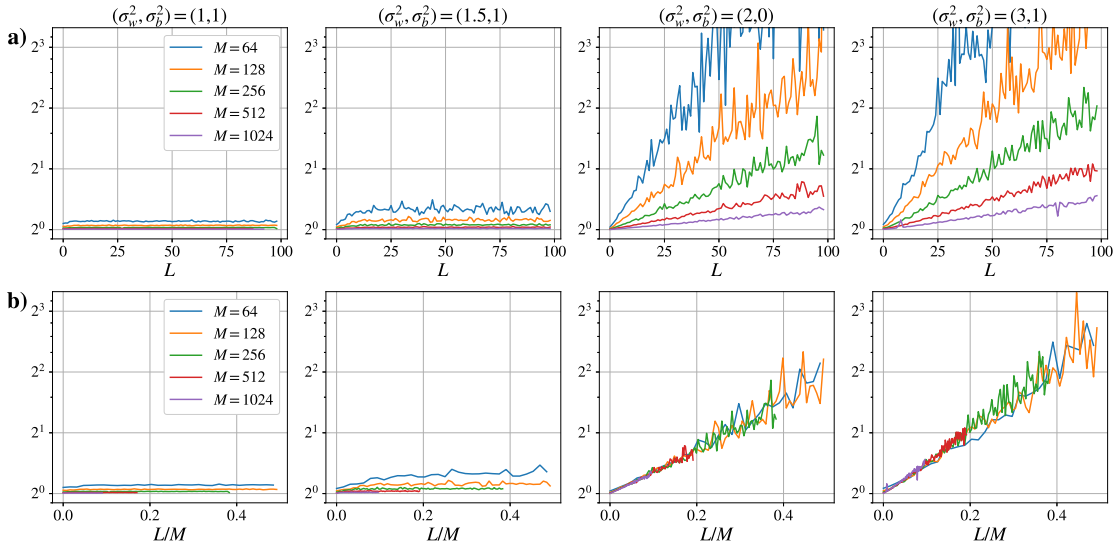


Figure 2: Dependence of ratio $\dfrac{\mathbb{E}[\Theta^0(x,x)^2]}{\mathbb{E}^2[\Theta^0(x,x)]}$ on L/M with different initialization parameters and width values for ReLU networks. Both rows show the same curves plotted against a) depth L, b) ratio L/M. The expectations are computed by sampling 200 random initializations of the network.

$\mathbb{E}[\Theta^0(x,x)^2]/\mathbb{E}^2[\Theta^0(x,x)]$ for a wider range of M values for four different initialization parameters sets: $(\sigma_w^2, \sigma_b^2) \in [(1,1),(1.5,1),(2,0),(3,1)]$. Each curve is plotted against both L and L/M. We notice that in the ordered phase ($\sigma_w^2 = 1$ and $\sigma_w^2 = 1.5$) the ratio is close to 1, does not grow with L/M and decreases with M. In this phase, the NTK converges to its deterministic limit with increasing M regardless of the L value, which is the expected behaviour within NTK theory. However, in the chaotic phase ($\sigma_w^2 = 3$) the ratio grows exponentially as a function of L/M. This observation gives a precise scaling for minimal M values required to assume that the NTK of a network with a given depth L is deterministic at initialization, which improves the previous asymptotic result in Jacot et al. (2018) and the bound on required M in Arora et al. (2019). In case of ReLU networks and initialization $(\sigma_w^2, \sigma_b^2) = (2,0)$, Hanin and Nica (2019) theoretically showed that the $\mathbb{E}[\Theta^0(x,x)^2]/\mathbb{E}^2[\Theta^0(x,x)]$ ratio is indeed exponential in L/M, but their analysis is not trivially generalizable for different activation functions and initialization parameters. Our experiments confirm these findings in the special case but also show that changing initialization parameters impacts the behaviour of the the NTK variance significantly.

We also checked if the value of σ_b^2 impacts the NTK variance behavior at initialization significantly. In Appendix D, we provide figures showing the NTK variance with different σ_b^2 values. We observed that lower σ_b^2 values yield narrower boundary between the two phases identified in Figure 1, but the general picture stays similar.

4. NTK change during training

In this section we present the numerical experiments that we conducted to check whether the second result of NTK theory (3) holds, i.e. whether the empirical NTK of finite-width ReLU and `tanh` networks stays approximately constant during training with gradient descent. We trained networks with a variety of hyperparameters $(\sigma_w^2, \sigma_b^2, L)$ and measured the relative change of NTK's Frobenious norm $\|\Theta^t - \Theta^0\|_F / \|\Theta^0\|_F$ that occurs during training. The results for `tanh` and ReLU networks are in Figures 3a and 4a. In Figures 3b and 4b, we also plotted the minimal losses that the networks reached in the experiments.

We draw the following conclusions from the experiments' results:

- **Phase transition for empirical NTK**. For both ReLU and `tanh` networks, the NTK behavior during training changes significantly around the theoretical border between chaotic and ordered phases.

- **Chaotic phase**. In the chaotic phase, the relative change in the NTK matrix norm is significant and increases with depth L, so one cannot assume that the kernel stays constant during training for deep networks. However, for very shallow networks the NTK at initialization may still be a good approximation for the NTK after training. In the previous section we also saw that the NTK matrix of shallow networks in the chaotic phase is close to deterministic at initialization, which shows that NTK theory approximates only shallow networks in the chaotic phase.

- **Ordered phase**. In the ordered phase, the relative change in the NTK matrix norm is small throughout training for any depth. We saw in the previous section that the NTK is also close to deterministic at initialization in this phase. It follows that in the ordered phase finite-width DNNs behave as NTK theory suggests even when depth L is large.

- **EOC**. There is a region close to the border between phases where the change in the NTK norm is larger than in the ordered phase but still remains way below 1 for deep networks. We also saw in the previous section that in this region the standard deviation of the NTK is lower than its mean value for deep networks. Thus, NTK theory can approximate behavior of deeper networks in case of EOC initialization in comparison to the chaotic phase, but the effects of randomness and change during training may still play a significant role.

- **Trainability**. Networks become untrainable with depth much faster in the ordered phase than in the chaotic phase. In our experiments, networks in the ordered phase with $L = 20$ already mostly cannot reach low training loss values. This is consistent with the results on trainability provided in Xiao et al. (2019).

We thus have discovered two regions in the hyperparameters space $(\sigma_w^2, \sigma_b^2, L, M)$ where both statements of NTK theory (3) and (4) hold: the ordered phase with any depth L and the chaotic phase where the L/M ratio is low. For other choices of architecture and initialization, our experiments suggest that finite-width networks do not behave according to NTK theory.

Note that the networks in Figures 3a and 4a take different number of training steps to reach their final loss values. Somewhat counterintuitively, we observe that the networks which take more iterations to train show mostly small changes in the NTK matrix norm. To provide more insight about the NTK dynamics during different stages of training, we also include figures that show

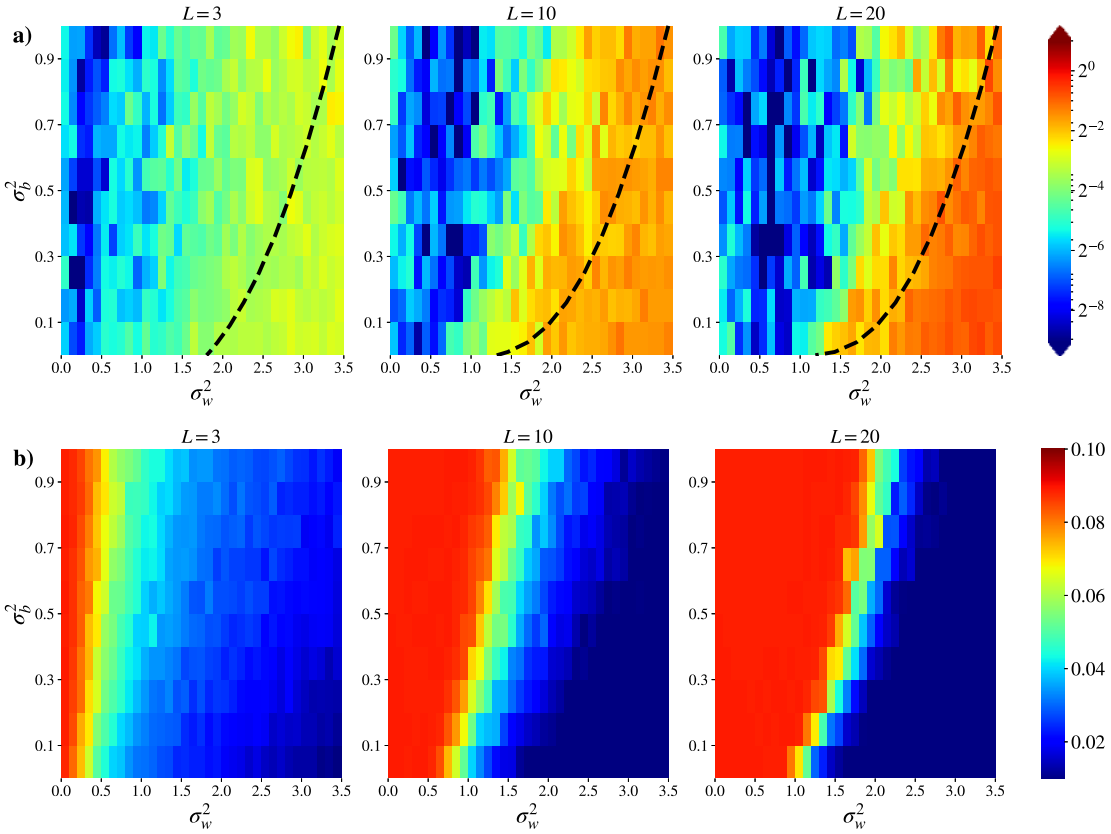


Figure 3: a) Relative change in the NTK norm $\frac{\|\Theta^t - \Theta^0\|_F}{\|\Theta^0\|_F}$ for `tanh` networks of width $M = 256$ trained by gradient descent with MSE loss on a subset of MNIST (128 samples). The dashed line indicates the theoretical border between ordered and chaotic phases ($\chi_1^l = 1$). We used early stopping when the loss did not decrease by at least 10^{-7} in 100 consecutive steps, otherwise the number of training steps was limited by 10^5. The learning rate is constant and equals 10^{-5} for all the networks, which is chosen so that, for all the hyperparameters, it does not exceed the theoretical maximal learning rate for wide networks derived in Karakida et al. (2018). b) Minimal loss value that the networks managed to reach in our experiments. Networks in the red area are untrainable with the given learning rate, networks in the blue area are trainable.

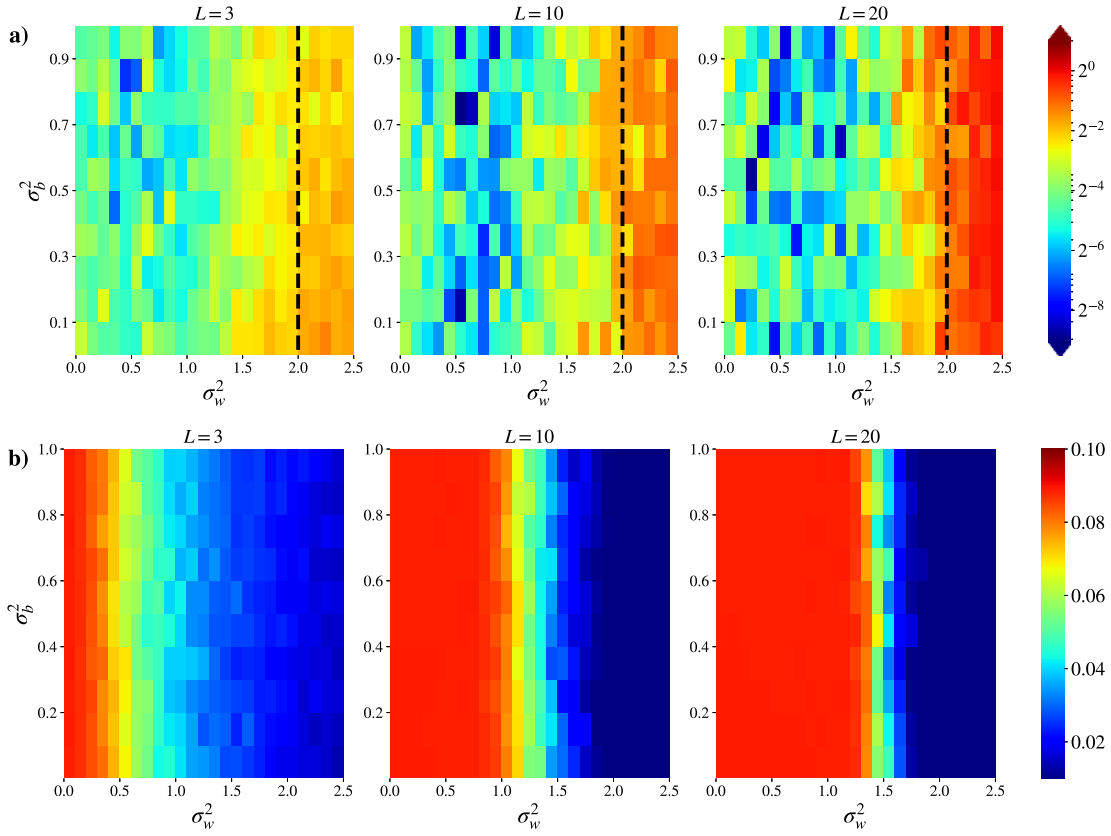


Figure 4: a) Relative change in the NTK norm $\frac{\|\Theta^t - \Theta^0\|_F}{\|\Theta^0\|_F}$ for ReLU networks of width $M = 256$ trained by gradient descent with MSE loss on a subset of MNIST (128 samples). The dashed line indicates the theoretical border between ordered and chaotic phases ($\chi_1^l = 1$). We used early stopping when the loss did not decrease by at least 10^{-7} in 100 consecutive steps, otherwise the number of training steps was limited by 10^5. The learning rate is constant and equals 10^{-5} for all the networks, which is chosen so that, for all the hyperparameters, it does not exceed the theoretical maximal learning rate for wide networks derived in Karakida et al. (2018). b) Minimal loss value that the networks managed to reach in our experiments. Networks in the red area are untrainable with the given learning rate, networks in the blue area are trainable.

changes in the NTK matrix norm as a function of the number of training steps, as well as figures with changes of the NTK for different M values, in Appendix D.

5. NTK theory approach for generalization

If the NTK stays constant during training (3), then the dynamics in (2) are identical to kernel regression with kernel Θ^0. In such dynamics, the output function of a network that is trained until convergence ($t \to \infty$) by gradient flow with MSE loss is given by:

$$f^{t=\infty}(x) = \Theta^0(x, X)\Theta^0(X)^{-1}Y + f^0(x) - \Theta^0(x, X)\Theta^0(X)^{-1}f^0(X), \tag{10}$$

where $\Theta^0(X)$ is the kernel matrix of all the pairs of inputs in $X = [x_s]_{s=1,...S}$, i.e. $\Theta(X) = [\Theta^0(x_s, x_r)]_{s,r=1,...S}$, and $\Theta(x, X) = [\Theta^0(x, x_s)]_{s=1,...S}$ and $f^0(X) = [f^0(x_s)]^T_{s=1,...S}$. One can refer to Arora et al. (2019) or Lee et al. (2019) for the derivation of this equation. If the NTK is also deterministic at initialization (4), then the only variables in (10) that are random with respect to the network's parameters at initialization w_0 are $f^0(x)$ and $f^0(X)$, which greatly simplifies the analysis of the generalization properties of $f^{t=\infty}$.

Let us denote $R(x) := \mathbb{E}_{w_0,D}[(f^{t=\infty}(x) - y_{true})^2]$ – the expected error on an arbitrary test point x, given that the initialization is random. Then we can write the bias-variance decomposition as follows:

$$R(x) = Var(f^{t=\infty}(x)) + Bias(f^{t=\infty}(x)),$$

where

$$Var(f^{t=\infty}(x)) = \mathbb{E}_{w_0,D}[(f^{t=\infty}(x) - \mathbb{E}_{w_0,D}[f^{t=\infty}(x)])^2],$$
$$Bias(f^{t=\infty}(x)) = \mathbb{E}_{w_0,D}[(\mathbb{E}_{w_0,D}[f^{t=\infty}(x)] - y_{true})^2].$$

Then NTK theory allows us to analyze the variance term to characterize the generalization error of the network $\mathbb{E}_x[R(x)]$. To do so, first let us show how distributions of the terms in (10) can be characterized by the mean field theory quantities introduced in Section 2. First of all, the distribution of the network's output at initialization is given directly by the definitions of q^L and q^L_{sr}. Hence, the following lemma is immediate.

Lemma 1 *The variance of the output function f^0 of a randomly initialized network and the covariance of outputs on two different input vectors are given by:*

$$\mathbb{E}[(f^0(x))^2] = \mathbb{E}[(\mathbf{h}^L_i(x))^2] = q^L(x),$$

$$\mathbb{E}[f^0(x_s)f^0(x_r)] = \mathbb{E}[\mathbf{h}^L_i(x_s)\mathbf{h}^L_i(x_r)] = q^L_{sr}(x_s, x_r).$$

Recall that the NTK is composed of gradients as $\Theta^0(x_s, x_r) = \nabla_w f^0(x_s)^T \nabla_w f^0(x_s)$ and its expected values are therefore proportional to the variances of gradients, considered in Section 2. Then, assuming that the the NTK matrix at initialization is deterministic and equal to its expected value, we can express it through quantities $q^l, p^l, q^l_{sr}, p^l_{sr}$ by the following lemma.

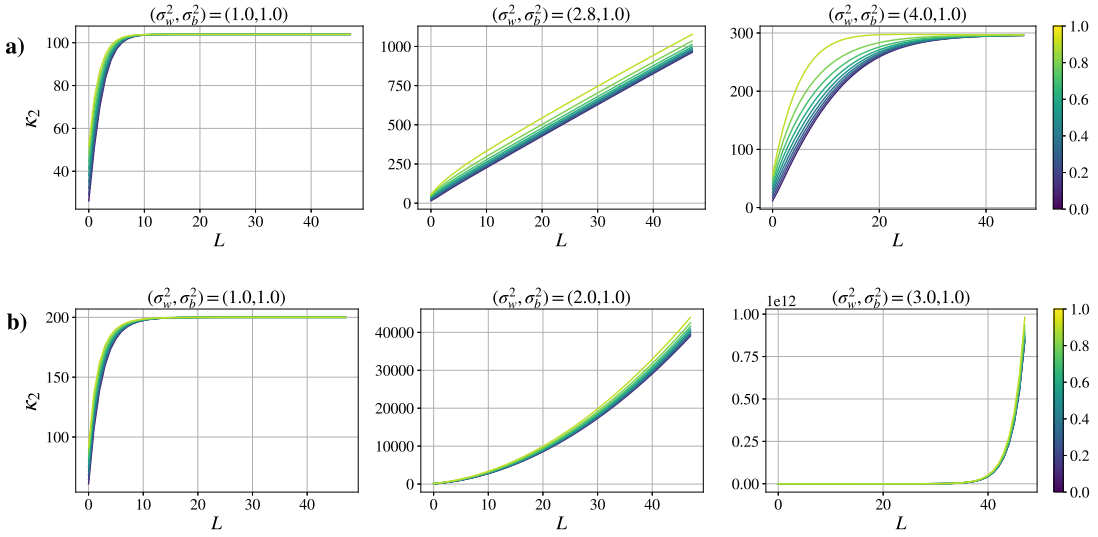


Figure 5: κ_2 as a function of depth for a) `erf`, b) ReLU networks. The colorbar shows the initial value of the covariance between inputs $x_s^T x_r \in [0, 1]$. For both activation functions, (σ_w^2, σ_b^2) values are chosen to lie in ordered and chaotic phases and at the border between them.

Lemma 2 *For a fully-connected network with widths $M_l = \alpha_l M, l = 0, \dots L$ (where M_0 is the input dimension), deterministic the NTK matrix on a sample $X = \{x_s\}_{s=1,\dots S}$ at initialization is given by:*

$$\Theta^*(X) = \alpha M\big(\Lambda + O(1/M)\big),$$

$$\Lambda = \begin{bmatrix} \kappa_1(x_1) & \kappa_2(x_1, x_2) & \dots & \kappa_2(x_1, x_S) \\ \kappa_2(x_1, x_2) & \kappa_1(x_2) & & \dots \\ \dots & & & \kappa_2(x_1, x_{S-1}) \\ \kappa_2(x_1, x_S) & \dots & \kappa_2(x_1, x_{S-1}) & \kappa_1(x_S) \end{bmatrix},$$

$$\kappa_1(x) = \sum_{l=1}^{L} \frac{\alpha_{l-1}}{\alpha} \dot{q}^{l-1}(x) p^l(x), \quad \kappa_2(x_s, x_r) = \sum_{l=1}^{L} \frac{\alpha_{l-1}}{\alpha} \dot{q}_{sr}^{l-1}(x_s, x_r) p_{sr}^l(x_s, x_r),$$

where $\alpha = \sum_{l=1}^{L-1} \alpha_l \alpha_{l-1}$.

We give a proof for this lemma in Appendix A. We note that the same statement is also proven in Karakida et al. (2018) as a part of Theorem 3.

We can also notice that κ_1 and q^l depend only on the norm of input x, so for normalized inputs they become data-independent. On the other hand, κ_2 and q_{sr}^l depend on covariances of points in the dataset and therefore are data-dependent. However, it has also been observed in Poole et al. (2016) that both q^l and q_{sr}^l converge to their data-independent limits with depth. Let us denote their

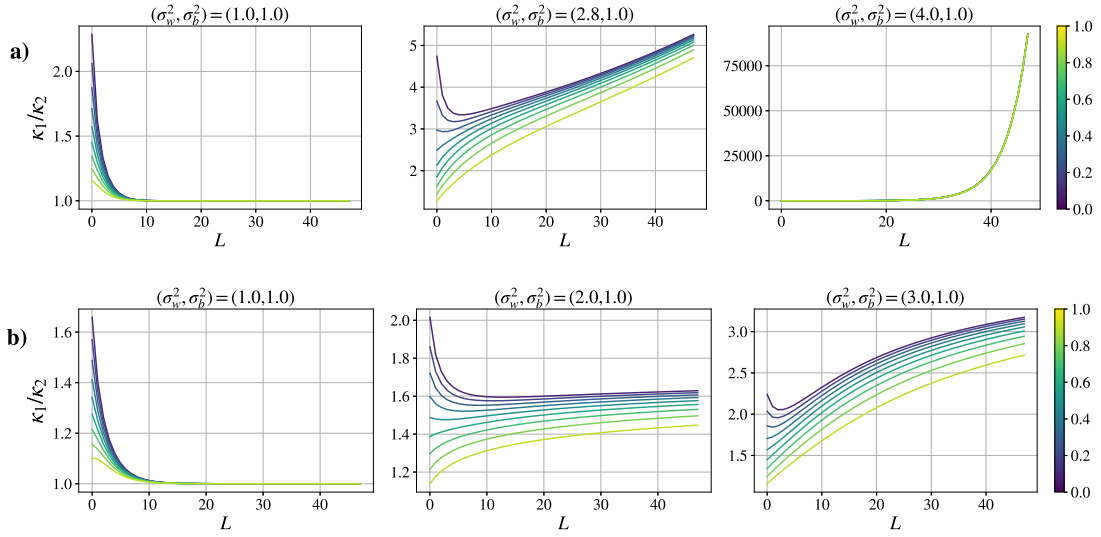


Figure 6: κ_1/κ_2 ration as a function of depth for a) `erf`, b) ReLU networks. The colorbar shows the initial value of the covariance between inputs $x_s^T x_r \in [0, 1]$. For both activation functions, (σ_w^2, σ_b^2) values are chosen to lie in ordered and chaotic phases and at the border between them.

data-independent means by $\bar{q}^l$ and $\bar{q}^l_{sr}$ respectively. Then we can also write data-independent means $\bar{p}^l$ and $\bar{p}^l_{sr}$ for the backpropagated errors, as well as $\hat{q}^l$ and $\hat{q}^l_{sr}$ for the activations. This leads to data-independent $\bar{\kappa}_1 = \sum_{l=1}^{L} \frac{\alpha_{l-1}}{\alpha} \hat{q}^{l-1} \bar{p}^l$ and $\bar{\kappa}_2 = \sum_{l=1}^{L} \frac{\alpha_{l-1}}{\alpha} \hat{q}^{l-1}_{sr} \bar{p}^l_{sr}$. We also notice that the changes in κ_2 that come from the changes in covariance are small with respect to its mean value $\bar{\kappa}_2$ for ReLU and `erf` networks[1]. Note that for these two activation functions, we can take the integrals in (5), (7), (8) and (9) analytically (see Appendix E) and calculate κ_2 for different values of the inputs' covariance, which is shown in Figure 5 for ordered and chaotic phases and at the border between them. Therefore, we can write the NTK as a sum of its data-independent part and a data-dependent perturbation:

$$\Theta^*(X) = \bar{\Theta}^*(\mathbf{I}_S + \epsilon(X)),$$
$$\bar{\Theta}^* = \alpha M\big((\bar{\kappa}_1 - \bar{\kappa}_2)\mathbf{I}_S + \bar{\kappa}_2 \mathbb{1}_S \mathbb{1}_S^T\big).$$

We note that this result about the structure of the NTK is consistent with the analysis of Xiao et al. (2019), where the authors study the NTK at large depths.

From the structure of Θ^*, one can see that its condition number depends on the ratio κ_1/κ_2: when its value is high, the NTK matrix is well-conditioned, and when the ratio approaches 1 the matrix becomes close to degenerate. Figure 6 shows κ_1/κ_2 ratio as a function of depth for `erf` and ReLU networks in ordered and chaotic phases and at the border between them. One can see

1. We expect `tanh`-networks that we studied empirically in other sections to behave similar to `erf`-networks.

from the graphs that the NTK matrix is well-conditioned in the chaotic phase and ill-conditioned in the ordered phase. Ill-conditioned NTK also implies that the maximum learning rate which allows to train the network is small (Xiao et al., 2019; Karakida et al., 2018). Therefore networks in the ordered phase rapidly become untrainable with depth, which is consistent with our observations in Section 4.

The following theorem characterizes the dependence of the variance of the output function $f^{t=\infty}(x)$ on the data-independent part of the NTK.

Theorem 3 *Suppose a network evolves according to NTK theory under gradient flow and is fully trained ($t \to \infty$) on a dataset of size S. Suppose also that the NTK matrix is well-conditioned. Then the variance of its output is characterized by:*

$$Var(f^{t=\infty}(x)) \approx (1 + \frac{A^2}{S})(\bar{q}^L - \bar{q}^L_{sr}) + (A-1)^2 \bar{q}^L_{sr},$$

where $A = A(\kappa_1, \kappa_2) = \frac{S}{\bar{\kappa}_1/\bar{\kappa}_2 + (S-1)}$.

We give a proof for this result in the Appendix B. In the next paragraphs, we analyze the behavior of the given variance expression and the applicability of the theorem in different situations:

- **Ordered phase.** One can notice that in the ordered phase $A(\kappa_1, \kappa_2)$ converges to 1 rapidly with depth, as $\bar{\kappa}_1/\bar{\kappa}_2 \to 1$. This implies $Var(f^{t=\infty}(x)) \propto \bar{q}^L - \bar{q}^L_{sr}$, i.e. the variance is small and decreases with depth. However, the NTK is also ill-conditioned, therefore small data-dependent changes can cause significant changes in the output function. Thus, the data-independent estimate for variance given by NTK theory does not explain the behavior of DNNs in the ordered phase and it is important to take into account data-dependent effects.

- **Chaotic phase.** In the chaotic phase, the NTK is well-conditioned for any depth. However, only networks with depth to width ratio $L/M \approx 0$ behave as NTK theory suggests under gradient flow in the chaotic phase according to our experiments. As we saw in the previous sections, the NTK changes significantly during training and is random at initialization for deep networks, therefore the expression for the output function after training (10) does not hold. The ratio $\bar{\kappa}_1/\bar{\kappa}_2$ increases with depth in the chaotic phase, so $A(\kappa_1, \kappa_2)$ decreases, and $\bar{q}^L$ is much larger than $\bar{q}^L_{st}$ (Poole et al., 2016). Therefore the data-independent variance $Var(f^{t=\infty}(x)) \propto \bar{q}^L$ is high and proportional to the variance of outputs of a randomly initialized network. This is consistent with observations in Chizat et al. (2019) and Xiao et al. (2019). Thus, NTK theory can explain poor generalization, which shallow wide networks in the chaotic phase display. However, deeper networks may have very different behavior due to randomness at initialization and changes during gradient descent training, so they require more investigation.

- **EOC.** At EOC, the conditioning of the NTK as a function of depth is similar to the chaotic phase: $\bar{\kappa}_1/\bar{\kappa}_2$ grows with depth, hence the kernel is well-conditioned. However, at EOC $\bar{q}^L$ is smaller than in the chaotic phase (Poole et al., 2016). This implies that networks initialized close to EOC generalize better than networks in the chaotic phase and at the same time remain trainable at large depths. We observed in the previous sections that at the border between phases NTK theory gives an approximation of network's average behavior even for deep networks, but the finite-width effects can still be significant and should be considered.

6. Conclusions and future work

In this work, we have shown that NTK theory does not generally describe the training dynamics of finite-width DNNs accurately. Only relatively shallow networks and deep networks in the ordered phase, i.e. initialized with small σ_w^2, behave as NTK theory suggests under gradient descent. The analysis of the data-independent variance of the output function based on NTK theory shows that it is proportional to the output variance at initialization q^L in the chaotic phase and at EOC. This result is not surprising, in a sense that it does not explain how training effects NNs' performance. It would provide more insight into networks' behavior if we could understand the data-dependent changes in the NTK, which are significant for deep networks in the ordered phase and at EOC, and study how these changes impact the output function. To study deep networks in the chaotic phase and at EOC, it is also essential to account for randomness in the NTK matrix at initialization and its changes during training, which cannot be done within NTK theory. Thus, an entirely new conceptual viewpoint is required to provide a full theoretical analysis of DNNs behavior under gradient descent.

Acknowledgments

GK would like to acknowledge partial support by the NSF-Simons Research Collaboration THEORINET.

References

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pages 8139–8148, 2019.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine learning and the bias-variance trade-off. *arXiv preprint arXiv:1812.11118*, 2018.

Mikhail Belkin, Daniel Hsu, and Ji Xu. Two models of double descent for weak features. *arXiv preprint arXiv:1903.07571*, 2019.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, pages 2937–2947, 2019.

Mario Geiger, Arthur Jacot, Stefano Spigler, Franck Gabriel, Levent Sagun, Stéphane d'Ascoli, Giulio Biroli, Clément Hongler, and Matthieu Wyart. Scaling description of generalization with number of parameters in deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(2):023401, 2020.

Ian J Goodfellow, Oriol Vinyals, and Andrew M Saxe. Qualitatively characterizing neural network optimization problems. *arXiv preprint arXiv:1412.6544*, 2014.

Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel. *arXiv preprint arXiv:1909.05989*, 2019.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234. PMLR, 2016.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019.

Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. On the selection of initialization and activation function for deep neural networks. *arXiv preprint arXiv:1805.08266*, 2018.

Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. Mean-field behaviour of neural tangent kernel for deep neural networks. *arXiv preprint arXiv:1905.13654*, 2019.

Jiaoyang Huang and Horng-Tzer Yau. Dynamics of deep neural networks and neural tangent hierarchy. In *International Conference on Machine Learning*, pages 4542–4551. PMLR, 2020.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. Universal statistics of fisher information in deep neural networks: Mean field approach. *arXiv preprint arXiv:1806.01316*, 2018.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *arXiv preprint arXiv:1711.00165*, 2017.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in neural information processing systems*, pages 8572–8583, 2019.

Alexander G de G Matthews, Mark Rowland, Jiri Hron, Richard E Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pages 2924–2932, 2014.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *arXiv preprint arXiv:1912.02292*, 2019.

Roman Novak, Lechao Xiao, Jaehoon Lee, Yasaman Bahri, Greg Yang, Jiri Hron, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. *arXiv preprint arXiv:1810.05148*, 2018.

Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances in neural information processing systems*, pages 3360–3368, 2016.

Samuel S Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. *arXiv preprint arXiv:1611.01232*, 2016.

Lechao Xiao, Jeffrey Pennington, and Samuel S Schoenholz. Disentangling trainability and generalization in deep learning. *arXiv preprint arXiv:1912.13053*, 2019.

Ge Yang and Samuel Schoenholz. Mean field residual networks: On the edge of chaos. In *Advances in neural information processing systems*, pages 7103–7114, 2017.

Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*, 2020.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Appendix A. Lemma 2

By definition, each component of the NTK matrix is a scalar product of network's gradient vectors:

$$\Theta^0(X) = [\nabla_w f^0(x_s)^T \nabla_w f^0(x_r)]_{x_s \in X, x_r \in X}.$$

In Section 2 we show for the network's gradients that

$$\mathbb{E}\Big[(\frac{\partial f^0(x)}{\partial \mathbf{W}_{ij}^l})^2\Big] = \mathbb{E}[(\delta_i^l)^2]\mathbb{E}[(\phi(\mathbf{h}_j^{l-1}))^2] = \frac{1}{M_l} p^l(x)\hat{q}^{l-1}(x),$$
$$\mathbb{E}\Big[(\frac{\partial f^0(x)}{\partial \mathbf{b}_i^l})^2\Big] = \mathbb{E}[(\delta_i^l)^2] = \frac{1}{M_l} p^l(x),$$

and similarly

$$\begin{aligned}\mathbb{E}\Big[\frac{\partial f^0(x_s)}{\partial \mathbf{W}_{ij}^l}\frac{\partial f^0(x_r)}{\partial \mathbf{W}_{ij}^l}\Big] &= \mathbb{E}[\delta_i^l(x_s)\delta_i^l(x_r)]\mathbb{E}[\phi(\mathbf{h}_j^{l-1})(x_s)\phi(\mathbf{h}_j^{l-1})(x_r)] \\ &= \frac{1}{M_l} p_{sr}^l(x_s, x_r)\hat{q}_{sr}^{l-1}(x_s, x_r), \\ \mathbb{E}\Big[\frac{\partial f^0(x_s)}{\partial \mathbf{b}_i^l}\frac{\partial f^0(x_r)}{\partial \mathbf{b}_i^l}\Big] &= \mathbb{E}[\delta_i^l(x_s)\delta_i^l(x_r)] = \frac{1}{M_l} p_{sr}^l(x_s, x_r).\end{aligned}$$

Thus, we get the following expression for non-diagonal elements of the NTK:

$$\begin{aligned}
\Theta^0(x_s, x_r) &= \sum_{i,j,l} \Big[\frac{\partial f^0(x_s)}{\partial \mathbf{W}^l_{ij}} \frac{\partial f^0(x_r)}{\partial \mathbf{W}^l_{ij}} \Big] + \sum_{i,l} \Big[\frac{\partial f^0(x_s)}{\partial \mathbf{b}^l_{i}} \frac{\partial f^0(x_r)}{\partial \mathbf{b}^l_{i}} \Big] \\
&= \sum_l M_l M_{l-1} \mathbb{E}[\delta^l_i(x_s)\delta^l_i(x_r)] \mathbb{E}[\phi(\mathbf{h}^{l-1}_j)(x_s)\phi(\mathbf{h}^{l-1}_j)(x_r)] \\
&\quad + \sum_l M_l \mathbb{E}[\delta^l_i(x_s)\delta^l_i(x_r)] \\
&= \sum_l \alpha_{l-1} M p^l_{sr}(x_s, x_r) q^{l-1}_{sr}(x_s, x_r) + \sum_l p^l_{sr}(x_s, x_r) \\
&= \alpha M \Big(\sum_l \frac{\alpha_{l-1}}{\alpha} p^l_{sr}(x_s, x_r) q^{l-1}_{sr}(x_s, x_r) + O(1/M) \Big) \\
&= \alpha M (\kappa_2(x_s, x_r) + O(1/M))
\end{aligned}$$

Similarly, we get the expression for diagonal elements of the NTK matrix:

$$\Theta^0(x, x) = \alpha M (\kappa_1(x) + O(1/M)),$$

which gives the statement of the lemma.

Appendix B. Theorem 3

Recall the formula of the output function after training:

$$f^{t=\infty}(x) = \Theta^0(x, X)\Theta^0(X)^{-1} Y + f^0(x) - \Theta^0(x, X)\Theta^0(X)^{-1} f^0(X).$$

As initialization of the network's parameters w_0 is centered Gaussian, the expectation of the output at initialization is equal to zero:

$$\mathbb{E}_{w_0}[f^0(x)] = 0, \quad \mathbb{E}_{w_0}[f^0(X)] = \mathbf{0}_S.$$

Then if the NTK is deterministic at initialization we can write the expectation as follows:

$$\mathbb{E}_{w_0}[f^{t=\infty}(x)] = \mathbb{E}_{w_0}[\Theta^0(x, X)\Theta^0(X)^{-1} Y] = \Theta^*(x, X)\Theta^*(X)^{-1} Y$$

because neither Y nor Θ^* are random with respect to the initialization parameters.

To obtain the variance of output, we also need to write the expected values of all the terms of squared $f^{t=\infty}$. First, by Lemma 1:

$$\mathbb{E}_{w_0}[(f^0(x))^2] = q^L(x).$$

Then,

$$\mathbb{E}_{w_0}[(\Theta^0(x, X)\Theta^0(X)^{-1} Y)^2] = (\Theta^*(x, X)\Theta^*(X)^{-1} Y)^2 = \mathbb{E}^2_{w_0}[f^{t=\infty}(x)].$$

And

$$\begin{aligned}\mathbb{E}_{w_0}[(\Theta^0(x,X)\Theta^0(X)^{-1}f^0(X))^2] \\ &= tr(\mathbb{E}_{w_0}[f^0(X)f^0(X)^T]\Theta^*(X)^{-1}\Theta^*(x,X)^T\Theta^*(x,X)\Theta^*(X)^{-1}) \\ &= tr(K(X)\Theta^*(X)^{-1}\Theta^*(x,X)^T\Theta^*(x,X)\Theta^*(X)^{-1}),\end{aligned}$$

where

$$K(X) = \begin{bmatrix} q^L(x_1) & q^L_{sr}(x_1,x_2) & \dots & q^L_{sr}(x_1,x_S) \\ q^L_{sr}(x_1,x_2) & q^L(x_2) & & \dots \\ \dots & & & q^L_{sr}(x_1,x_{S-1}) \\ q^L_{sr}(x_1,x_S) & \dots & q^L_{sr}(x_1,x_{S-1}) & q^L(x_S) \end{bmatrix}.$$

$K(X)$ is the NNGP matrix, which characterizes the Gaussian process of a randomly initialized network. Finally:

$$\begin{aligned}\mathbb{E}_{w_0}[f^0(x)\Theta^0(x,X)\Theta^0(X)^{-1}f^0(X)] &= \Theta^*(x,X)\Theta^*(X)^{-1}\mathbb{E}_{w_0}[f^0(x)f^0(X)] \\ &= \Theta^*(x,X)\Theta^*(X)^{-1}q^L_{sr}(x,X),\end{aligned}$$

where $q^L_{sr}(x,X) = [q^L_{sr}(x,x_s)]^T_{s=1,\dots S}$. The other terms are equal to zero. Moreover, we can see that terms of variance with Y cancel each other.

We now recall that $\Theta^*(X) = \bar{\Theta}^*(\mathbf{I}_S + \epsilon(X))$ and $\bar{\Theta}^* = \alpha M\big((\bar{\kappa}_1 - \bar{\kappa}_2)\mathbf{I}_S + \bar{\kappa_2}\mathbb{1}_S\mathbb{1}_S^T\big)$. Then we can invert $\bar{\Theta}^*$ by Woodbury identity:

$$\bar{\Theta}^{*\,-1} = \frac{1}{\alpha M(\bar{\kappa}_1 - \bar{\kappa}_2)}\big(\mathbf{I}_S - \frac{\bar{\kappa}_2}{\bar{\kappa}_1 + (S-1)\bar{\kappa}_2}\mathbb{1}_S\mathbb{1}_S^T\big)$$

We assumed that the NTK matrix is well-conditioned, so the change in the $\bar{\Theta}^{*\,-1}$ caused by the perturbation term is relatively small and we can write $\Theta^{*\,-1}(X) = \bar{\Theta}^{*\,-1}(\mathbf{I}_S + \tilde{\epsilon}(X))$. Then we can also approximate the above expectation as follows:

$$\begin{aligned}\Theta^*(x,X)\Theta^*(X)^{-1}q^L_{sr}(x,X) &\approx \frac{\bar{\kappa_2}}{(\bar{\kappa}_1 - \bar{\kappa}_2)}\mathbb{1}_S^T\big(\mathbf{I}_S - \frac{\bar{\kappa}_2}{\bar{\kappa}_1 + (S-1)\bar{\kappa}_2}\mathbb{1}_S\mathbb{1}_S^T\big)q^L_{sr}(x,X) \\ &= \frac{\bar{\kappa_2}}{(\bar{\kappa}_1 - \bar{\kappa}_2)}\big(1 - \frac{\bar{\kappa}_2 S}{\bar{\kappa}_1 + (S-1)\bar{\kappa}_2}\big)\mathbb{1}_S^T q^L_{sr}(x,X) \\ &= \frac{S}{(\bar{\kappa}_1/\bar{\kappa}_2 + (S-1))}\langle q^L_{sr}(x_s,x)\rangle_{s=1,\dots S},\end{aligned}$$

$$\begin{aligned}tr(K(X)\Theta^*(X)^{-1}\Theta^*(x,X)^T\Theta^*(x,X)\Theta^*(X)^{-1}) \\ &\approx \frac{\bar{\kappa}_2^2}{(\bar{\kappa}_1 - \bar{\kappa}_2)^2}(1 - \frac{\bar{\kappa}_2 S}{\bar{\kappa}_1 + (S-1)\bar{\kappa}_2})^2 tr(K(X)\mathbb{1}_S\mathbb{1}_S^T) \\ &= \frac{S^2}{(\bar{\kappa}_1/\bar{\kappa}_2 + (S-1))^2}(\frac{1}{S}\langle q^L(x_s)\rangle + (1 - \frac{1}{S})\langle q^L_{sr}(x_s,x_r)\rangle).\end{aligned}$$

Taking expectation of the above expressions over a random dataset D, which is independent to random initialization w_0, we get

$$\begin{aligned}
\mathbb{E}_{w_0,D}[f^0(x)\Theta^0(x,X)\Theta^0(X)^{-1}f^0(X)] &= \frac{S}{\bar{\kappa}_1/\bar{\kappa}_2+(S-1)}\mathbb{E}_X[\langle q^L_{sr}(x_s,x)\rangle] \\
&= \frac{S}{\bar{\kappa}_1/\bar{\kappa}_2+(S-1)}\bar{q}^L_{sr}, \\
\mathbb{E}_{w_0,X}[(\Theta^0(x,X)\Theta^0(X)^{-1}f^0(X))^2] &= \frac{S^2}{(\bar{\kappa}_1/\bar{\kappa}_2+(S-1))^2}\cdot \\
&\quad \cdot \mathbb{E}_X(\frac{1}{S}\langle q^L(x_s)\rangle + (1-\frac{1}{S})\langle q^L_{sr}(x_s,x_r)\rangle) \\
&= \frac{S^2}{(\bar{\kappa}_1/\bar{\kappa}_2+(S-1))^2}(\frac{1}{S}\bar{q}^L + (1-\frac{1}{S})\bar{q}^L_{sr}).
\end{aligned}$$

Putting everything together, we get

$$\begin{aligned}
\mathbb{E}_{w_0,X}[(f^{t=\infty}_{lin}(x))^2] - \mathbb{E}_{w_0,X}[f^{t=\infty}_{lin}(x)]^2 \approx \bar{q}^L - 2\frac{S}{\bar{\kappa}_1/\bar{\kappa}_2+(S-1)}\bar{q}^L_{sr} \\
+\frac{S^2}{(\bar{\kappa}_1/\bar{\kappa}_2+(S-1))^2}(\frac{1}{S}\bar{q}^L + (1-\frac{1}{S})\bar{q}^L_{sr}).
\end{aligned}$$

Denoting $A = \dfrac{S}{\bar{\kappa}_1/\bar{\kappa}_2+(S-1)}$, we can rewrite the above expression as

$$Var(f^{t=\infty}(x)) \approx (1+\frac{A^2}{S})(\bar{q}^L - \bar{q}^L_{sr}) + (A-1)^2\bar{q}^L_{sr}.$$

Appendix C. Effects of biases on the NTK variance at initialization

Figure 7 shows the dependence of the NTK variance at initialization on σ_b^2. One can see that lower σ_b^2 values yield narrower boundary between the two phases, but the general picture stays similar to the one in Figure 1.

Appendix D. Additional experiments on the NTK change during training

Here we provide additional figures on changes of the NTK during gradient descent training.

Figures 8 and 9 show changes in the NTK matrix norm as a function of the number of training steps for `tanh` and ReLU networks, respectively. One can see how the NTK changes after $10, 10^2, 10^3$ and 10^4 training steps. The findings from these figures are similar to the analysis we provided in Section 4: the NTK behaviour changes significantly around the border between ordered and chaotic phases. One can also see that for deep networks in the chaotic phase the NTK changes significantly already in the early stages of training, while networks in the ordered phase display very low changes in the NTK norm for a long time.

Figures 10 and 11 show the effects of the network width on the changes of the NTK matrix during training. We provide experiments for $M = 128, 256, 512$. One can see that, as expected in NTK theory, higher M values overall result in smaller changes of the NTK. However, with all the width values, one can see the transition from ordered to chaotic phase, which gets more pronounced with the network's depth.

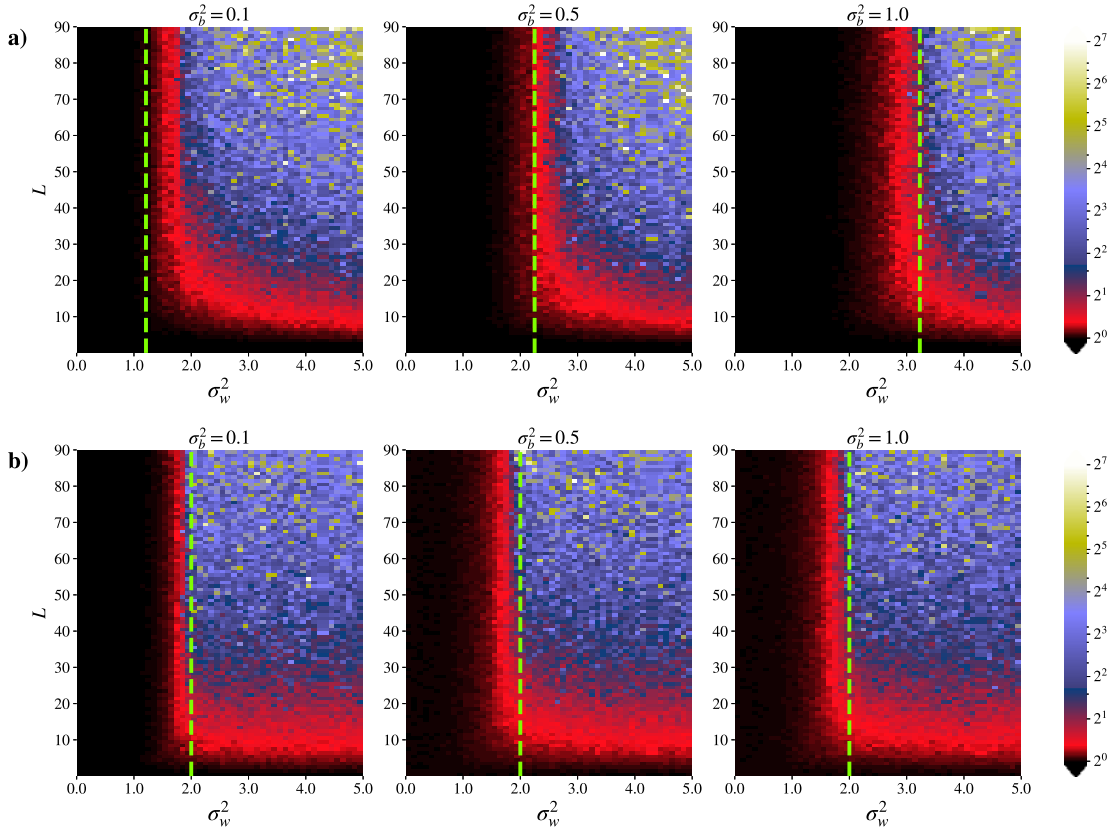


Figure 7: $\frac{\mathbb{E}[\Theta^0(x,x)^2]}{\mathbb{E}^2[\Theta^0(x,x)]}$ ratio for fully-connected a) `tanh`, b) ReLU networks of width $M = 100$ for different σ_b values. The dashed line shows the theoretical border between ordered and chaotic phases ($\chi_1^l = 1$) for the given hyperparameters. For `tanh` networks the location of the border between phases depends on σ_b^2, while for ReLU networks it is the same for all the σ_b^2 values.

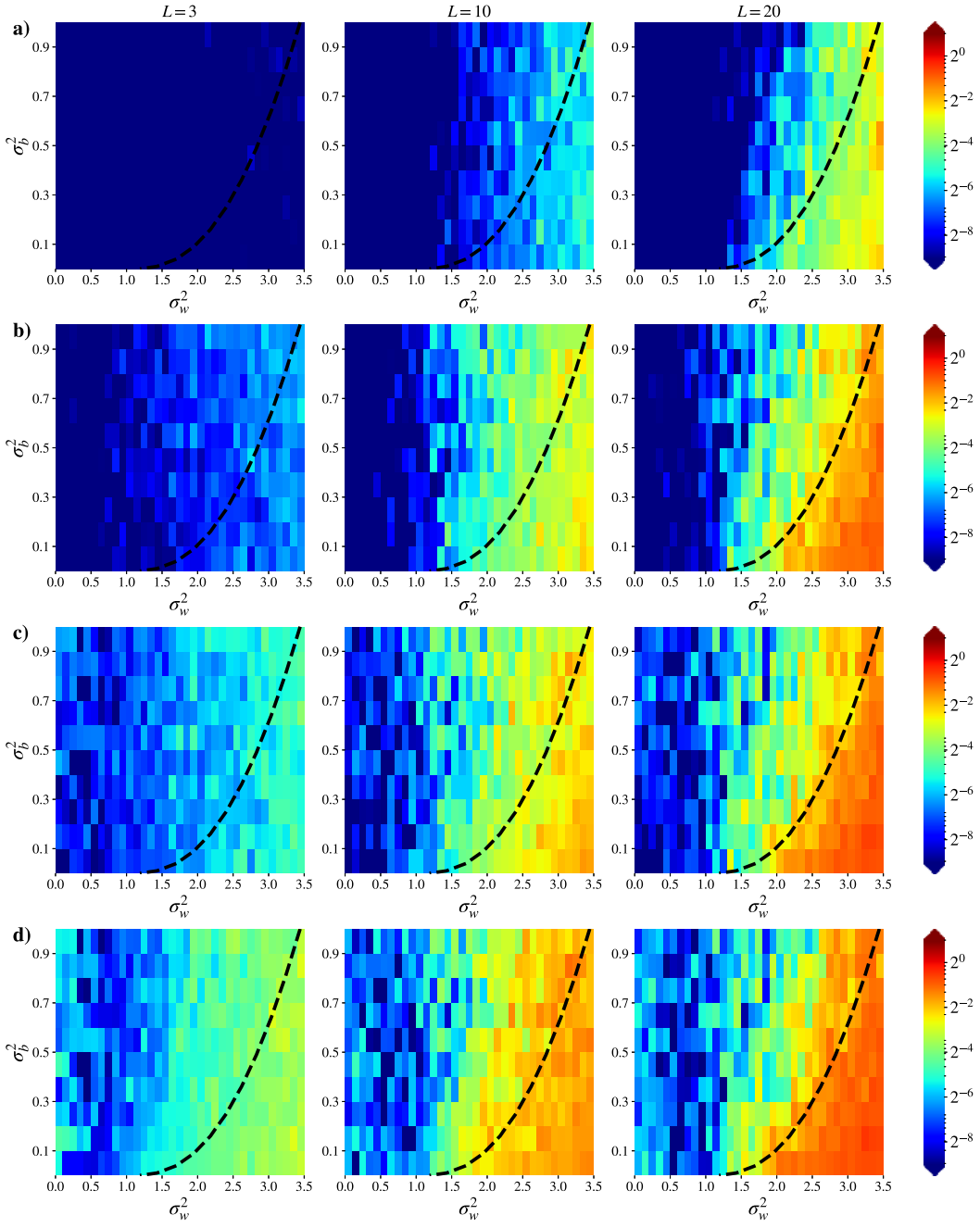


Figure 8: Relative change in the NTK norm $\|\Theta^t - \Theta^0\|_F / \|\Theta^0\|_F$ for `tanh` networks after a) 10, b) 10^2, c) 10^3, d) 10^4 gradient descent steps. The training parameters are the same as in Figure 3.

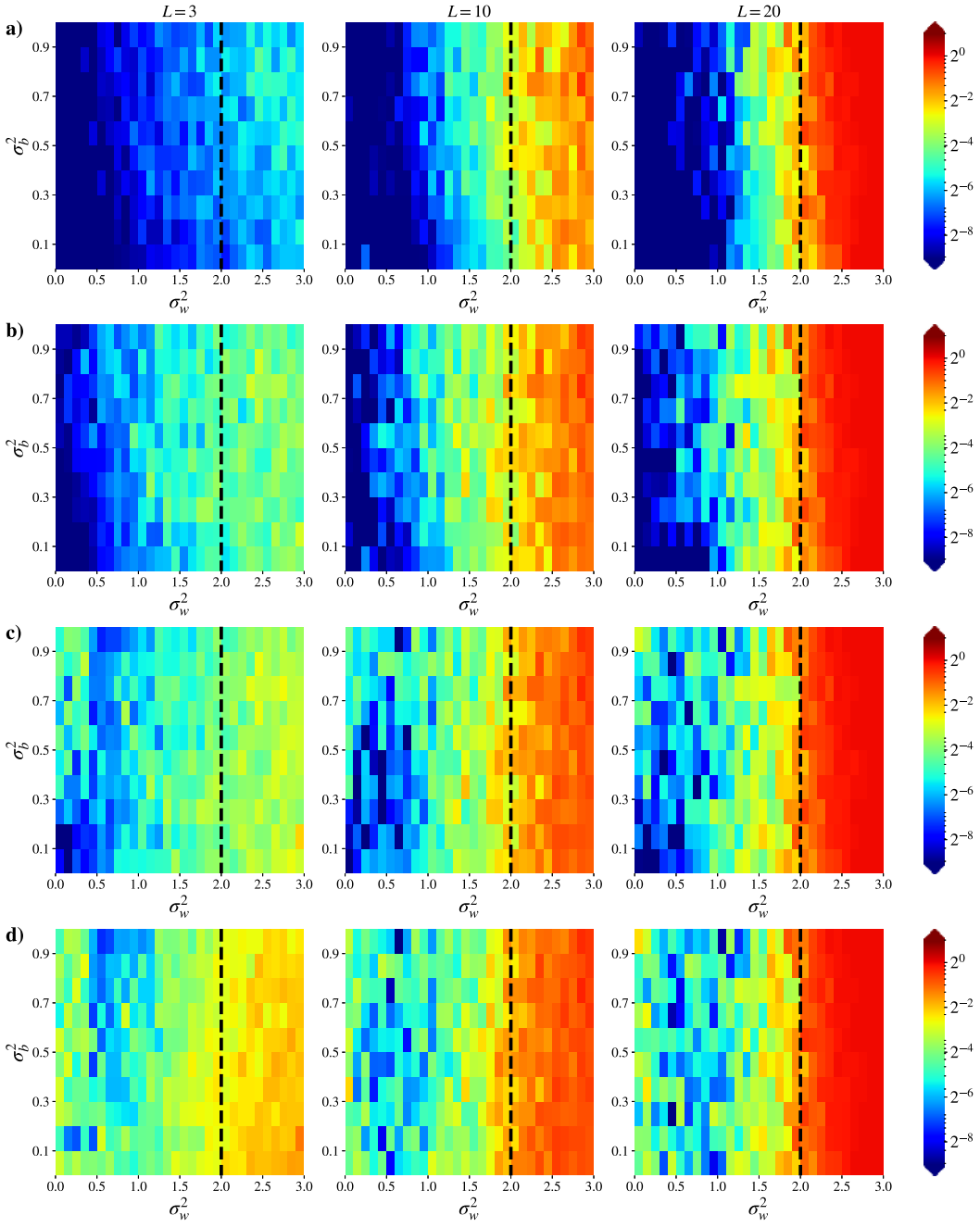


Figure 9: Relative change in the NTK norm $\|\Theta^t - \Theta^0\|_F / \|\Theta^0\|_F$ for ReLU networks after a) 10, b) 10^2, c) 10^3, d) 10^4 gradient descent steps. The training parameters are the same as in Figure 4.

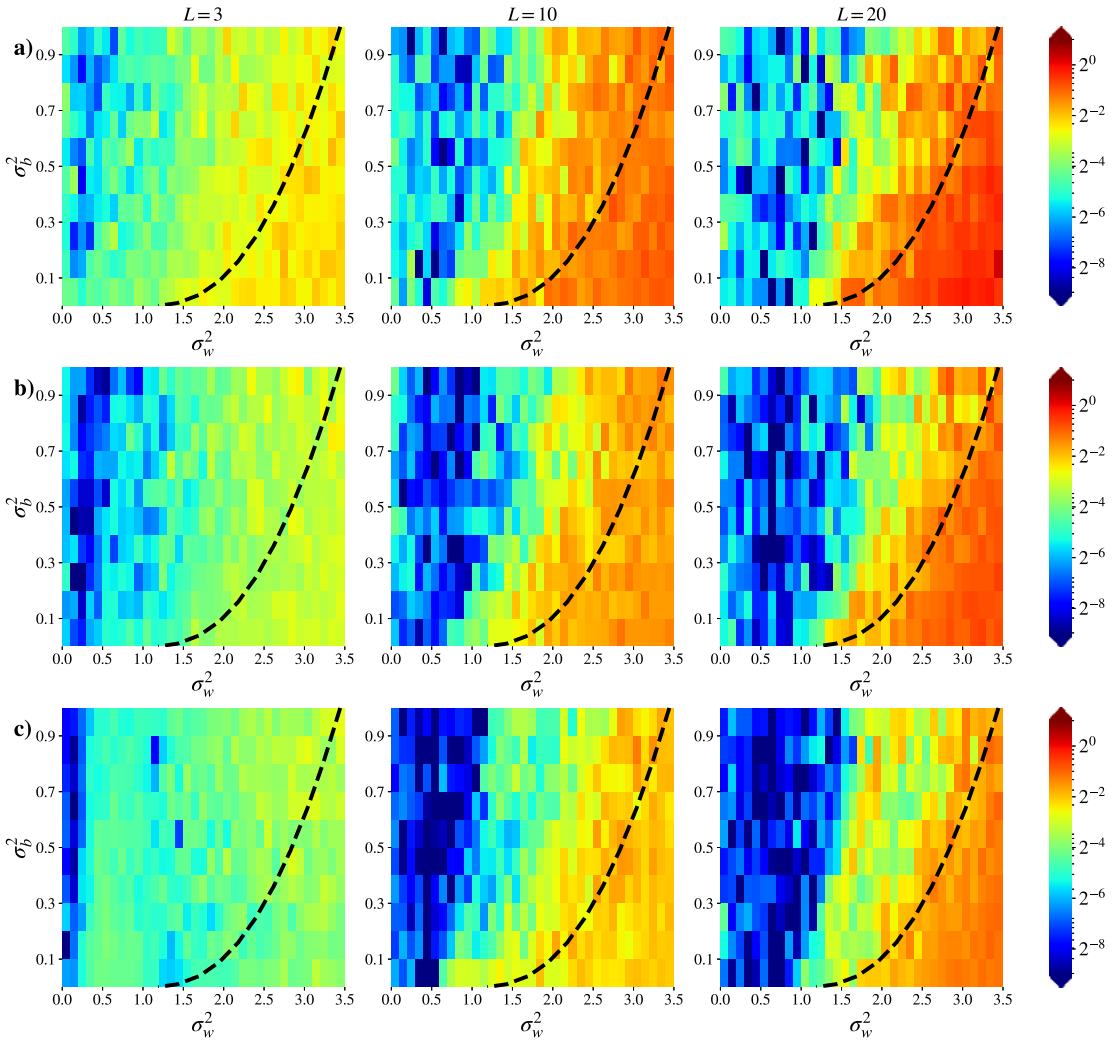


Figure 10: Relative change in the NTK norm $\|\Theta^t - \Theta^0\|_F / \|\Theta^0\|_F$ for `tanh` networks of width a) $M = 128$, b) $M = 256$, c) $M = 512$ in the end of training. The training parameters are the same as in Figure 3.

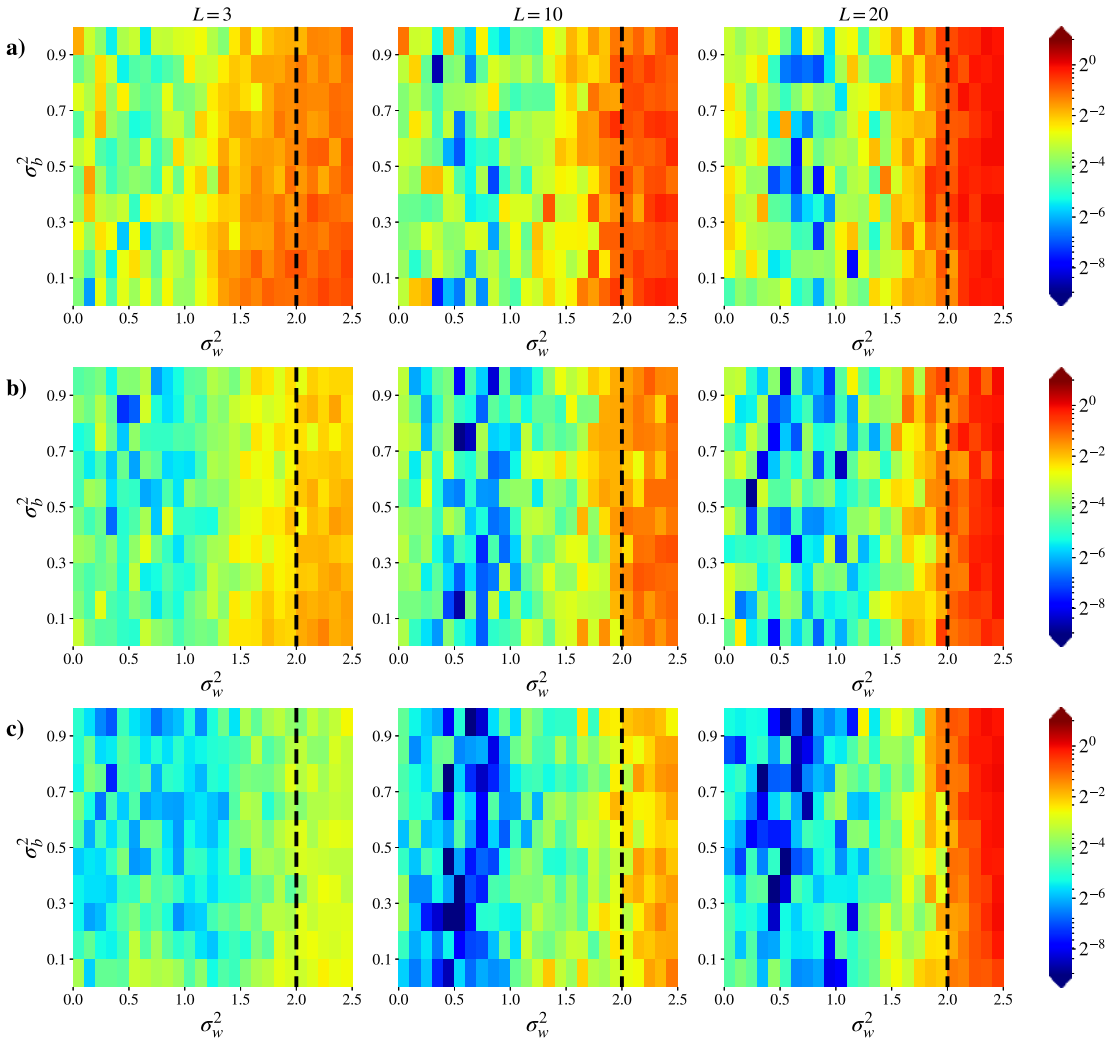


Figure 11: Relative change in the NTK norm $\|\Theta^t - \Theta^0\|_F / \|\Theta^0\|_F$ for ReLU networks of width a) $M = 128$, b) $M = 256$, c) $M = 512$ in the end of training. The training parameters are the same as in Figure 4.

Appendix E. Analytical relations for integrals in Section 2

E.1. ReLU networks

ReLU activation function is defined by

$$\phi(x) = \begin{cases} x & x > 0, \\ 0 & x \leq 0. \end{cases}$$

Then to obtain analytical expressions for q^l and q^l_{sr} we can take the following integrals, which appear in (5) and (8):

$$\int Dz \cdot \phi(az)^2 = a^2/2,$$
$$\int Dz \cdot [\phi^{'}(az)]^2 = 1/2,$$

Then we immediately get

$$q^l = \frac{\sigma_w^2}{2} q^{l-1} + \sigma_b^2,$$
$$p^{l-1} = \frac{\sigma_w^2}{2} p^l \frac{M_l}{M_{l+1}}.$$

Similarly, to get analytical expressions for q^l_{sr} and p^l_{sr}, we can take the integrals in (7) and (9):

$$\int Dz_1 Dz_2 \cdot \phi(az_1)\phi(bz_1 + \sqrt{a^2 - b^2} z_2) = \frac{a}{2\pi}(\sqrt{1-c^2} + c\pi/2 + c \arcsin(c)),$$
$$\int Dz_1 Dz_2 \cdot \phi^{'}(az_1)\phi^{'}(bz_1 + \sqrt{a^2 - b^2} z_2) = \frac{1}{2\pi}(\pi/2 + \arcsin(c)),$$

where $c = b/a$, to obtain the following expressions:

$$q^l_{sr} = \frac{\sigma_w^2}{2\pi} q^{l-1}(\sqrt{1-c^2} + c\pi/2 + c \arcsin c) + \sigma_b^2,$$
$$p^{l-1}_{sr} = \frac{\sigma_w^2}{2\pi} p^l \frac{M_l}{M_{l+1}}(\pi/2 + \arcsin c),$$

where $c = q^{l-1}_{st}/q^{l-1}$.

Then, to compute the values of q^l, q^l_{st}, p^l and p^l_{st} in all the layers, we only need to set the following initial conditions: $q^0 = 1$ when data is normalized, $q^0_{st} \in [0, 1]$ is the covariance between two inputs, $p^L = p^L_{st} = 1$ as the output depends linearly on the activations in the last layer.

E.2. `Erf` networks

Error function, which is a kind of sigmoid functions, is defined by

$$\phi(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt.$$

Then, same as for ReLU activation, we analytically take the integrals from (5) and (8):

$$\int Dz \cdot \phi(az)^2 = \frac{2}{\pi} \arctan \frac{a^2}{\sqrt{a^2 + 1/4}},$$
$$\int Dz \cdot [\phi^{'}(az)]^2 = \frac{2}{\pi} \frac{1}{\sqrt{a^2 + 1/4}}$$

to obtain expressions for q^l and p^l:

$$q^l = \frac{2\sigma_w^2}{\pi} \arctan \frac{q^{l-1}}{\sqrt{q^{l-1} + 1/4}} + \sigma_b^2,$$
$$p^{l-1} = \frac{2\sigma_w^2}{\pi} p^l \frac{1}{\sqrt{q^{l-1} + 1/4}} \frac{M_l}{M_{l+1}}.$$

And similarly we take the integrals in (7) and (9):

$$\int Dz_1 Dz_2 \cdot \phi(az_1)\phi(bz_1 + \sqrt{a^2 - b^2} z_2) = \frac{2}{\pi} \arctan \frac{2b}{\sqrt{(1+2a)^2 - 4b^2}},$$
$$\int Dz_1 Dz_2 \cdot \phi^{'}(az_1)\phi^{'}(bz_1 + \sqrt{a^2 - b^2} z_2) = \frac{4}{\pi} \frac{1}{\sqrt{(1+2a)^2 - 4b^2}},$$

to obtain the analytical expressions for q_{sr}^l and p_{sr}^l:

$$q_{sr}^l = \frac{2\sigma_w^2}{\pi} \arctan \frac{2\sqrt{q_{sr}^{l-1}}}{\sqrt{(1 + 2\sqrt{q^{l-1}})^2 - 4q_{sr}^{l-1}}} + \sigma_b^2,$$
$$p_{sr}^{l-1} = \frac{4\sigma_w^2}{\pi} p^l \frac{M_l}{M_{l+1}} \frac{1}{\sqrt{(1 + 2\sqrt{q^{l-1}})^2 - 4q_{sr}^{l-1}}}.$$

And the initial conditions can be specified in the same way as for the ReLU networks in the previous subsection.

3.2 Neural Tangent Kernel Beyond the Infinite-Width Limit: Effects of Depth and Initialization

Contributing article: Seleznova, M. and Kutyniok, G. (2022b). Neural Tangent Kernel Beyond the Infinite-Width Limit: Effects of Depth and Initialization. In *Proceedings of the 39th International Conference on Machine Learning*, pages 19522–19560. PMLR.

Author contributions: Mariia Seleznova developed the original research idea to derive the infinite-depth-and-width limit of the NTK for different phases of initialization. In fact, this idea stems from the empirical results of the previous paper (Seleznova and Kutyniok, 2022a). Mariia Seleznova formulated all the theorems and derived all the proofs presented in the paper, designed and programmed all the numerical experiments, wrote the paper's main text and appendices, and designed all the figures. As the main author, Mariia Seleznova also managed the publication process: paper submission to the conference, writing a rebuttal after the initial reviews, addressing reviewers' concerns, and producing the camera-ready version of the paper. Gitta Kutyniok took part in the project discussions at all the stages, provided feedback, reviewed and proofread the paper.

Additional resources:

- Paper link: https://proceedings.mlr.press/v162/seleznova22a.html
- Slides: https://icml.cc/media/icml-2022/Slides/16473.pdf
- Video presentation: https://slideslive.com/38983579
- Source code: https://github.com/mselezniova/ntk_beyond_limit

Neural Tangent Kernel Beyond the Infinite-Width Limit: Effects of Depth and Initialization

Mariia Seleznova [1] **Gitta Kutyniok** [1]

Abstract

Neural Tangent Kernel (NTK) is widely used to analyze overparametrized neural networks due to the famous result by Jacot et al. (2018): in the infinite-width limit, the NTK is deterministic and constant during training. However, this result cannot explain the behavior of deep networks, since it generally does not hold if depth and width tend to infinity simultaneously. In this paper, we study the NTK of fully-connected ReLU networks with depth comparable to width. We prove that the NTK properties depend significantly on the depth-to-width ratio and the distribution of parameters at initialization. In fact, our results indicate the importance of the three phases in the hyperparameter space identified in Poole et al. (2016): *ordered*, *chaotic* and the *edge of chaos* (EOC). We derive exact expressions for the NTK dispersion in the infinite-depth-and-width limit in all three phases and conclude that the NTK variability grows exponentially with depth at the EOC and in the chaotic phase but not in the ordered phase. We also show that the NTK of deep networks may stay constant during training only in the ordered phase and discuss how the structure of the NTK matrix changes during training.

1. Introduction

Despite the widespread use of Deep Neural Networks (DNNs), the theory behind their success is still poorly understood. For instance, no present theory can explain why highly overparametrized DNNs generalize very well in practice, contrary to classical statistical learning theory predictions. Likewise, it is surprising that optimizing a highly non-convex loss function of a DNN with a variant of Gradient Descent (GD) typically yields a good local minimum.

Although training dynamics and generalization capabilities of DNNs stand among the biggest open problems of deep learning theory, it is possible to address these challenges in the special case of infinitely-wide DNNs using the so-called *Neural Tangent Kernel* (NTK). This kernel captures the first-order approximation of DNN's evolution during GD training. Consider the gradient flow dynamics of the DNN's parameters:

$$\dot{\mathbf{w}} = -\nabla_{\mathbf{w}}\mathcal{L}(\mathcal{D}) = -\sum_{(x_i,y_i)\in\mathcal{D}} \nabla_{\mathbf{w}} f(x_i)\frac{\partial \mathcal{L}(\mathcal{D})}{\partial f(x_i)}, \quad (1)$$

where $\mathbf{w}$ is the vector of all the trainable parameters, $f(\cdot)$ is the DNN's output function (defined in Section 2), $\mathcal{L}(\cdot)$ is the loss function and $\mathcal{D}$ is the dataset. Then the dynamics of the DNN's output function is given by:

$$\dot{f}(x) = \nabla_{\mathbf{w}} f(x)\cdot\dot{\mathbf{w}} = -\sum_{(x_i,y_i)\in\mathcal{D}} \Theta(x,x_i)\frac{\partial \mathcal{L}(\mathcal{D})}{\partial f(x_i)}, \quad (2)$$

where the kernel $\Theta(x_i,x_j) := \left\langle \nabla_{\mathbf{w}} f(x_i), \nabla_{\mathbf{w}} f(x_j)\right\rangle$ is called the NTK.

A famous result by Jacot et al. (2018) states that in the infinite-width limit, the NTK is deterministic under proper random initialization and stays constant during training. Thereby, the dynamics in (2) is equivalent to kernel regression and has an analytical solution expressed in terms of the kernel. It is then possible to derive properties of trained infinitely-wide DNNs theoretically by means of their NTKs. Hence, many recent works used the NTK to explain empirically known properties of DNNs (Huang et al., 2020; Adlam & Pennington, 2020; Wang et al., 2022; Tirer et al., 2021; Geiger et al., 2019). Numerous contributions also derived the infinite-width limit of the NTK for popular DNN architectures (Yang, 2020; Du et al., 2019; Alemohammad et al., 2021). Other papers established some non-asymptotic results on the concentration of the NTK at initialization (Arora et al., 2019; Buchanan et al., 2021) and stability of the NTK during training (Huang & Yau, 2020; Lee et al., 2019).

However, the extent to which the results in the infinite-width limit extrapolate to realistic DNNs remains largely

[1]Department of Mathematics, Ludwig-Maximilians-Universität München, Munich, Germany. Correspondence to: Mariia Seleznova <selez@math.lmu.de>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

an open question. Indeed, multiple authors have argued that the NTK regime and, in general, the infinite-width limit cannot explain the success of DNNs (Chizat et al., 2019; Hanin & Nica, 2020; Aitchison, 2020; Li et al., 2021; Seleznova & Kutyniok, 2021; Bai et al., 2020; Huang & Yau, 2020). The first argument in this direction is that no feature learning occurs if the NTK stays constant during training. Moreover, several works showed that the infinite-width limit of the NTK becomes completely data-independent as depth increases (Xiao et al., 2020; Hayou et al., 2019), which suggests poor generalization performance for deep networks in the NTK regime. Finally, numerous empirical results demonstrated that the performance of trained DNNs and the corresponding kernel methods often differs in practice (Fort et al., 2020; Lee et al., 2020; Arora et al., 2020). That is why it is essential to understand the statistical properties of the NTK and how they depend on the myriad of settings of a given DNN to assess if the infinite-width limit provides a reasonable approximation for this network. We contribute to this line of research by exploring the combined effect of two factors on the NTK: the network's depth and initialization hyperparameters.

Network's depth Most results on the NTK are derived in the setting where the network's depth is kept constant while the width tends to infinity. This limit can only model very wide and shallow networks since the depth-to-width ratio tends to zero in it. Indeed, several recent papers demonstrated that infinite-width approximations often get worse as the depth increases (Li et al., 2021; de G. Matthews et al., 2018; Yang & Schoenholz, 2017). In particular, Hanin & Nica (2020) first showed that the NTK of fully-connected ReLU DNNs may be random and change during training if depth and width are comparable. Hu & Huang (2021) also studied the effects of depth on the NTK distribution and derived an upper bound for the NTK moments. We expand on these results by precisely characterizing the variability of the NTK at initialization and generalizing to different initialization settings described below.

Initialization hyperparameters There are three phases in the initialization hyperparameter space where the properties of untrained infinitely-wide DNNs differ significantly: *ordered*, *chaotic* and the *edge of chaos* (EOC) (Poole et al., 2016). In the ordered phase, the gradient norms decrease with depth, whereas in the chaotic phase the gradient norms increase, and the edge of chaos is the initialization at the border between these two phases (Schoenholz et al., 2017). The results by Hanin & Nica (2020) concerned the statistical properties of the NTK of wide and deep ReLU networks at the EOC. At the same time, several contributions demonstrated that the properties of the infinite-width NTK depend significantly on the phase of initialization (Xiao et al., 2020; Hayou et al., 2019). However, these results do not apply to networks with depth comparable to width since they assume infinite width before considering the effects of growing depth. We fill this gap by deriving statistical properties of the NTK for wide and deep ReLU networks in all three phases of initialization.

1.1. Contributions

We study the **variability of the NTK at initialization** for fully-connected ReLU DNNs with depth comparable to width and varying initialization hyperparameters in Section 3. Our contributions are as follows:

- We precisely characterize the dispersion of the diagonal elements $\Theta(x, x)$ of the NTK (for arbitrary input x) in the **infinite-depth-and-width limit** and conclude that the variability of the NTK grows exponentially with the depth-to-width ratio at the EOC and in the chaotic phase. Conversely, the variance of $\Theta(x, x)$ tends to zero in the same limit in the ordered phase. Our results allow to evaluate the variance of the NTK for a given DNN with any depth-to-width ratio and initialization.
- We provide non-asymptotic expressions for the first two moments of $\Theta(x, x)$ and illustrate **finite-width effects** that follow. We show that the variance of the finite-width NTK in the ordered phase gradually increases as the initialization approaches the EOC, which describes the transition between the two kinds of behavior in the limit. We also notice that the NTK dispersion depends on the architecture, i.e. on the varying widths of the fully-connected layers. Notably, the dispersion of $\Theta(x, x)$ decreases with depth in the ordered phase if the DNN increases the dimensionality in consecutive layers. This enables us to conclude that deeper networks are more robust to random initialization in this setting.
- We lower-bound the ratio of the expected **non-diagonal elements** of the NTK, i.e. $\Theta(x, \tilde{x})$ with $x \neq \tilde{x}$, and the diagonal elements $\Theta(x, x)$ in the infinite-depth-and-width limit. We also upper-bound the dispersion of the non-diagonal elements. In the ordered phase, our results allow to ensure that the whole NTK matrix is approximately deterministic and thus can be approximated by the infinite-width limit.
- We provide extensive **numerical experiments** to verify our theoretical results. We use JAX (Bradbury et al., 2018) and Flax (neural network library for JAX) (Heek et al., 2020) to compute the NTK of fully-connected ReLU networks effortlessly. Source code to reproduce the presented results is available at: https://github.com/mselezniova/ntk_beyond_limit.

We study the **training dynamics of the NTK** for fully-connected ReLU DNNs with depth comparable to width and varying initialization hyperparameters in Section 4. Our contributions are as follows:

- We show that the expected relative change of $\Theta(x, x)$ in **the first GD step** tends to infinity in the chaotic phase and to zero in the ordered phase in the infinite-depth-and-width limit. Combined with the result by Hanin & Nica (2020), which states that the expected relative change of $\Theta(x, x)$ in the first GD step is exponential in the depth-to-width ratio at the EOC, we can conclude that the NTK of deep networks can stay approximately constant during GD training only in the ordered phase.
- We discuss how the **structure of the NTK** matrix changes during training outside of the NTK regime. The NTK matrix at initialization has a diagonal structure with larger values on the main diagonal as compared to the non-diagonal ones. We speculate that the training process changes the NTK structure to block-diagonal with blocks of larger values corresponding to classes and provide experiments to support this sentiment.

2. Preliminaries

We consider fully-connected ReLU DNNs of depth $L \in \mathbb{N}$ with linear output layer and widths $(n_\ell)_{0 \le \ell \le L}$, where $n_0 \in \mathbb{N}$ is the input dimension and $n_L = 1$ is the output dimension. Forward propagation in such a network is defined as follows:

$$\begin{aligned} \mathbf{x}^\ell(x) &:= \phi(\mathbf{h}^\ell(x)), \quad \mathbf{x}^0(x) := x \in \mathbb{R}^{n_0}, \\ \mathbf{h}^\ell(x) &:= \mathbf{W}^\ell \mathbf{x}^{\ell-1}(x) + \mathbf{b}^\ell, \quad 1 \le \ell \le L-1, \\ f(x) &:= \mathbf{W}^L \mathbf{x}^{L-1}(x) + \mathbf{b}^L \in \mathbb{R}, \end{aligned} \tag{3}$$

where $\phi(x) := x \mathbb{1}_{\{x>0\}}$ denotes the ReLU function, $\mathbf{W}^\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ and $\mathbf{b}^\ell \in \mathbb{R}^{n_\ell}$ are the weights and the biases and $f(x)$ is the output function of the DNN. The NTK of this network on a pair of inputs $(x, \tilde{x})$ is given by:

$$\begin{aligned} \Theta(x, \tilde{x}) &:= \Theta_W(x, \tilde{x}) + \Theta_b(x, \tilde{x}), \\ \Theta_W(x, \tilde{x}) &:= \sum_{\ell=1}^{L} \sum_{j=1}^{n_\ell} \sum_{i=1}^{n_{\ell-1}} \frac{\partial f(x)}{\partial \mathbf{W}_{ij}^\ell} \frac{\partial f(\tilde{x})}{\partial \mathbf{W}_{ij}^\ell}, \\ \Theta_b(x, \tilde{x}) &:= \sum_{\ell=1}^{L} \sum_{j=1}^{n_\ell} \frac{\partial f(x)}{\partial \mathbf{b}_j^\ell} \frac{\partial f(\tilde{x})}{\partial \mathbf{b}_j^\ell}, \end{aligned} \tag{4}$$

where $\Theta_W(x, \tilde{x})$ comprises the gradients w.r.t. the weights and $\Theta_b(x, \tilde{x})$ — the gradients w.r.t. the biases.

When we consider wide networks with unequal widths in the hidden layers, we define a width scale parameter M and constants λ, $(\alpha_\ell)_{0 \le \ell \le L-1}$ such that:

$$\frac{L}{M} = \lambda \in \mathbb{R}, \quad \frac{n_\ell}{M} = \alpha_\ell \in \mathbb{R}, \quad 0 \le \ell \le L-1. \tag{5}$$

Then we can describe the asymptotic behavior of the NTK in terms of M and the constants defined above.

2.1. Initialization and parametrization

We consider random i.i.d. initialization given by:

$$\mathbf{W}_{ij}^\ell \sim \mathcal{N}\Big(0, \frac{\sigma_w^2}{n_{\ell-1}}\Big), \quad \mathbf{b}_i^\ell \sim \mathcal{N}(0, \sigma_b^2), \tag{6}$$

where (σ_w, σ_b) are the initialization hyperparameters. This initialization corresponds to the so-called standard parametrization (SP), where the weights and the biases defined in (3) are the trainable parameters. We note that the NTK is often considered in the so-called NTK parametrization (NTP), where the weights in (3) are the scaled versions of trainable parameters: $\mathbf{W}_{ij}^\ell = \sigma_w / \sqrt{n_{\ell-1}} \mathbf{w}_{ij}^\ell$ for trainable $\mathbf{w}^\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ initialized as $\mathbf{w}_{ij}^\ell \sim \mathcal{N}(0, 1)$ i.i.d. This reparametrization, of course, does not change the distribution of the DNN's components. However, it scales the gradients by $O(1/M)$ and gives the NTK a well-defined infinite-width limit for fixed L. At the same time, NTP is equivalent to setting an individual learning rate in each layer inverse-proportionally to width, as explained, e.g., in Yang & Hu (2021). In this paper, we focus on the NTK in SP since this parametrization is more common in practice (indeed, SP is the default setting in PyTorch). However, our results can be generalized to NTP straightforwardly.

2.2. Information propagation in DNNs

Results on information propagation in infinitely-wide DNNs established that the initialization hyperparameters (σ_w, σ_b) determine the evolution of the variances $\mathbb{E}[(\mathbf{x}_i^\ell(x))^2]$ and the covariances $\mathbb{E}[\mathbf{x}_i^\ell(x)\mathbf{x}_i^\ell(\tilde{x})]$ as they propagate through the DNN's layers. Based on this, Poole et al. (2016) identified three phases with distinct properties in the hyperparameter space: *ordered*, *chaotic* and the *edge of chaos* (EOC). Schoenholz et al. (2017) subsequently showed that the ordered phase corresponds to vanishing gradients and the chaotic phase corresponds to exploding gradients, i.e. the gradient norm decreases with depth in the ordered phase and increases in the chaotic phase. The edge of chaos is the initialization at the border between these two phases, which allows deeper signal propagation through a DNN by avoiding vanishing or exploding gradients. Consider backpropagation equations given by:

$$\begin{aligned} \frac{\partial f(x)}{\partial \mathbf{W}_{ij}^\ell} &= \boldsymbol{\delta}_i^\ell \mathbf{x}_j^{\ell-1}, \quad \frac{\partial f(x)}{\partial \mathbf{b}_i^\ell} = \boldsymbol{\delta}_i^\ell, \\ \boldsymbol{\delta}_i^\ell &:= \frac{\partial f(x)}{\partial \mathbf{h}_i^\ell} = \phi'(\mathbf{h}_i^\ell) \sum_j \boldsymbol{\delta}_j^{\ell+1} \mathbf{W}_{ji}^{\ell+1}, \end{aligned} \tag{7}$$

Schoenholz et al. (2017) studied the evolution of $\mathbb{E}[(\boldsymbol{\delta}_i^\ell)^2]$ along with $\mathbb{E}[(\mathbf{x}_i^\ell)^2]$ to find the distribution of DNNs' gradients. Some recent publications used these results to derive the properties of the infinite-width NTK in all the three phases of initialization (Karakida et al., 2019; Xiao et al.,

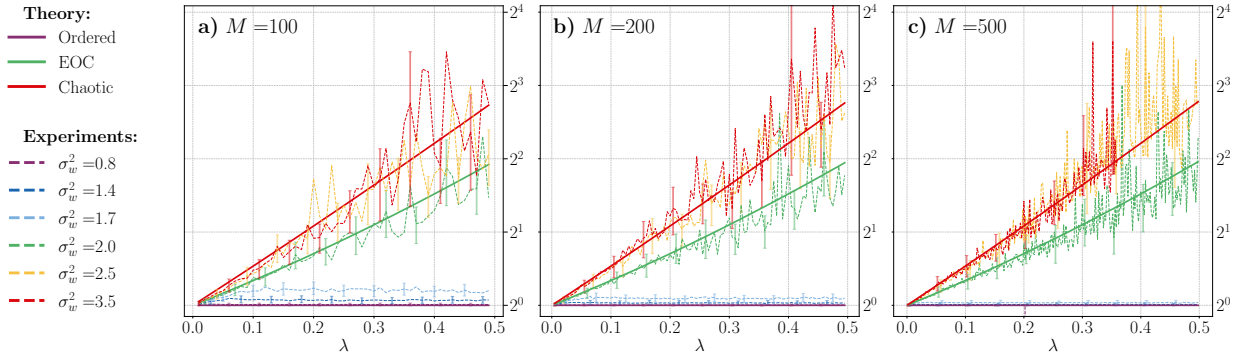


Figure 1. Ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ at initialization for fully-connected ReLU networks of constant width $M \in \{100, 200, 500\}$ with $\alpha_0 = 1$. The dashed lines represent the experimental results and the solid lines correspond to the theoretical predictions from Theorem 3.1. For each DNN configuration, we sampled 500 random initializations and computed an unbiased estimator for the ratio (see details in Appendix C.3). The error bars (indicated by the vertical lines) show the bootstrap estimation of the standard error (only in a subset of points to keep the figure readable). We provide additional figures with continuous error bars in Appendix C.2.

2020). Hayou et al. (2019) also showed that the infinite-*depth* limit of the infinite-width NTK (when first the limit $M \to \infty$ is taken with fixed L and then $L \to \infty$) yields a data-independent kernel and thus cannot explain properties of finite DNNs. Although our approach is different from the mentioned results since we do not assume infinite width before increasing depth, we show that the statistical properties of $\boldsymbol{\delta}^\ell$ and $\mathbf{x}^\ell$ can still be derived and lead to results on the NTK in our setting.

The initialization hyperparameters that comprise each phase differ depending on the chosen activation function. Since we are interested in ReLU networks, we note that the ordered phase corresponds to $\sigma_w^2 < 2$ and the chaotic phase — to $\sigma_w^2 > 2$ for this activation function. The EOC is the initialization with $\sigma_w^2 = 2$. We refer, e.g., to Schoenholz et al. (2017) for a method to compute the border between phases for a given activation function.

3. Variability of the NTK

In the infinite-width limit, the NTK is deterministic under random initialization, which is one of the main results of the NTK theory. We investigate when this result holds outside of the NTK limit and, consequently, when the infinite-width behavior of the NTK gives a good approximation for realistic DNNs.

3.1. Infinite-depth-and-width limit

Most results on the NTK assume that the network's depth is fixed as the width tends to infinity, i.e., $L/M \to 0$ in the limit. This setting, of course, does not describe deep finite-width networks since their depth-to-width ratio is bounded away from zero. Indeed, some recent works demonstrated that infinite-width approximations often get worse as the network's depth increases (Li et al., 2021; de G. Matthews et al., 2018; Yang & Schoenholz, 2017). In particular, Hanin & Nica (2020) considered this effect for the NTK and derived bounds for the ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ in case of ReLU DNNs initialized at the EOC ($\sigma_w = 2$). This ratio characterizes the dispersion of the NTK: it is close to one if the NTK is approximately deterministic and is larger than two if the NTK's distribution is of high variance. Our first main result characterizes this ratio in the infinite-depth-and-width limit under different initializations:

Theorem 3.1 (Dispersion of the NTK at initialization in the limit)**.** *Consider a ReLU DNN as defined in (3) with constant width of hidden layers $M \in \mathbb{N}$, input dimension $n_0 = \alpha_0 M$, $\alpha_0 \in \mathbb{R}$ and output dimension $n_L = 1$. The initialization is given by (6) and the biases are initialized to zero, i.e. $\sigma_b = 0$. Then, in the infinite-depth-and-width limit $M \to \infty$, $L \to \infty$, $L/M \to \lambda \in \mathbb{R}$, the following holds for the dispersion of the NTK:*

*1. In the **chaotic phase** ($a := \sigma_w^2/2 > 1$), the NTK dispersion grows exponentially with depth-to-width ratio $\lambda := L/M$ as follows:*

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \to \frac{1}{2\lambda} e^{5\lambda}\left(1 - \frac{1}{4\lambda}(1 - e^{-4\lambda})\right). \quad (8)$$

*2. At the **EOC** ($a = 1$), the NTK dispersion grows exponentially with depth-to-width ratio λ as well, but with a slower rate given by:*

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \to \frac{1}{(1+\alpha_0)^2}\left[e^{5\lambda}\left(\frac{1}{2\lambda} + \frac{2\alpha_0^2 - 8\alpha_0}{25\lambda^2}\right) + (e^{\lambda} - e^{5\lambda})\frac{1 - 4\alpha_0}{8\lambda^2} + \frac{2\alpha_0}{5\lambda}\left(\frac{4 - \alpha_0}{5\lambda} - 1 - \alpha_0\right)\right]. \quad (9)$$

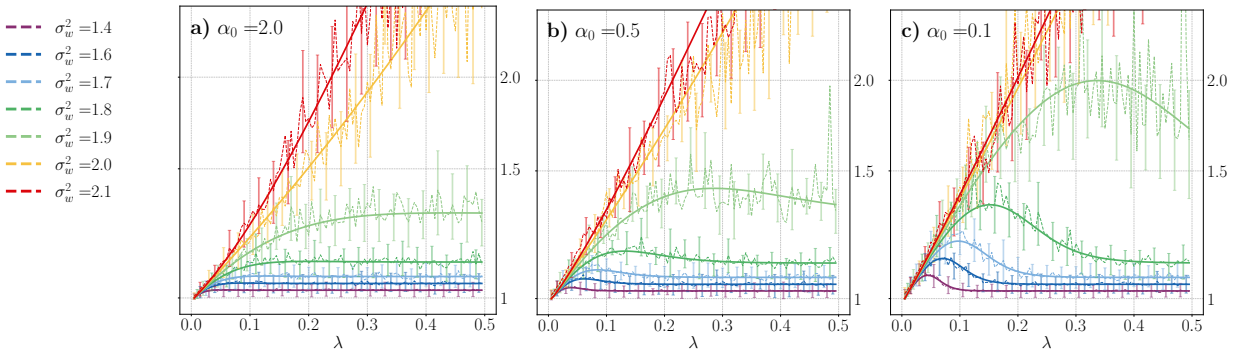


Figure 2. Ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ at initialization for fully-connected ReLU networks of constant width $M = 200$ with the ratio $\alpha_0 := n_0/M \in \{2.0, 0.5, 0.1\}$. The initialization hyperparameter σ_w^2 is close to the EOC for all the lines. The dashed lines represent the experimental results (computed as described in Figure 1) and the solid lines show the theoretical predictions given by Theorem 3.2. The error bars are shown only for a subset of points to keep the figure readable. We provide additional figures with continuous error bars in Appendix C.2.

*3. In the **ordered phase** ($a < 1$), the NTK dispersion tends to one:*

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \to 1. \tag{10}$$

Our numerical experiments in Figure 1 demonstrate that Theorem 3.1 provides accurate approximations for the behavior of sufficiently deep and wide DNNs. Indeed, the proofs listed in Appendix A.1 show that the expressions in the above theorem are true up to the approximation given by $(1 + c/M)^L \approx e^{c\lambda}$ and $O(1/\sqrt{M})$ in the coefficients of the exponents in case of finite width and depth.

Remark 1. The EOC expression in Theorem 3.1 tends to the chaotic phase expression if $\alpha_0 := n_0/M$ tends to zero (i.e. when the input dimension is fixed). We discuss this effect in Appendix B.1.

Remark 2. Model scaling introduced in papers on the so-called "lazy training" phenomenon (Chizat et al., 2019) does not change the results of Theorem 3.1. We discuss lazy training and its effects on our analysis in Appendix B.3.

3.2. Finite depth and width effects

We notice that some features of the NTK dispersion are still not visible in the infinite-depth-and-width limit. One can see in Figure 1 that the NTK variance in the ordered phase is not exactly zero for finite-width DNNs, contrary to the prediction in the limit. This is especially noticeable for initialization close to the EOC, where the transition between the two kinds of limiting behavior occurs. Moreover, Theorem 3.1 cannot reveal the effects of the architecture since it considers only DNNs of constant width. Therefore, we provide non-asymptotic expressions for the first two moments of the NTK at initialization in the following theorem and show that these expressions accurately describe the behavior of finite-width DNNs.

Theorem 3.2 (Moments of the NTK at initialization)**.** *Consider a ReLU DNN defined in (3) with widths scaling defined in (5) and the output dimension $n_L = 1$. The initialization is given by (6) and $\sigma_b = 0$. Then the expectation of the NTK is determined by the following terms:*

$$\mathbb{E}[\Theta_W(x,x)] = \|\mathbf{x}^0\|^2 a^{L-1} \sum_{\ell=1}^{L} \frac{n_{\ell-1}}{n_0}, \tag{11}$$

$$\mathbb{E}[\Theta_b(x,x)] = \sum_{\ell=1}^{L} a^{L-\ell}, \tag{12}$$

where the NTK components Θ_W and Θ_b are defined in (4). Moreover, the second moment of the NTK is determined by:

$$\frac{\mathbb{E}[\Theta_W^2(x,x)]}{\|\mathbf{x}^0\|^4 a^{2(L-1)}} = \mathcal{X}_{(1,L)} \left[\sum_{\ell=1}^{L} \frac{n_{\ell-1}^2}{n_0^2} + \sum_{\ell_1<\ell_2} \frac{n_{\ell_2-1} n_{\ell_1-1}}{n_0^2} \frac{\mathcal{C}_{(\ell_1,\ell_2)}}{\mathcal{X}_{(\ell_1,\ell_2)}} \right], \tag{13}$$

$$\frac{\mathbb{E}[\Theta_b(x,x)^2]}{a^{2L}} = \sum_{\ell=1}^{L} \frac{\mathcal{X}_{(\ell,L)}}{a^{2\ell}} + 2 \sum_{\ell_1<\ell_2} \frac{\mathcal{X}_{(\ell_2,L)}}{a^{\ell_1+\ell_2}}, \tag{14}$$

$$\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{\|\mathbf{x}^0\|^2 a^{2L-1}} = \sum_{\ell=1}^{L} \frac{n_{\ell-1}}{n_0} \frac{\mathcal{X}_{(\ell,L)}}{a^{\ell}} + \sum_{\ell_1<\ell_2} \frac{\mathcal{X}_{(\ell_2,L)}}{a^{\ell_1}} \frac{n_{\ell_1-1}}{n_0} \left(\frac{n_{\ell_2-1}}{n_{\ell_1-1}} \mathcal{C}_{(\ell_1,\ell_2)} + \frac{a^{\ell_1}}{a^{\ell_2}} \right), \tag{15}$$

where we denoted $\mathcal{X}_{(i,j)} := \prod_{k=i}^{j-1} \left(1 + \frac{5}{n_k} + O(M^{-3/2})\right)$, $\mathcal{C}_{(i,j)} := \prod_{k=i}^{j-1} \left(1 + \frac{1}{n_k} + O(M^{-3/2})\right)$ and $a := \sigma_w^2/2$.

These expressions are derived in Appendix A.1 as a part of the proof of Theorem 3.1 and they simplify to the results in the limit by noticing that $\mathcal{X}_{(1,L)} \to e^{5\lambda}$ and $\mathcal{C}_{(1,L)} \to e^{\lambda}$.

Figure 2 examines how well the above expressions approximate the NTK of DNNs with varying ratios $\alpha_0 := n_0/M$ between the input dimension and the width of hidden layers. One can see that the NTK variance in the ordered phase indeed grows as the initialization approaches the EOC. This effect is due to the terms proportional to $\big((a-1)M\big)^{-1}$ in the moments of $\Theta_b(x,x)$ and $\Theta_W(x,x)\Theta_b(x,x)$. When the initialization is close enough to the EOC, $(a-1)^{-1}$ becomes comparable with finite M, and therefore the behavior diverges from the limit.

Another remarkable observation is that the NTK dispersion may decrease with depth in the ordered phase for DNNs that increase the dimensionality (i.e. $n_0 \leq n_1 \leq \dots n_{L-1}$), which means that deeper networks can be more robust. Indeed, in Subfigures b) and c) of Figure 2, the dispersion reaches its peak at a certain depth and then decreases. We provide additional results characterizing this effect in DNNs with non-constant width in hidden layers in Appendix B.2.

3.3. Non-diagonal elements of the NTK

The results stated so far only concern the diagonal elements of the NTK. To generalize to the whole kernel, we provide the following theorem proven in Appendix A.2:

Theorem 3.3 (Non-diagonal elements of the NTK)**.** *Consider a ReLU DNN from Theorem 3.2. The following bounds hold for the ratio of non-diagonal and diagonal elements of the NTK:*

$$1 \geq \lim_{\substack{L\to\infty, M\to\infty \\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta(x,\tilde{x})]}{\mathbb{E}[\Theta(x,x)]} \geq \frac{1}{4}. \tag{16}$$

Moreover, the dispersion of the non-diagonal elements is bounded by the dispersion of diagonal ones:

$$\lim_{\substack{L\to\infty \\ M\to\infty \\ L/M\to\lambda}} \frac{\mathbb{E}[\Theta^2(x,\tilde{x})]}{\mathbb{E}^2[\Theta(x,\tilde{x})]} \leq 16 \lim_{\substack{L\to\infty \\ M\to\infty \\ L/M\to\lambda}} \frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]}. \tag{17}$$

Of course, the bound in (17) is too loose for practical applications if the goal is to prove that the NTK is approximately deterministic. However, we note that the ratio of non-diagonal and diagonal elements can be close to the lower bound only in the chaotic phase. In the ordered phase, our proof suggests the following bound for sufficiently wide and deep networks:

$$\frac{\mathbb{E}[\Theta(x,\tilde{x})]}{\mathbb{E}[\Theta(x,x)]} \gtrsim \frac{\sum_{\ell=1}^{L} a^{L-\ell} \prod_{k=\ell}^{L-1} g(\rho_{k-1})}{\sum_{\ell=1}^{L} a^{L-\ell}}, \tag{18}$$

where $g(t) := \frac{1}{\pi}(\pi/2 + \arcsin t)$ and ρ_k is the infinite-width approximation of the cosine distance between $\mathbf{x}^k$ and $\tilde{\mathbf{x}}^k$, which only increases with depth and is given by applying the function $r(t) := \frac{1}{\pi}\big(\sqrt{1-t^2} + t\pi/2 + t\arcsin t\big)$ to $\langle \mathbf{x}^0, \tilde{\mathbf{x}}^0 \rangle$ consecutively k times. The function $r(\cdot)$ arises from the expectation of a product of two correlated Gaussian variables under ReLU function.

We provide empirical results on the ratio of non-diagonal and diagonal elements of the NTK in Figure 3. We also plot the estimate for the ratio given by (18) in the same figure. One can see that the ratio quickly increases with depth in the ordered phase. Moreover, the lower bound in (18) gives a good approximation for the experimental results. Then for a given network in the ordered phase one can replace the coefficient 16 in the bound (17) with $1/c^2$, where c is a better estimate for the lower bound of $\mathbb{E}[\Theta(x,\tilde{x})]/\mathbb{E}[\Theta(x,x)]$ and can be close to one in the ordered phase.

We also provide experiments on the dispersion of the non-diagonal elements in Appendix C.1. Our results indicate that, in practice, the dispersion here is only slightly higher than the prediction for the diagonal elements. The general picture stays the same as in Figure 1: the dispersion is low and does not grow with depth in the ordered phase but increases exponentially with the depth-to-width ratio at the EOC and in the chaotic phase. The finite-width effects represented in Figure 2 also remain the same for the non-diagonal elements.

3.4. Proof ideas

All our proofs are based on the following decomposition of the NTK:

$$\Theta(x,x) = \sum_{\ell=1}^{L} \|\boldsymbol{\delta}^{\ell}(x)\|^2 \big(\|\mathbf{x}^{\ell-1}(x)\|^2 + 1\big), \tag{19}$$

which directly follows from (4) and the representation of the gradients in backpropagation (7). Using forward-propagation equations (3) and backpropagation equations (7), we derive the first two moments for the ratios $\mathcal{N}_x^{\ell} := \|\mathbf{x}^{\ell}\|^2/\|\mathbf{x}^{\ell-1}\|^2$ and $\mathcal{N}_{\delta}^{\ell} := \|\boldsymbol{\delta}^{\ell}\|^2/\|\boldsymbol{\delta}^{\ell+1}\|^2$ in Lemmas A.1 and A.2. We then notice that $\mathcal{N}_x^{\ell}$ are uncorrelated in different layers of the networks, as well as $\mathcal{N}_{\delta}^{\ell}$, while $\mathcal{N}_x^{\ell}$ and $\mathcal{N}_{\delta}^{\ell}$ in the same layer can be weakly correlated and we quantify the effects of this dependence in Lemma A.4. Given the moments of $\mathcal{N}_x^{\ell}$ and $\mathcal{N}_{\delta}^{\ell}$ and the results on their correlations, we can represent summands of the NTK as the following telescopic products:

$$\|\boldsymbol{\delta}^{\ell}\|^2 \|\mathbf{x}^{\ell-1}\|^2 = \|\mathbf{x}^0\|^2 \|\boldsymbol{\delta}^L\|^2 \prod_{k=1}^{\ell-1} \mathcal{N}_x^k \prod_{p=\ell}^{L-1} \mathcal{N}_{\delta}^p \tag{20}$$

and use this decomposition to compute the expectation and the second moment of the NTK. We derive the first two moments for $\Theta_W(x,x)$ and $\Theta_b(x,x)$ separately in Lemmas A.5 and A.6. These two components have very different

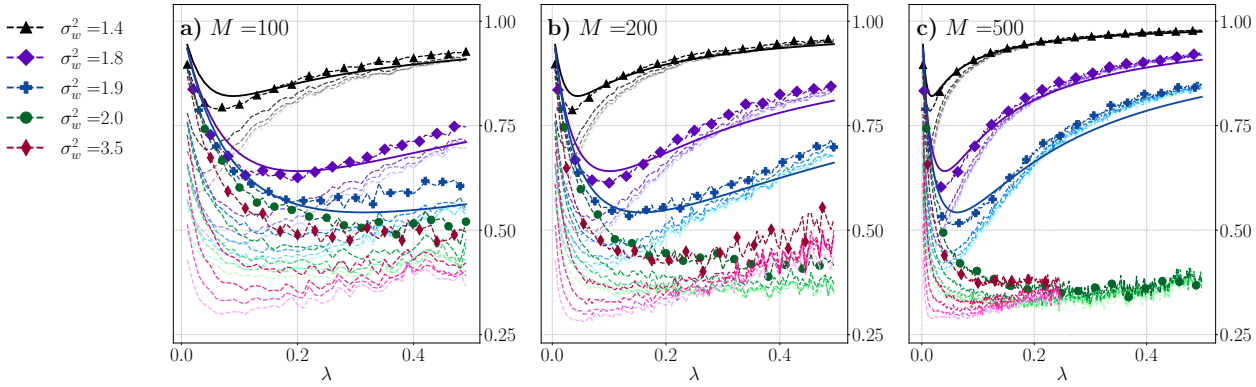


Figure 3. Ratio $\mathbb{E}[\Theta(x,\tilde{x})]/\mathbb{E}[\Theta(x,x)]$ at initialization for fully-connected ReLU networks of constant width $M \in \{100, 200, 500\}$ with $\alpha_0 = 1$. Colors and markers indicate different values of σ_w^2. There are 5 dashed lines for each σ_w^2 value, which correspond to 5 values of the initial angle between input samples $\langle \mathbf{x}^0, \tilde{\mathbf{x}}^0 \rangle \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Darker lines (which also display larger values of the ratio of interest) correspond to larger product $\langle \mathbf{x}^0, \tilde{\mathbf{x}}^0 \rangle$. Expectations are computed by sampling 500 random initializations of each DNN configuration. The solid lines show the estimate for the ratio of interest given by (18) in the ordered phase.

properties in the infinite-depth-and-width limit and, as we show in the proof of Theorem 3.1, the behavior of the NTK is determined by $\Theta_W(x,x)$ in the chaotic phase and by $\Theta_b(x,x)$ in the ordered phase. We also derive the expectation of $\Theta_W(x,x)\Theta_b(x,x)$ in Lemma A.7 to complete the calculations of the second moment of the NTK.

We note that many papers on the NTK use the so-called gradient independence assumption (GIA), which leads to the independence of $\mathcal{N}_x^\ell$ and $\mathcal{N}_\delta^\ell$. This assumption often leads to correct results in the infinite-width limit, as discussed in Yang (2019). However, in our case of infinite depth and width, it may have a non-negligible effect even for simple fully-connected networks with all the weights initialized independently. Thus, we have to calculate this effect explicitly in our proofs. We also note that Li et al. (2021) used a similar technique involving telescoping products of weakly-correlated variables to derive the distribution of the activation norms of ResNets.

4. Training dynamics of the NTK

In the infinite-width limit, the NTK stays constant during training, which allows to study the gradient flow dynamics of infinitely-wide DNNs analytically. In this section, we discuss when this result holds outside of the infinite-width limit and how the empirical NTK changes during training.

4.1. The first GD step

Hanin & Nica (2020) proved that the NTK of over-parametrized fully-connected ReLU networks initialized at the EOC can evolve non-trivially during GD training if depth and width of the network are comparable. In particular, their result bounds the relative change of the diagonal elements of the NTK $\Theta(x,x)$ in the first GD step carried out on a single sample x above and below by an exponential function of the depth-to-width ratio λ. We generalize this result to different initializations with the following theorem proven in Appendix A.3:

Theorem 4.1 (GD step of the NTK)**.** *Consider a ReLU DNN from Theorem 3.2. A single GD update on a sample $(x,y) \in \mathcal{D}$ results in the following changes of the NTK:*

1. *In the **chaotic phase** ($a := \sigma_w^2/2 > 1$), the changes to the NTK value are infinite in the limit for a constant learning rate $\eta \in \mathbb{R}$:*

$$\frac{\mathbb{E}[\Delta\Theta(x,x)]}{\mathbb{E}[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \infty. \tag{21}$$

2. *In the **ordered phase** ($a < 1$), the NTK stays constant in the limit:*

$$\frac{\mathbb{E}[\Delta\Theta(x,x)]}{\mathbb{E}[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0. \tag{22}$$

This result shows that deep networks can potentially behave according to the NTK theory during GD training only in case of initialization in the ordered phase. We refer to experiments in Seleznova & Kutyniok (2021), which confirm that the relative change of the NTK during training on MNIST is significant and grows with depth in the chaotic phase and at the EOC but not in the ordered phase. However, it is unclear how to generalize this result to realistic scenarios of DNN training, which include randomly selected batches of

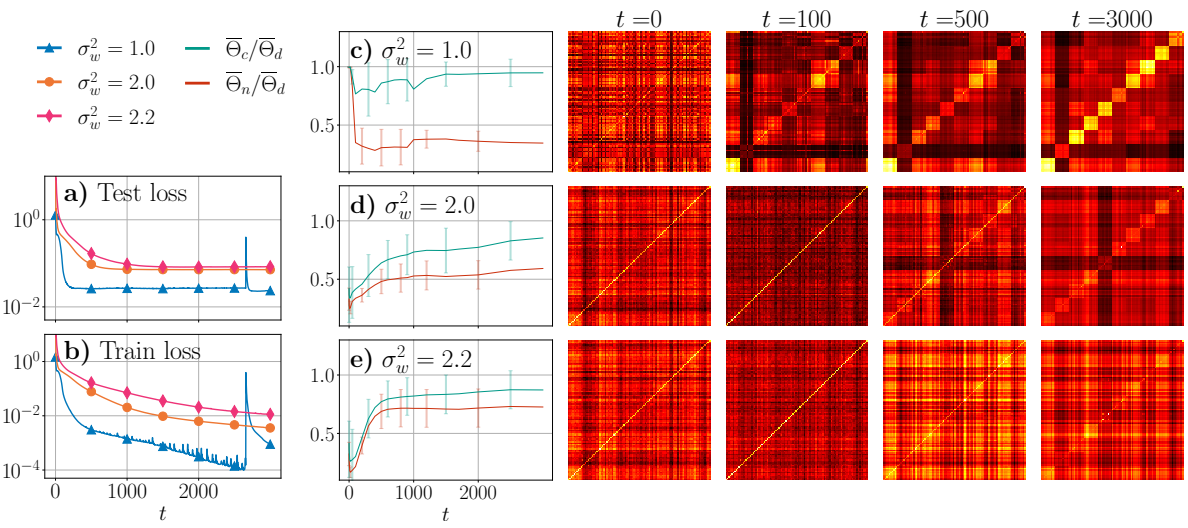


Figure 4. Structure of the NTK matrix in different stages of training for fully-connected ReLU DNNs with $L = 20$ and $M = 300$. The DNNs are initialized with $\sigma_w^2 \in \{1.0, 2.0, 2.2\}$ and trained on MNIST using Adam algorithm with learning rate 10^{-5}. Subplots **a)** and **b)** show the test and the train loss achieved by each DNN. Subplots **c)**, **d)** and **e)** characterize label-awareness of the NTK. Variables $\overline{\Theta}_d$, $\overline{\Theta}_c$ and $\overline{\Theta}_n$ are defined in (23). The heatmaps show the NTK matrix on MNIST subsample of size 100 at epoch $t \in \{0, 100, 500, 3000\}$. The subsample contains 10 elements of each class and is arranged so that consecutive diagonal blocks of size 10 contain pairwise NTK values on each class. The color range in the heatmaps is adjusted to include the interval between the maximal and the minimal values of the NTK in a given epoch, i.e. the colors correspond to different values for different epochs. Brighter colors indicate larger values.

arbitrary size and optimization algorithms beyond vanilla GD. Our experiments in the next subsection show that the NTK evolution is in general non-trivial even in the ordered phase.

Remark 3. Deep networks rescaled as in Chizat et al. (2019) can exhibit lazy training (with random NTK at initialization) in the chaotic phase only if the scaling parameter grows exponentially with depth L. We discuss the lazy training phenomenon and its effects on our results in Appendix B.3.

4.2. Changes of the NTK structure

The NTK at initialization is label-agnostic, i.e. its value on a pair $(x, \tilde{x})$ is independent of whether the labels of x and $\tilde{x}$ are the same or not. Clearly, label-agnostic features cannot provide an optimal representation system for an arbitrary task and many authors studied the benefits of adding label information to kernels (Cristianini et al., 2001; Gönen & Alpaydin, 2011; Tishby & Zaslavsky, 2015). In particular, Chen et al. (2020) argued that label-agnosticism can explain the performance gap between trained DNNs and the NTK and demonstrated that adding label-awareness improves the performance of the infinite-width NTK. Thus, it is important to characterize label-awareness of the empirical NTK and how the training process leads to it to understand the properties of DNNs.

We saw in Section 3.3 that the NTK at initialization has an approximately diagonal structure with the diagonal values larger than the non-diagonal ones. On the contrary, the "optimal kernel" for a classification task would be block-diagonal with blocks of larger values corresponding to samples of the same class. Thus, we expect the NTK to naturally change towards the block-diagonal structure during the training process. Our experiments in Figure 4 confirm this intuition in a simple setting of fully-connected ReLU networks trained on MNIST. Let us define the following variables that characterize label-awareness of the NTK matrix:

$$\begin{aligned}
\overline{\Theta}_d &:= \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \Theta(x, x), \\
\overline{\Theta}_c &:= \frac{1}{K} \sum_{k=1}^{K} \frac{1}{|\mathcal{X}_k|(|\mathcal{X}_k| - 1|)} \sum_{\substack{x_i \neq x_j, \\ x_i, x_j \in \mathcal{X}_k}} \Theta(x_i, x_j), \\
\overline{\Theta}_n &:= \frac{1}{K} \sum_{k=1}^{K} \frac{1}{|\mathcal{X}_k|(|\mathcal{X}| - |\mathcal{X}_k|)} \sum_{\substack{x_i \in \mathcal{X}_k, \\ x_j \notin \mathcal{X}_k}} \Theta(x_i, x_j),
\end{aligned} \tag{23}$$

where $\mathcal{X} = \cup_{k=1}^{K} \mathcal{X}_k$ is the decomposition of the dataset $\mathcal{X}$ into K classes. Then $\overline{\Theta}_d$ is the mean diagonal value, $\overline{\Theta}_c$ is the mean value of the NTK on samples from the same class and $\overline{\Theta}_n$ is the mean value on samples from different classes. Figure 4 suggests that a larger gap between $\overline{\Theta}_n/\overline{\Theta}_d$ and $\overline{\Theta}_c/\overline{\Theta}_d$ may be related to better performance of DNNs. Moreover, the gap between 1 and the ratio $\overline{\Theta}_c/\overline{\Theta}_d$ may characterize overfitting. Therefore, we believe that the structure

of the NTK can be a proxy for generalization of DNNs even outside of the NTK regime. One can also see that the structure of the NTK changes more rapidly in the early stages of training, which is coherent with the conclusion in Fort et al. (2020) that useful features are mostly learned in the first epochs of training. Thus, dynamics of the NTK may provide information about the state of the training process.

5. Conclusions and future work

This paper adds to the line of research on the statistical properties of the NTK and the correspondence between finite-width DNNs and their infinite-width approximations. Our results in Section 3 precisely quantify variability of the NTK at initialization for a given fully-connected ReLU DNN and assess how well the kernel is approximated by its infinite-width limit. Combining our findings from Section 3 with the results on the GD update of the NTK in Section 4.1, we conclude that the NTK regime can approximate trained networks with non-trivial depth-to-width ratio only in the ordered phase. At the same time, the behavior of overparametrized DNNs outside of the NTK regime is very poorly understood so far. It is unclear how to characterize DNNs' training dynamics in the general case and what role the properties of the (random and dynamic) NTK play here. We make a step into this direction in Section 4.2 by demonstrating how the NTK acquires a block-diagonal structure during training. We believe that precisely characterizing the effects of this NTK structure on the generalization of DNNs is a promising direction for future work. In general, we hope to establish new connections between the NTK and other aspects of DNN training outside of the NTK regime.

Acknowledgements

G.K. acknowledges partial support by the NSF–Simons Research Collaboration on the Mathematical and Scientific Foundations of Deep Learning (MoDL) (NSF DMS 2031985) and DFG SPP 1798, KU 1446/27-2 and KU 1446/21-2.

References

Adlam, B. and Pennington, J. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *International Conference on Machine Learning*, pp. 74–84. PMLR, 2020.

Aitchison, L. Why bigger is not always better: on finite and infinite neural networks. In *International Conference on Machine Learning, ICML*, volume 119, pp. 156–164. PMLR, 2020.

Alemohammad, S., Wang, Z., Balestriero, R., and Baraniuk, R. G. The recurrent neural tangent kernel. In *International Conference on Learning Representations, ICLR*, 2021.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pp. 8139–8148, 2019.

Arora, S., Du, S. S., Li, Z., Salakhutdinov, R., Wang, R., and Yu, D. Harnessing the power of infinitely wide deep nets on small-data tasks. In *International Conference on Learning Representations, ICLR*, 2020.

Bai, Y., Krause, B., Wang, H., Xiong, C., and Socher, R. Taylorized training: Towards better approximation of neural network training at finite width. *CoRR*, abs/2002.04010, 2020. URL https://arxiv.org/abs/2002.04010.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Buchanan, S., Gilboa, D., and Wright, J. Deep networks and the multiple manifold problem. In *International Conference on Learning Representations, ICLR*, 2021.

Chen, S., He, H., and Su, W. J. Label-aware neural tangent kernel: Toward better generalization and local elasticity. In *Advances in Neural Information Processing Systems*, 2020.

Chizat, L., Oyallon, E., and Bach, F. R. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, pp. 2933–2943, 2019.

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., and Kandola, J. S. On kernel-target alignment. In *Advances in Neural Information Processing Systems*, pp. 367–373. MIT Press, 2001.

de G. Matthews, A. G., Hron, J., Rowland, M., Turner, R. E., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations, ICLR*, 2018.

Du, S. S., Hou, K., Salakhutdinov, R., Póczos, B., Wang, R., and Xu, K. Graph neural tangent kernel: Fusing graph neural networks with graph kernels. In *Advances in Neural Information Processing Systems*, pp. 5724–5734, 2019.

Fort, S., Dziugaite, G. K., Paul, M., Kharaghani, S., Roy, D. M., and Ganguli, S. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. In *Advances in Neural Information Processing Systems*, 2020.

Geiger, M., Jacot, A., Spigler, S., Gabriel, F., Sagun, L., d'Ascoli, S., Biroli, G., Hongler, C., and Wyart, M. Scaling description of generalization with number of parameters in deep learning. *CoRR*, abs/1901.01608, 2019. URL http://arxiv.org/abs/1901.01608.

Gönen, M. and Alpaydin, E. Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, 12:2211–2268, 2011.

Hanin, B. and Nica, M. Finite depth and width corrections to the neural tangent kernel. In *International Conference on Learning Representations, ICLR*, 2020.

Hayou, S., Doucet, A., and Rousseau, J. Mean-field behaviour of neural tangent kernel for deep neural networks. *CoRR*, abs/1905.13654, 2019. URL https://arxiv.org/abs/1905.13654.

Heek, J., Levskaya, A., Oliver, A., Ritter, M., Rondepierre, B., Steiner, A., and van Zee, M. Flax: A neural network library and ecosystem for JAX, 2020. URL http://github.com/google/flax.

Hu, Z. and Huang, H. On the random conjugate kernel and neural tangent kernel. In *International Conference on Machine Learning*, pp. 4359–4368. PMLR, 2021.

Huang, J. and Yau, H. Dynamics of deep neural networks and neural tangent hierarchy. In *International Conference on Machine Learning, ICML*, volume 119, pp. 4542–4551. PMLR, 2020.

Huang, K., Wang, Y., Tao, M., and Zhao, T. Why do deep residual networks generalize better than deep feedforward networks? — A neural tangent kernel perspective. In *Advances in Neural Information Processing System*, 2020.

Jacot, A., Hongler, C., and Gabriel, F. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pp. 8580–8589, 2018.

Karakida, R., Akaho, S., and Amari, S. Universal statistics of Fisher information in deep neural networks: Mean field approach. In *International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 89, pp. 1032–1041. PMLR, 2019.

Korotkov, N. E. and Korotkov, A. N. *Integrals Related to the Error Function*. CRC Press, 2020.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems*, pp. 8570–8581, 2019.

Lee, J., Schoenholz, S. S., Pennington, J., Adlam, B., Xiao, L., Novak, R., and Sohl-Dickstein, J. Finite versus infinite neural networks: an empirical study. In *Advances in Neural Information Processing Systems*, 2020.

Li, M. B., Nica, M., and Roy, D. M. The future is log-gaussian: ResNets and their infinite-depth-and-width limit at initialization. *CoRR*, abs/2106.04013, 2021. URL https://arxiv.org/abs/2106.04013.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. Exponential expressivity in deep neural networks through transientchaos. In *Advances in Neural Information Processing Systems*, pp. 3360–3368, 2016.

Schoenholz, S. S., Gilmer, J., Ganguli, S., and Sohl-Dickstein, J. Deep information propagation. In *International Conference on Learning Representations, ICLR*, 2017.

Seleznova, M. and Kutyniok, G. Analyzing finite neural networks: Can we trust neural tangent kernel theory? In *Conference on Mathematical and Scientific Machine Learning*, volume 145. PMLR, 2021.

Tirer, T., Bruna, J., and Giryes, R. Kernel-based smoothness analysis of residual networks. In *Conference on Mathematical and Scientific Machine Learning*, volume 145. PMLR, 2021.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop, ITW*, pp. 1–5. IEEE, 2015. doi: 10.1109/ITW.2015.7133169.

Wang, S., Yu, X., and Perdikaris, P. When and why PINNs fail to train: A neural tangent kernel perspective. *J. Comput. Phys.*, 449:110768, 2022. doi: 10.1016/j.jcp.2021.110768.

Xiao, L., Pennington, J., and Schoenholz, S. Disentangling trainability and generalization in deep neural networks. In *International Conference on Machine Learning*, pp. 10462–10472. PMLR, 2020.

Yang, G. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *CoRR*, abs/1902.04760, 2019. URL http://arxiv.org/abs/1902.04760.

Yang, G. Tensor programs II: Neural tangent kernel for any architecture. *CoRR*, abs/2006.14548, 2020. URL https://arxiv.org/abs/2006.14548.

Yang, G. and Hu, E. J. Tensor programs IV: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*, volume 139, pp. 11727–11737. PMLR, 2021.

Yang, G. and Schoenholz, S. S. Mean field residual networks: On the edge of chaos. In *Advances in Neural Information Processing Systems*, pp. 7103–7114, 2017.

A. Proofs

A.1. Variability of the NTK at initialization

Lemma A.1 (Forward-propagation of variance)**.** *Consider a fully-connected DNN defined in (3) initialized as in (6). The activation function in the hidden layers is ReLU, i.e. $\phi(x) = x\mathbb{1}\{x > 0\}$. Assume further that the biases are initialized to zero, i.e. $\sigma_b = 0$. Then the following holds for the ratios of the activation norms in consecutive layers of the network, denoted $\mathcal{N}_x^\ell := \|\mathbf{x}^\ell\|^2/\|\mathbf{x}^{\ell-1}\|^2$, $\ell = 1, \dots, L-1$:*

$$\mathbb{E}[\mathcal{N}_x^\ell] = \frac{\sigma_w^2}{2}\frac{n_\ell}{n_{\ell-1}}, \quad \mathbb{E}[(\mathcal{N}_x^\ell)^2] = \Big(\frac{\sigma_w^2}{2}\Big)^2\Big(\frac{n_\ell}{n_{\ell-1}}\Big)^2\Big(1+\frac{5}{n_\ell}\Big). \tag{24}$$

$$\frac{\mathcal{N}_x^\ell - \mathbb{E}[\mathcal{N}_x^\ell]}{\sqrt{\mathbb{V}[\mathcal{N}_x^\ell]}} \xrightarrow[n_\ell\to\infty]{d} \mathcal{N}(0,1), \tag{25}$$

where $\mathcal{N}(0,1)$ is the standard normal distribution. Moreover, random variables $\{\mathcal{N}_x^\ell\}_{\ell=0,\dots,L-1}$ are mutually independent.

Proof. The squared norm of the activation vector in layer ℓ is given by

$$\|\mathbf{x}^\ell\|^2 = \sum_{i=1}^{n_\ell} \phi^2(\mathbf{W}_{i\cdot}^\ell \mathbf{x}^{\ell-1} + b_i^\ell)$$

Here $\mathbf{x}^{\ell-1}$ depends only on $\{(\mathbf{W}^j, \mathbf{b}^j)\}_{j=1,\dots\ell-1}$, therefore $\mathbf{x}^{\ell-1}$ is independent of $(\mathbf{W}^\ell, \mathbf{b}^\ell)$. Since elements of $\mathbf{W}^\ell$ are i.i.d Gaussian, the distribution of $\mathbf{W}^\ell\mathbf{x}^{\ell-1}$ depends only on the norm of $\mathbf{x}^{\ell-1}$ and not on the direction. Then we can write the following equalities in distribution:

$$\mathbf{W}_{i\cdot}^\ell \mathbf{x}^{\ell-1} = \sqrt{\frac{\sigma_w^2}{n_{\ell-1}}}\|\mathbf{x}^{\ell-1}\|\mathcal{U}_i^\ell,$$

$$\phi^2\Big(\mathbf{W}_{i\cdot}^\ell \mathbf{x}^{\ell-1} + b_i^\ell\Big) = \Big(\sqrt{\frac{\sigma_w^2}{n_{\ell-1}}}\|\mathbf{x}^{\ell-1}\| + \sigma_b\Big)^2 \phi^2(\mathcal{U}_i^\ell),$$

where we introduced i.i.d random variables $\mathcal{U}_i^\ell = \Big\langle \sqrt{\frac{n_{\ell-1}}{\sigma_w^2}}(\mathbf{W}_{i\cdot}^\ell)^T, \frac{\mathbf{x}^{\ell-1}}{\|\mathbf{x}^{\ell-1}\|}\Big\rangle \sim \mathcal{N}(0,1)$, $i = 1, \dots, n_\ell$, which are independent of $\mathbf{x}^{\ell-1}$, and used the fact that $\phi(\alpha x) = \alpha\phi(x)$ for $\alpha \in \mathbb{R}^+$. Therefore for the norm of the activation vector we have the following:

$$\|\mathbf{x}^\ell\|^2 = \sum_{i=1}^{n_\ell} \phi^2(\mathbf{W}_{i\cdot}^\ell \mathbf{x}^{\ell-1} + b_i^\ell) = \Big(\sqrt{\frac{\sigma_w^2}{n_{\ell-1}}}\|\mathbf{x}^{\ell-1}\| + \sigma_b\Big)^2 \sum_{i=1}^{n_\ell} \phi^2(\mathcal{U}_i^\ell),$$

where only the first bracket depends on $\mathbf{x}^{\ell-1}$. Then in case of zero biases, i.e. $\sigma_b = 0$, for the ratio between the norms of consecutive activation vectors we have

$$\mathcal{N}_x^\ell = \frac{\sigma_w^2}{n_{\ell-1}} \sum_{i=1}^{n_\ell} \phi^2(\mathcal{U}_i^\ell),$$

where the variables $\mathcal{U}_i^\ell, i = 1, \dots, n_\ell$ depend only on the weights in the given layer $\mathbf{W}^\ell$. Then the ratios $\mathcal{N}_x^\ell$ in different layers are independent and we can obtain the desired moments of $\mathcal{N}_x^\ell$ as follows:

$$\mathbb{E}[\mathcal{N}_x^\ell] = \frac{\sigma_w^2}{n_{\ell-1}} \sum_{i=1}^{n_\ell} \mathbb{E}[\phi^2(\mathcal{U}_i^\ell)] = \frac{\sigma_w^2}{2}\frac{n_\ell}{n_{\ell-1}},$$

$$\mathbb{E}[(\mathcal{N}_x^\ell)^2] = \Big(\frac{\sigma_w^2}{n_{\ell-1}}\Big)^2 \sum_{i=1}^{n_\ell} \mathbb{V}[\phi^2(\mathcal{U}_i^\ell)] + \mathbb{E}^2[(\mathcal{N}_x^\ell)] = \Big(\frac{\sigma_w^2}{2}\Big)^2\Big(\frac{n_\ell}{n_{\ell-1}}\Big)^2\Big(1+\frac{5}{n_\ell}\Big),$$

where we used the moments of variables $\phi(\mathcal{U}_i)$, which can be calculated by integration:

$$\mathbb{E}[\phi^2(\mathcal{U}_i^\ell)] = \frac{1}{2}, \quad \mathbb{V}[\phi^2(\mathcal{U}_i^\ell)] = \frac{5}{4}, \quad i = 1, \dots, n_\ell.$$

Moreover, by the central limit theorem we have

$$\frac{\mathcal{N}_x^\ell - \mathbb{E}[\mathcal{N}_x^\ell]}{\sqrt{\mathbb{V}[\mathcal{N}_x^\ell]}} = \frac{2\Big(\frac{1}{n_\ell}\sum_{i=1}^{n_\ell}\phi^2(\mathcal{U}_i^\ell) - \frac{1}{2}\Big)}{\sqrt{5/n_\ell}} \xrightarrow[n_\ell\to\infty]{d} \mathcal{N}(0,1).$$

□

Lemma A.2 (Backpropagation of variance)**.** *Consider the same setting as in Lemma A.1. Then the following holds for the ratios of norms of backpropagated errors (defined in (7)) in consecutive layers, denoted $\mathcal{N}_\delta^\ell := \|\boldsymbol{\delta}^\ell\|^2/\|\boldsymbol{\delta}^{\ell+1}\|^2$, $\ell = 1,\dots,L-1$:*

$$\mathbb{E}[\mathcal{N}_\delta^\ell] = \frac{\sigma_w^2}{2}, \quad \mathbb{E}[(\mathcal{N}_x^\ell)^2] = \Big(\frac{\sigma_w^2}{2}\Big)^2\Big(1+\frac{5}{n_\ell}\Big). \tag{26}$$

$$\frac{\mathcal{N}_\delta^\ell - \mathbb{E}[\mathcal{N}_\delta^\ell]}{\sqrt{\mathbb{V}[\mathcal{N}_\delta^\ell]}} \xrightarrow[n_\ell\to\infty]{d} \mathcal{N}(0,1), \tag{27}$$

where $\mathcal{N}(0,1)$ is the standard normal distribution.

Proof. The recursive formula for the backpropagated errors is given by

$$\boldsymbol{\delta}_i^\ell = \phi'(\mathbf{h}_i^\ell)\sum_{j=1}^{n_{\ell+1}}\mathbf{W}_{ji}^{\ell+1}\boldsymbol{\delta}_j^{\ell+1} = \phi'(\mathbf{h}_i^\ell)(\mathbf{W}^{\ell+1})_{i\cdot}^T\boldsymbol{\delta}^{\ell+1}.$$

Then, in the same way as in Lemma A.1, we have the following for the squared norm of $\boldsymbol{\delta}^\ell$:

$$\|\boldsymbol{\delta}^\ell\|^2 = \sum_{i=1}^{n_\ell}(\phi'(\mathbf{h}_i^\ell))^2\Big((\mathbf{W}^{\ell+1})_{i\cdot}^T\boldsymbol{\delta}^{\ell+1}\Big)^2 = \frac{\sigma_w^2}{n_\ell}\|\boldsymbol{\delta}^{\ell+1}\|^2\sum_{i=1}^{n_\ell}(\phi'(\mathbf{h}_i^\ell))^2(\mathcal{V}_i^{\ell+1})^2,$$

where we introduced i.i.d. random variables $\mathcal{V}_i^{\ell+1} = \Big\langle\sqrt{\frac{n_\ell}{\sigma_w^2}}\mathbf{W}_{\cdot i}^{\ell+1}, \frac{\boldsymbol{\delta}^{\ell+1}}{\|\boldsymbol{\delta}^{\ell+1}\|}\Big\rangle \sim \mathcal{N}(0,1)$, $i = 1,\dots,n_\ell$, which are independent of $\boldsymbol{\delta}^{\ell+1}$. One can also see that $\phi'(\mathbf{h}^\ell)$ can only depend on $\{(\mathbf{W}^j,\mathbf{b}^j)\}_{j=1,\dots\ell}$, therefore it is independent of $\|\boldsymbol{\delta}\|^{\ell+1}$ and of $\mathcal{V}_i^{\ell+1}, i = 1,\dots,n_\ell$. Moreover, $\phi'(\mathbf{h}_i^\ell) = \phi'(\mathbf{W}_{i\cdot}^\ell\mathbf{x}^{\ell-1}) = \phi'(\mathcal{U}_i^\ell)$ for all $i = 1,\dots,n_\ell$, therefore $\phi'(\mathbf{h}^\ell)$ depends only on $\mathbf{W}^\ell$. Then we can write the following for the ratio of interest and its moments:

$$\mathcal{N}_\delta^\ell = \frac{\sigma_w^2}{n_\ell}\sum_{i=1}^{n_\ell}\phi'(\mathcal{U}_i^\ell)(\mathcal{V}_i^{\ell+1})^2,$$

$$\mathbb{E}[\mathcal{N}_\delta^\ell] = \frac{\sigma_w^2}{2}, \qquad \mathbb{E}[(\mathcal{N}_\delta^\ell)^2] = \Big(\frac{\sigma_w^2}{2}\Big)^2\Big(1+\frac{5}{n_\ell}\Big),$$

where we calculated the moments of the summands as

$$\mathbb{E}[(\phi'(\mathbf{h}_i^\ell))^2(\mathcal{V}_i^2)^2] = \mathbb{E}[(\phi'(\mathbf{h}_i^\ell))^2]\mathbb{E}[\mathcal{V}_i^2] = \frac{1}{2}, \quad \mathbb{V}[(\phi'(\mathbf{h}_i^\ell))^2\mathcal{V}_i^2] = \mathbb{E}[(\phi'(\mathbf{h}_i^\ell))^4]\mathbb{E}[\mathcal{V}_i^4] - \frac{1}{4} = \frac{5}{4}.$$

Here we used that, in case of ReLU activation, $\phi'(\mathbf{h}_i^\ell), i = 1,\dots,n_\ell$ are Bernoulli variables with probability of 1 and 0 equal to $1/2$, since $\mathbf{h}^\ell$ is symmetric around zero. Therefore, $\mathbb{E}[(\phi'(\mathbf{h}_i^\ell))^2] = \mathbb{E}[(\phi'(\mathbf{h}_i^\ell))^4] = 1/2$.

Same as in Lemma A.1, the limiting distribution of $\mathcal{N}_\delta^\ell$ is given by the central limit theorem.

□

As we note in Section 3.4, many papers that study the NTK adopt the following assumption:

Assumption A.3 (Gradient independence assumption (GIA))**.** Matrix $(\mathbf{W}^\ell)^T$ in backpropagation equations (7) and matrix $\mathbf{W}^\ell$ in forward-propagation equations (3) are independent for all $\ell \in \{1,\dots,L\}$.

This assumption is of course not true; however, the products $(\mathbf{W}^\ell)^T_{i\cdot}\mathbf{x} = \sum_{k=1}^{n_\ell} \mathbf{W}^\ell_{ki}\mathbf{x}_k$ and $\mathbf{W}^\ell_{j\cdot}\mathbf{x} = \sum_{k=1}^{n_{\ell-1}} \mathbf{W}^\ell_{jk}\mathbf{x}_k$ are only dependent through the single summand containing $\mathbf{W}^\ell_{ij}$. Thus, the correlations caused by this dependence are of order $O(1/M)$ and can be disregarded in the infinite-width limit. However, in our case of the infinite-depth-and-width limit terms of order $O(1/M)$ can have a non-trivial impact on the computations. Therefore, we calculate the effects of the dependence between the forward-propagated chain and the backpropagated chain in the following lemma.

Lemma A.4 (Gradient independence assumption (GIA))**.** *Consider the same setting as in Lemma A.1. Then the following statements hold:*

1. *GIA does not change the expectation of $\|\boldsymbol{\delta}^\ell\|^2/\|\boldsymbol{\delta}^{\ell+k+1}\|^2$:*

$$\mathbb{E}\Big[\prod_{p=0}^{k} \mathcal{N}_\delta^{\ell+p}\Big] = \prod_{p=0}^{k} \mathbb{E}[\mathcal{N}_\delta^{\ell+p}] \tag{28}$$

2. *GIA changes the expectation of $\|\boldsymbol{\delta}^\ell\|^2/\|\boldsymbol{\delta}^{\ell+k+1}\|^2 \cdot \|\mathbf{x}^{\ell+k}\|^2/\|\mathbf{x}^{\ell-1}\|^2$ by a term that has a non-trivial depth-and-width limit where $M \to \infty, L \to \infty, L/M \to \lambda \in \mathbb{R}$. In particular, we have:*

$$\mathbb{E}\Big[\prod_{p=0}^{k} \mathcal{N}_\delta^{\ell+p}\mathcal{N}_x^{\ell+p}\Big] = \prod_{p=0}^{k} \mathbb{E}[\mathcal{N}_\delta^{\ell+p}]\mathbb{E}[\mathcal{N}_x^{\ell+p}]\Big(1 + \frac{1}{n_{\ell+p}} + O\Big(\frac{1}{M^{3/2}}\Big)\Big), \tag{29}$$

   where $n_\ell = \alpha_\ell M, \alpha_\ell \in \mathbb{R}, \ell = 1, \dots, L-1$

3. *GIA does not change the expectation of $(\|\boldsymbol{\delta}^\ell\|^2/\|\boldsymbol{\delta}^{\ell+k+1}\|^2)^2$ in the infinite-depth-and-width limit where $M \to \infty, L \to \infty, L/M \to \lambda \in \mathbb{R}$. In particular, we have:*

$$\mathbb{E}\Big[\prod_{p=0}^{k} (\mathcal{N}_\delta^{\ell+p})^2\Big] = \prod_{p=0}^{k} \mathbb{E}[(\mathcal{N}_\delta^{\ell+p})^2]\Big(1 + O\Big(\frac{1}{M^{3/2}}\Big)\Big), \tag{30}$$

   where $n_\ell = \alpha_\ell M, \alpha_\ell \in \mathbb{R}, \ell = 1, \dots, L-1$:

Proof. In Lemmas A.1 and A.2 we derived the following equations for $\mathcal{N}_\delta^\ell$ and $\mathcal{N}_x^\ell$:

$$\mathcal{N}_\delta^\ell = \frac{\sigma_w^2}{n_\ell}\sum_{i=1}^{n_\ell}(\phi'(\mathbf{h}_i^\ell))^2(\mathcal{V}_i^{\ell+1})^2 = \frac{\sigma_w^2}{n_\ell}\sum_{i=1}^{n_\ell}(\phi'(\mathcal{U}_i^\ell))^2(\mathcal{V}_i^{\ell+1})^2,$$
$$\mathcal{N}_x^\ell = \frac{\sigma_w^2}{n_{\ell-1}}\sum_{i=1}^{n_\ell}\phi^2(\mathcal{U}_i^\ell),$$

where $\mathcal{U}_i^\ell$ depends only on the i-th row of the weights matrix $\mathbf{W}^\ell_{i\cdot}$ and $\mathcal{V}_j^\ell$ depends only on j-th column of the same matrix $\mathbf{W}^\ell_{\cdot j}$ for $i = 1, \dots, n_\ell, j = 1, \dots, n_{\ell-1}, \ell = 1, \dots, L-1$. Therefore, variables $\mathcal{U}_i^\ell$ and $\mathcal{V}_j^\ell$ are only dependent through the single weight $\mathbf{W}^\ell_{ij}$, which nevertheless makes $\mathcal{N}_\delta^\ell$ and $\mathcal{N}_\delta^{\ell+1}$ dependent for any $\ell = 1, \dots, L-2$. One can also see that $\mathcal{N}_\delta^\ell$ and $\mathcal{N}_x^\ell$ are dependent through $\{\mathcal{U}_i^\ell\}_{i=1,\dots,n_\ell}$. The objective of this lemma is to determine the effects of these weak dependencies on the expectation of products that appear in the NTK.

Part 1. We first consider the product of ratios of the backpropagated errors:

$$\begin{aligned}
\mathbb{E}\Big[\prod_{p=0}^{k} \mathcal{N}_\delta^{\ell+p}\Big] &= \prod_{p=0}^{k}\frac{\sigma_w^2}{n_{\ell+p}}\sum_{i_0=1}^{n_\ell}\cdots\sum_{i_k=1}^{n_{\ell+k}}\mathbb{E}[\phi'(\mathcal{U}_{i_0}^\ell)(\mathcal{V}_{i_0}^{\ell+1})^2\phi'(\mathcal{U}_{i_1}^{\ell+1})(\mathcal{V}_{i_1}^{\ell+2})^2\cdot\dots\cdot\phi'(\mathcal{U}_{i_k}^{\ell+k})(\mathcal{V}_{i_k}^{\ell+k+1})^2] \\
&= \prod_{p=0}^{k}\frac{\sigma_w^2}{n_{\ell+p}}\sum_{i_0=1}^{n_\ell}\cdots\sum_{i_k=1}^{n_{\ell+k}}\mathbb{E}[\phi'(\mathcal{U}_{i_0}^\ell)]\mathbb{E}[(\mathcal{V}_{i_k}^{\ell+k+1})^2]\prod_{p=1}^{k}\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})] \\
&= \frac{1}{2}\prod_{p=0}^{k}\frac{\sigma_w^2}{n_{\ell+p}}\sum_{i_0=1}^{n_\ell}\cdots\sum_{i_k=1}^{n_{\ell+k}}\prod_{p=1}^{k}\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})].
\end{aligned}$$

As $\mathcal{U}_{i_p}^{\ell+p}$ that $\mathcal{V}_{i_{p-1}}^{\ell+p}$ depend only through $\mathbf{W}_{i_p i_{p-1}}^{\ell+p}$, we can condition the expectation of their product as follows:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})] = \mathbb{E}\big[\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \mid \mathbf{W}_{i_p i_{p-1}}^{\ell+p}] \cdot \mathbb{E}[\phi'(\mathcal{U}_{i_p}^{\ell+p}) \mid \mathbf{W}_{i_p i_{p-1}}^{\ell+p}]\big].$$

To simplify the notation, let us denote $w_{i_p i_{p-1}} := \sqrt{\frac{n_{\ell+p-1}}{\sigma_w^2}}\mathbf{W}_{i_p i_{p-1}}^{\ell+p} \sim \mathcal{N}(0,1)$, $a_j := \mathbf{x}_j^{\ell+p}/\|\mathbf{x}^{\ell+p}\|$ and $b_k := \boldsymbol{\delta}_k^{\ell+p}/\|\boldsymbol{\delta}^{\ell+p}\|$. Then we have $\mathcal{V}_{i_{p-1}}^{\ell+p} = \sum_{k=1}^{n_{\ell+p}} w_{ki_{p-1}}b_k = w_{i_p i_{p-1}}b_{i_p} + \sum_{k\neq i_p} w_{ki_{p-1}}b_k$ and $\mathcal{U}_{i_p}^{\ell+p} = \sum_{j=1}^{n_{\ell+p-1}} w_{ji_{p-1}}a_j = w_{i_p i_{p-1}}a_{i_{p-1}} + \sum_{j\neq i_{p-1}} w_{i_p j}a_j$. We can then open the conditional expectations:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \mid \mathbf{W}_{i_p i_{p-1}}^{\ell+p}] = w_{i_p i_{p-1}}^2 b_{i_p}^2 + \mathbb{E}[(\sum_{k\neq i_p} w_{ki_{p-1}}b_k)^2] = 1 - b_{i_p}^2(1 - w_{i_p i_{p-1}}^2),$$

$$\mathbb{E}[\phi'(\mathcal{U}_{i_p}^{\ell+p}) \mid \mathbf{W}_{i_p i_{p-1}}^{\ell+p}] = \mathbb{P}[\sum_{j\neq i_{p-1}} w_{i_p j}b_k > -w_{i_p i_{p-1}}a_{i_{p-1}}] = \Phi\Big(\frac{w_{i_p i_{p-1}}a_{i_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2}}\Big),$$

where $\Phi(\cdot)$ is the CDF of the standard normal distribution. Here we used that $\sum_{k\neq i_p} w_{ki_{p-1}}b_k \sim \mathcal{N}(0, 1-b_{i_p}^2)$ and $\sum_{j\neq i_{p-1}} w_{i_p j}a_j \sim \mathcal{N}(0, 1-a_{i_{p-1}}^2)$. Then we have:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})] = (1-b_{i_p}^2)\mathbb{E}[\Phi(A\cdot w_{i_p i_{p-1}})] + b_{i_p}^2\mathbb{E}[w_{i_p i_{p-1}}^2\Phi(A\cdot w_{i_p i_{p-1}})] = \frac{1}{2},$$

where we used the following integrals:

$$\mathbb{E}[\Phi(A\cdot w_{i_p i_{p-1}})] = \frac{1}{2} + \frac{1}{2}\int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi}}\operatorname{erf}\Big(\frac{A}{\sqrt{2}}w_{i_p i_{p-1}}\Big)\exp\Big(-\frac{w_{i_p i_{p-1}}^2}{2}\Big)dw_{i_p i_{p-1}} = \frac{1}{2},$$

$$\begin{aligned}\mathbb{E}[w_{i_p i_{p-1}}^2\Phi(A\cdot w_{i_p i_{p-1}})] &= \frac{1}{2}\mathbb{E}[w_{i_p i_{p-1}}^2] \\ &\quad + \frac{1}{2}\int_{-\infty}^{\infty}\frac{1}{\sqrt{2\pi}}w_{i_p i_{p-1}}^2\operatorname{erf}\Big(\frac{A}{\sqrt{2}}w_{i_p i_{p-1}}\Big)\exp\Big(-\frac{w_{i_p i_{p-1}}^2}{2}\Big)dw_{i_p i_{p-1}} = \frac{1}{2}.\end{aligned}$$

Thus, the expectation of the product of the ratios of backpropagated errors is exactly equal to the product of their expectations:

$$\mathbb{E}\Big[\prod_{p=0}^{k}\mathcal{N}_\delta^{\ell+p}\Big] = \frac{1}{2}\prod_{p=0}^{k}\frac{\sigma_w^2}{n_{\ell+p}}\sum_{i_0=1}^{n_\ell}\cdots\sum_{i_k=1}^{n_{\ell+k}}\frac{1}{2^k} = \Big(\frac{\sigma_w^2}{2}\Big)^{k+1} = \prod_{p=0}^{k}\mathbb{E}[\mathcal{N}_\delta^{\ell+p}],$$

which completes the proof of the first statement.

Part 2. We now consider the expectation of products of the activations' ratios and the backpropagated errors' ratios for the same layers. The product in a single layer is given by:

$$\mathcal{N}_\delta^\ell\mathcal{N}_x^\ell = \frac{\sigma_w^2}{n_{\ell-1}}\frac{\sigma_w^2}{n_\ell}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_\ell}\phi^2(\mathcal{U}_j^\ell)(\phi'(\mathcal{U}_i^\ell))^2(\mathcal{V}_i^{\ell+1})^2 = \frac{\sigma_w^2}{n_{\ell-1}}\frac{\sigma_w^2}{n_\ell}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_\ell}\phi'(\mathcal{U}_i^\ell)\phi'(\mathcal{U}_j^\ell)(\mathcal{U}_j^\ell)^2(\mathcal{V}_i^{\ell+1})^2,$$

where we noticed that $\phi(\mathcal{U}_i^\ell)\phi'(\mathcal{U}_i^\ell) = \phi(\mathcal{U}_i^\ell) = \mathcal{U}_i^\ell\phi'(\mathcal{U}_i^\ell)$. Then for the product involving multiple layers we have:

$$\begin{aligned}\prod_{p=0}^{k}\mathcal{N}_\delta^{\ell+p}\mathcal{N}_x^{\ell+p} = \frac{\sigma_w^2}{n_{\ell-1}}\frac{\sigma_w^2}{n_{\ell+k}}\prod_{j=0}^{k-1}\Big(\frac{\sigma_w^2}{n_{\ell+j}}\Big)^2\sum_{i_0=1}^{n_\ell}\sum_{j_0=1}^{n_\ell}\cdots \\ \cdots\sum_{i_k=1}^{n_{\ell+k}}\sum_{j_k=1}^{n_{\ell+k}}\phi'(\mathcal{U}_{i_0}^\ell)\phi'(\mathcal{U}_{j_0}^\ell)(\mathcal{U}_{j_0}^\ell)^2(\mathcal{V}_{i_0}^{\ell+1})^2\ldots\phi'(\mathcal{U}_{i_k}^{\ell+k})\phi'(\mathcal{U}_{j_k}^{\ell+k})(\mathcal{U}_{j_k}^{\ell+k})^2(\mathcal{V}_{i_k}^{\ell+k+1})^2,\end{aligned}$$

And the expectation can be decomposed into products as follows:

$$\mathbb{E}\Big[\prod_{p=0}^{k}\mathcal{N}_\delta^{\ell+p}\mathcal{N}_x^{\ell+p}\Big] = \frac{\sigma_w^2}{n_{\ell-1}}\frac{\sigma_w^2}{n_{\ell+k}}\prod_{j=0}^{k-1}\Big(\frac{\sigma_w^2}{n_{\ell+j}}\Big)^2\sum_{i_0=1}^{n_\ell}\sum_{j_0=1}^{n_\ell}\cdots$$
$$\cdots\sum_{i_k=1}^{n_{\ell+k}}\sum_{j_k=1}^{n_{\ell+k}}\mathbb{E}[\phi'(\mathcal{U}_{i_0}^\ell)\phi'(\mathcal{U}_{j_0}^\ell)(\mathcal{U}_{j_0}^\ell)^2]\mathbb{E}[(\mathcal{V}_{i_k}^{\ell+k+1})^2]\cdot\prod_{p=1}^{k}\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})(\mathcal{U}_{j_p}^{\ell+p})^2]$$
$$= \frac{n_\ell}{4}\Big(1+\frac{1}{n_\ell}\Big)\frac{\sigma_w^2}{n_{\ell-1}}\frac{\sigma_w^2}{n_{\ell+k}}\prod_{j=0}^{k-1}\Big(\frac{\sigma_w^2}{n_{\ell+j}}\Big)^2\sum_{i_0=1}^{n_\ell}\cdots\sum_{i_k=1}^{n_{\ell+k}}\sum_{j_k=1}^{n_{\ell+k}}\prod_{p=1}^{k}\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})(\mathcal{U}_{j_p}^{\ell+p})^2],$$

where we used that $\mathbb{E}[(\mathcal{V}_{i_k}^{l+k+1})^2]=1$ and $\sum_{j_0=1}^{n_\ell}\mathbb{E}[\phi'(\mathcal{U}_{i_0}^\ell)\phi'(\mathcal{U}_{j_0}^\ell)(\mathcal{U}_{j_0}^\ell)^2] = \frac{1}{4}(n_\ell-1)+\frac{1}{2} = \frac{n_\ell}{4}\Big(1+\frac{1}{n_\ell}\Big)$. If $j_p\neq i_p$, we also know that the terms in $\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})(\mathcal{U}_{j_p}^{\ell+p})^2]$ are only dependent through $\{\mathbf{W}_{i_pi_{p-1}}^{\ell+p},\mathbf{W}_{j_p,i_{p-1}}^{\ell+p}\}$. Same as in Part 1, we can condition the product on these weights:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})(\mathcal{U}_{j_p}^{\ell+p})^2] = \mathbb{E}\big[\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\mid w_{i_pi_{p-1}},w_{j_pi_{p-1}}]\cdot\mathbb{E}[\phi'(\mathcal{U}_{i_p}^{\ell+p})\mid w_{i_pi_{p-1}}]\cdot$$
$$\cdot\,\mathbb{E}[\phi'(\mathcal{U}_{j_p}^{\ell+p})\mid w_{j_pi_{p-1}}]\cdot\mathbb{E}[(\mathcal{U}_{j_p}^{\ell+p})^2\mid w_{j_pi_{p-1}}]\big].$$

And we can again write the conditional expectations in case $j_p\neq i_p$ as follows:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\mid w_{i_pi_{p-1}},w_{j_pi_{p-1}}] = 1-b_{i_p}^2(1-w_{i_pi_{p-1}}^2)-b_{j_p}^2(1-w_{j_pi_{p-1}}^2)+2b_{i_p}b_{j_p}w_{i_pi_{p-1}}w_{j_pi_{p-1}},$$
$$\mathbb{E}[\phi'(\mathcal{U}_{i_p}^{\ell+p})\mid w_{i_pi_{p-1}}] = \Phi\Big(\frac{w_{i_pi_{p-1}}a_{i_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2}}\Big),$$
$$\mathbb{E}[\phi'(\mathcal{U}_{j_p}^{\ell+p})\mid w_{j_pi_{p-1}}] = \Phi\Big(\frac{w_{j_pi_{p-1}}a_{i_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2}}\Big),$$
$$\mathbb{E}[(\mathcal{U}_{j_p}^{\ell+p})^2\mid w_{j_pi_{p-1}}] = 1-a_{i_{p-1}}^2(1-w_{j_pi_{p-1}}^2).$$

To calculate the expectation of the product here we will need to use that $\mathbb{E}[\Phi(A\cdot w)] = \mathbb{E}[w^2\Phi(A\cdot w)] = \frac{1}{2}$, which we already computed in Part 1. One can also easily see that $\mathbb{E}[w^4\Phi(A\cdot w)] = \frac{3}{2}$. The other expectations involved in the product can be calculated as follows:

$$\mathbb{E}[w\Phi(A\cdot w)] = \frac{1}{2\sqrt{2\pi}}\int_{-\infty}^{\infty}w\,\mathrm{erf}\Big(\frac{A}{\sqrt{2}}w\Big)\exp\Big(-\frac{w^2}{2}\Big)dw = \frac{1}{\sqrt{2\pi}}\frac{A}{\sqrt{A^2+1}},$$
$$\mathbb{E}[w^3\Phi(A\cdot w)] = \frac{1}{2\sqrt{2\pi}}\int_{-\infty}^{\infty}w^3\,\mathrm{erf}\Big(\frac{A}{\sqrt{2}}w\Big)\exp\Big(-\frac{w^2}{2}\Big)dw = \frac{1}{\sqrt{2\pi}}\frac{A}{\sqrt{A^2+1}}\Big(2+\frac{1}{A^2+1}\Big).$$

Expressions for the integrals above can be found e.g. in (Korotkov & Korotkov, 2020). Using all the above expressions, we can obtain the following expression for the considered expectation in case $j_p\neq i_p$:

$$\begin{aligned}\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})(\mathcal{U}_{j_p}^{\ell+p})^2] &= \frac{1}{4}(1-b_{i_p}^2-b_{j_p}^2)(1-a_{i_{p-1}}^2)+\frac{1}{4}(b_{i_p}^2+b_{j_p}^2)(1-a_{i_{p-1}}^2)\\ &+\frac{1}{4}(1-b_{i_p}^2-b_{j_p}^2)a_{i_{p-1}}^2+\frac{1}{4}b_{i_p}^2a_{i_{p-1}}^2+\frac{3}{4}b_{j_p}^2a_{i_{p-1}}^2\\ &+2b_{i_p}b_{j_p}(1-a_{i_{p-1}}^2)\frac{4A^2}{A^2+1}+2b_{i_p}b_{j_p}a_{i_{p-1}}^2\frac{4A^2}{A^2+1}\Big(2+\frac{1}{A^2+1}\Big)\\ &= \frac{1}{4}+\frac{1}{2}b_{j_p}^2a_{i_{p-1}}^2+8b_{i_p}b_{j_p}a_{i_{p-1}}^2(1+2a_{i_{p-1}}^2-a_{i_{p-1}}^4)\end{aligned}$$

On the other hand, if $j_p=i_p$ we have $\phi'(\mathcal{U}_{i_p}^{l+p})\phi'(\mathcal{U}_{j_p}^{l+p})(\mathcal{U}_{j_p}^{l+p})^2 = \phi'(\mathcal{U}_{i_p}^{l+p}))(\mathcal{U}_{i_p}^{l+p})^2$ and therefore the expectation is given by:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})(\mathcal{U}_{i_p}^{\ell+p})^2] = \frac{1}{2}+a_{i_{p-1}}^2b_{i_p}^2.$$

We now notice that index j_p appears only in one expectation term in the product for each p. Therefore, we can sum over j_p independently for all p:

$$\sum_{j_p=1}^{n_{\ell+p}} \mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \phi'(\mathcal{U}_{i_p}^{\ell+p}) \phi'(\mathcal{U}_{j_p}^{\ell+p}) (\mathcal{U}_{j_p}^{\ell+p})^2] = \frac{n_{\ell+p}}{4}(1 + \frac{1}{n_{\ell+p}}) + \frac{1}{2} a_{i_{p-1}}^2 + \frac{1}{2} a_{i_{p-1}}^2 b_{i_p}^2 + \Big(\sum_{j_p=1}^{n_{\ell+p}} b_{j_p}\Big) 8 b_{i_p} a_{i_{p-1}}^2 (1 + 2a_{i_{p-1}}^2 - a_{i_{p-1}}^4).$$

On the other hand, we need to sum over i_{p-1} values sequentially over different values of p. First, we can calculate the following sum:

$$\sum_{i_{p-1}=1}^{n_{\ell+p-1}} \sum_{j_p=1}^{n_{\ell+p}} \mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \phi'(\mathcal{U}_{i_p}^{\ell+p}) \phi'(\mathcal{U}_{j_p}^{\ell+p}) (\mathcal{U}_{j_p}^{\ell+p})^2] = \frac{n_{\ell+p-1} n_{\ell+p}}{4}(1 + \frac{1}{n_{\ell+p}}) + \frac{1}{2} + \frac{1}{2} b_{i_p}^2 + \Big(\sum_{j_p=1}^{n_{\ell+p}} b_{j_p}\Big) 8 b_{i_p} (1 + 2 \sum_{i_{p-1}=1}^{n_{\ell+p-1}} a_{i_{p-1}}^4 - \sum_{i_{p-1}=1}^{n_{\ell+p-1}} a_{i_{p-1}}^6).$$

Then we can obtain the following bounds for the sum of b_{j_p} and $a_{i_{p-1}}$ given by Hölder's inequality:

$$\Big|\sum_{j_p=1}^{n_{\ell+p}} b_{j_p}\Big| \le \sum_{j_p=1}^{n_{\ell+p}} |b_{j_p}| = \|\mathbf{b}\|_1 \le \sqrt{n_{\ell+p}} \|\mathbf{b}^2\| = \sqrt{n_{\ell+p}},$$

$$0 \le 1 + 2 \sum_{i_{p-1}=1}^{n_{\ell+p-1}} a_{i_{p-1}}^4 - \sum_{i_{p-1}=1}^{n_{\ell+p-1}} a_{i_{p-1}}^6 \le 3,$$

$$|b_{i_p}| \le 1, \quad b_{i_p}^2 \le 1.$$

Therefore, we can rewrite the previous sum as

$$\sum_{i_{p-1}=1}^{n_{\ell+p-1}} \sum_{j_p=1}^{n_{\ell+p}} \mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \phi'(\mathcal{U}_{i_p}^{\ell+p}) \phi'(\mathcal{U}_{j_p}^{\ell+p}) (\mathcal{U}_{j_p}^{\ell+p})^2] = \frac{n_{\ell+p-1} n_{\ell+p}}{4} \Big(1 + \frac{1}{n_{\ell+p}} + O(\frac{1}{M^{3/2}})\Big)$$

Finally, for the expectation of the whole product we have:

$$\begin{aligned}
\mathbb{E}\Big[\prod_{p=0}^{k} \mathcal{N}_\delta^{\ell+p} \mathcal{N}_x^{\ell+p}\Big] &= \frac{\sigma_w^2}{n_{\ell-1}} \frac{\sigma_w^2}{n_{\ell+k}} \prod_{j=0}^{k-1} \Big(\frac{\sigma_w^2}{n_{\ell+j}}\Big)^2 \frac{n_\ell n_{\ell+k}}{4} \Big(1 + \frac{1}{n_\ell}\Big) \prod_{p=1}^{k} \frac{n_{\ell+p-1} n_{\ell+p}}{4} \Big(1 + \frac{1}{n_{\ell+p}} + O(\frac{1}{M^{3/2}})\Big) \\
&= \Big(\frac{\sigma_w^2}{2}\Big)^{2(k+1)} \frac{n_{\ell+k}}{n_{\ell-1}} \prod_{p=0}^{k} \Big(1 + \frac{1}{n_{\ell+p}} + O(\frac{1}{M^{3/2}})\Big) \\
&= \prod_{p=0}^{k} \mathbb{E}[\mathcal{N}_\delta^{\ell+p}] \mathbb{E}[\mathcal{N}_x^{\ell+p}] \Big(1 + \frac{1}{n_{\ell+p}} + O(\frac{1}{M^{3/2}})\Big)
\end{aligned}$$

Part 3. Finally, we consider the expectation of a product of squared ratios of the backpropagated errors. In a single layer we have:

$$(\mathcal{N}_\delta^\ell)^2 = \Big(\frac{\sigma_w^2}{n_\ell}\Big)^2 \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_\ell} \phi'(\mathcal{U}_i^\ell) \phi'(\mathcal{U}_j^\ell) (\mathcal{V}_i^{\ell+1})^2 (\mathcal{V}_j^{\ell+1})^2.$$

And for the product in multiple layers we have:

$$\prod_{p=0}^{k}(\mathcal{N}_\delta^{\ell+p})^2 = \prod_{p=0}^{k}\Big(\frac{\sigma_w^2}{n_{\ell+j}}\Big)^2 \sum_{i_0=1}^{n_\ell}\sum_{j_0=1}^{n_\ell}\cdots$$
$$\cdots\sum_{i_k=1}^{n_{\ell+k}}\sum_{j_k=1}^{n_{\ell+k}}\phi'(\mathcal{U}_{i_0}^{\ell})\phi'(\mathcal{U}_{j_0}^{\ell})(\mathcal{V}_{i_0}^{\ell+1})^2(\mathcal{V}_{j_0}^{\ell+1})^2\ldots\phi'(\mathcal{U}_{i_k}^{\ell+k})\phi'(\mathcal{U}_{j_k}^{\ell+k})(\mathcal{V}_{j_k}^{\ell+k+1})^2(\mathcal{V}_{i_k}^{\ell+k+1})^2,$$

$$\mathbb{E}\Big[\prod_{p=0}^{k}(\mathcal{N}_\delta^{\ell+p})^2\Big] = \prod_{p=0}^{k}\Big(\frac{\sigma_w^2}{n_{\ell+p}}\Big)^2 \sum_{i_0=1}^{n_\ell}\sum_{j_0=1}^{n_\ell}\cdots$$
$$\cdots\sum_{i_k=1}^{n_{\ell+k}}\sum_{j_k=1}^{n_{\ell+k}}\mathbb{E}[\phi'(\mathcal{U}_{i_0}^{\ell})\phi'(\mathcal{U}_{j_0}^{\ell})]\mathbb{E}[(\mathcal{V}_{i_k}^{\ell+k+1})^2(\mathcal{V}_{j_k}^{\ell+k+1})^2]\cdot\prod_{p=1}^{k}\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2(\mathcal{V}_{j_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})]$$

Here the expectations under product are more complicated since the variables in $\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2(\mathcal{V}_{j_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})]$ are dependent through $\{\mathbf{W}_{i_pi_{p-1}}^{\ell+p}, \mathbf{W}_{j_p,i_{p-1}}^{\ell+p}, \mathbf{W}_{i_pj_{p-1}}^{\ell+p}, \mathbf{W}_{j_pj_{p-1}}^{\ell+p}\}$. Nevertheless, we can still decompose the expectation as before into the following terms:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2(\mathcal{V}_{j_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})] = \mathbb{E}\big[\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \mid w_{i_pi_{p-1}}, w_{j_pi_{p-1}}]\cdot\mathbb{E}\big[\mathbb{E}[(\mathcal{V}_{j_{p-1}}^{\ell+p})^2 \mid w_{i_pj_{p-1}}, w_{j_pj_{p-1}}]\cdot$$
$$\mathbb{E}[\phi'(\mathcal{U}_{i_p}^{\ell+p}) \mid w_{i_pi_{p-1}}, w_{i_pj_{p-1}}]\cdot\mathbb{E}[\phi'(\mathcal{U}_{j_p}^{\ell+p}) \mid w_{j_pi_{p-1}}, w_{j_pj_{p-1}}]\big].$$

And each conditional expectations can again be calculated explicitly:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2 \mid w_{i_pi_{p-1}}, w_{j_pi_{p-1}}] = 1 - b_{i_p}^2(1-w_{i_pi_{p-1}}^2) - b_{j_p}^2(1-w_{j_pi_{p-1}}^2) + 2b_{i_p}b_{j_p}w_{i_pi_{p-1}}w_{j_pi_{p-1}},$$
$$\mathbb{E}[(\mathcal{V}_{j_{p-1}}^{\ell+p})^2 \mid w_{i_pj_{p-1}}, w_{j_pj_{p-1}}] = 1 - b_{i_p}^2(1-w_{i_pj_{p-1}}^2) - b_{j_p}^2(1-w_{j_pj_{p-1}}^2) + 2b_{i_p}b_{j_p}w_{i_pj_{p-1}}w_{j_pj_{p-1}},$$
$$\mathbb{E}[\phi'(\mathcal{U}_{i_p}^{\ell+p}) \mid w_{i_pi_{p-1}}, w_{i_pj_{p-1}}] = \Phi\Big(\frac{w_{i_pi_{p-1}}a_{i_{p-1}} + w_{i_pj_{p-1}}a_{j_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2-a_{j_{p-1}}^2}}\Big),$$
$$\mathbb{E}[\phi'(\mathcal{U}_{j_p}^{\ell+p}) \mid w_{j_pi_{p-1}}, w_{j_pj_{p-1}}] = \Phi\Big(\frac{w_{j_pi_{p-1}}a_{i_{p-1}} + w_{j_pj_{p-1}}a_{j_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2-a_{j_{p-1}}^2}}\Big).$$

We open the expectation using the following expressions, which, as before, are integrals involving the error function computed e.g. in (Korotkov & Korotkov, 2020):

$$\mathbb{E}[\Phi(A_iw_i + A_jw_j)] = \mathbb{E}[w_i^2\Phi(A_iw_i + A_jw_j)] = \frac{1}{2},$$
$$\mathbb{E}\left[w_i\Phi\Big(\frac{w_ia_{i_{p-1}} + w_ja_{j_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2-a_{j_{p-1}}^2}}\Big)\right] = \sqrt{\frac{1}{2\pi}}a_{i_{p-1}},$$
$$\mathbb{E}\left[w_iw_j\Phi\Big(\frac{w_ia_{i_{p-1}} + w_ja_{j_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2-a_{j_{p-1}}^2}}\Big)\right] = 0,$$
$$\mathbb{E}\left[w_i^2w_j^2\Phi\Big(\frac{w_ia_{i_{p-1}} + w_ja_{j_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2-a_{j_{p-1}}^2}}\Big)\right] = \frac{1}{2},$$
$$\mathbb{E}\left[w_iw_j^2\Phi\Big(\frac{w_ia_{i_{p-1}} + w_ja_{j_{p-1}}}{\sqrt{1-a_{i_{p-1}}^2-a_{j_{p-1}}^2}}\Big)\right] = \frac{1}{\sqrt{2\pi}}a_{i_{p-1}}(1-a_{j_{p-1}}^2).$$

Using all of the above, we get the following expression for the expectation in case $i_{p-1} \neq j_{p-1}$:

$$\mathbb{E}[(\mathcal{V}_{i_{p-1}}^{\ell+p})^2(\mathcal{V}_{j_{p-1}}^{\ell+p})^2\phi'(\mathcal{U}_{i_p}^{\ell+p})\phi'(\mathcal{U}_{j_p}^{\ell+p})] = \begin{cases}\frac{1}{2} & i_p = j_p,\\ \frac{1}{4} + \frac{1}{\pi}b_{i_p}b_{j_p}(a_{i_{p-1}}^2 + a_{j_{p-1}}^2 - 2a_{i_{p-1}}^2a_{j_{p-1}}^2(b_{i_p}^2 + b_{j_p}^2)) & i_p \neq j_p,\end{cases}$$

In case $i_{p-1} = j_{p-1}$, we have

$$\mathbb{E}[(\mathcal{V}^{\ell+p}_{i_{p-1}})^4 \mid w_{i_p i_{p-1}}, w_{j_p i_{p-1}}] = 1 - b^2_{i_p}(1 - w^2_{i_p i_{p-1}}) - b^2_{j_p}(1 - w^2_{j_p i_{p-1}}) + 2b_{i_p} b_{j_p} w_{i_p i_{p-1}} w_{j_p i_{p-1}},$$
$$\mathbb{E}[\phi'(\mathcal{U}^{\ell+p}_{i_p}) \mid w_{i_p i_{p-1}}] = \Phi\Big(\frac{w_{i_p i_{p-1}} a_{i_{p-1}}}{\sqrt{1 - a^2_{i_{p-1}}}}\Big),$$
$$\mathbb{E}[\phi'(\mathcal{U}^{\ell+p}_{j_p}) \mid w_{j_p i_{p-1}}] = \Phi\Big(\frac{w_{j_p i_{p-1}} a_{i_{p-1}}}{\sqrt{1 - a^2_{i_{p-1}}}}\Big).$$

Therefore, the expectation in this case is given by:

$$\mathbb{E}[(\mathcal{V}^{\ell+p}_{i_{p-1}})^4 \phi'(\mathcal{U}^{\ell+p}_{i_p}) \phi'(\mathcal{U}^{\ell+p}_{j_p})] = \begin{cases} \frac{3}{2} & i_p = j_p, \\ \frac{3}{4} + \frac{2}{\pi} b_{i_p} b_{j_p} a^2_{i_{p-1}} (3 - a^2_{i_{p-1}}(b^2_{i_p} + b^2_{j_p})) & i_p \neq j_p. \end{cases}$$

To compute the sum, we now notice that equality of indices in one layer ($i_p = j_p$) amounts to multiplying the product $\prod_{p=1}^k \mathbb{E}[(\mathcal{V}^{\ell+p}_{i_{p-1}})^2 (\mathcal{V}^{\ell+p}_{j_{p-1}})^2 \phi'(\mathcal{U}^{\ell+p}_{i_p}) \phi'(\mathcal{U}^{\ell+p}_{j_p})]$ by $6 + O(1/M^{3/2})$ and for every pair of indices (i_p, j_p) there are only $n_{\ell+p}$ summands with this multiplier and $n_{\ell+p}(n_{\ell+p} - 1)$ summands without it. We can also see that if we computed the sum with all the pairs of indices not equal, we would get $\prod_{p=0}^k \Big(n_{\ell+p}/2\Big)^2 \Big(1 - 1/n_{\ell+p} + O(1/M^{3/2})\Big)$. Therefore, we get the desired expression for the expectation of the product:

$$\begin{aligned} \mathbb{E}\Big[\prod_{p=0}^k (\mathcal{N}^{\ell+p}_\delta)^2\Big] &= \prod_{p=0}^k \Big(\frac{\sigma_w^2}{n_{\ell+p}}\Big)^2 \prod_{p=0}^k \Big(\frac{n_{\ell+p}}{2}\Big)^2 \Big(1 - \frac{1}{n_{\ell+p}} + \frac{6}{n_{\ell+p}} + O\Big(\frac{1}{M^{3/2}}\Big)\Big) \\ &= \prod_{p=0}^k \Big(\frac{\sigma_w^2}{2}\Big)^2 \Big(1 + \frac{5}{n_{\ell+p}} + O\Big(\frac{1}{M^{3/2}}\Big)\Big) \\ &= \prod_{p=0}^k \mathbb{E}[(\mathcal{N}^{\ell+p}_\delta)^2] \Big(1 + O\Big(\frac{1}{M^{3/2}}\Big)\Big). \end{aligned}$$

□

Lemma A.5 (Dispersion of $\Theta_W(x, x)$)**.** *Consider a fully-connected DNN of depth L defined in (3) initialized as in (6). The input dimension is given by $n_0 = \alpha_0 M$, the output dimension is 1, and the hidden layers have constant width M. The activation function in the hidden layers is ReLU, i.e. $\phi(x) = x\mathbb{1}\{x > 0\}$, and the output layer is linear. Assume also that the biases are initialized to zero, i.e. $\sigma_b = 0$, and the input data is normalized. Then the component of the NTK corresponding to the weights $\Theta_W(x, x) := \sum_{\ell=1}^L \sum_{ij} \Big(\frac{\partial f(x)}{\partial \mathbf{W}^\ell_{ij}}\Big)^2$ has the following properties at initialization:*

$$\mathbb{E}[\Theta_W(x, x)] = \Big(\frac{\sigma_w^2}{2}\Big)^{L-1} \Big(1 + \frac{M}{n_0}(L - 1)\Big), \tag{31}$$

$$\frac{\mathbb{E}[\Theta^2_W(x, x)]}{\mathbb{E}^2[\Theta_W(x, x)]} \xrightarrow[\substack{M\to\infty, L\to\infty, \\ L/M \to \lambda \in \mathbb{R}}]{} \frac{1}{2\lambda} e^{5\lambda} \left(1 - \frac{1}{4\lambda}(1 - e^{-4\lambda})\right). \tag{32}$$

Proof. Using backpropagation formulas for the gradients, we can rewrite the NTK as follows:

$$\begin{aligned}\Theta_W(x,x) &= \sum_{\ell=1}^{L}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}(\boldsymbol{\delta}_i^\ell)^2(\mathbf{x}_j^{\ell-1})^2 \\ &= \sum_{\ell=1}^{L}\|\boldsymbol{\delta}^\ell\times\mathbf{x}^{\ell-1}\|^2 = \sum_{\ell=1}^{L}\|\boldsymbol{\delta}^\ell\|^2\|\mathbf{x}^{\ell-1}\|^2 \\ &= \sum_{\ell=1}^{L}\|\boldsymbol{\delta}^L\|^2\|\mathbf{x}^0\|^2\prod_{j=\ell}^{L-1}\frac{\|\boldsymbol{\delta}^j\|^2}{\|\boldsymbol{\delta}^{j+1}\|^2}\prod_{k=1}^{\ell-1}\frac{\|\mathbf{x}^k\|^2}{\|\mathbf{x}^{k-1}\|^2} \\ &= \sum_{\ell=1}^{L}\|\boldsymbol{\delta}^L\|^2\|\mathbf{x}^0\|^2\prod_{j=\ell}^{L-1}\mathcal{N}_\delta^j\prod_{k=1}^{\ell-1}\mathcal{N}_x^k.\end{aligned}$$

Here for the simplicity of notation we omit the dependence on $\boldsymbol{\delta}^\ell$ and $\mathbf{x}^{\ell-1}$ on the input x.

If the last layer has a linear activation and the input data is normalized, we also have that $\|\boldsymbol{\delta}^L\|^2\|\mathbf{x}^0\|^2 = 1$. Then, using the results about expectations of $\mathcal{N}_\delta^\ell$ and $\mathcal{N}_x^\ell$ from Lemma A.1 and Lemma A.2, as well as the results about correlations from Lemma A.4, we can write the following for the expectation of $\Theta_W(x,x)$:

$$\mathbb{E}[\Theta_W(x,x)] = \sum_{\ell=1}^{L}\prod_{j=\ell}^{L-1}\frac{\sigma_w^2}{2}\prod_{k=1}^{\ell-1}\frac{\sigma_w^2}{2}\frac{n_k}{n_{k-1}} = \Big(\frac{\sigma_w^2}{2}\Big)^{L-1}\sum_{\ell=1}^{L}\frac{n_{\ell-1}}{n_0}$$

And for constant width of hidden layers, i.e. $n_\ell = M, \ell = 1,\dots,L-1$, this simplifies to

$$\mathbb{E}[\Theta_W(x,x)] = \Big(\frac{\sigma_w^2}{2}\Big)^{L-1}\Big(1+\frac{M}{n_0}(L-1)\Big) \propto \Big(\frac{\sigma_w^2}{2}\Big)^{L}\frac{ML}{n_0}$$

Now we consider the second moment of the NTK, which is given by:

$$\begin{aligned}\mathbb{E}[\Theta_W^2(x,x)] &= \sum_{\ell=1}^{L}\mathbb{E}[\theta_\ell^2] + 2\sum_{1\le l_1<l_2\le L}\mathbb{E}[\theta_{\ell_1}\theta_{\ell_2}], \\ \theta_\ell &= \prod_{j=\ell}^{L-1}\mathcal{N}_\delta^j\prod_{k=1}^{\ell-1}\mathcal{N}_x^k, \quad \ell = 1,\dots,L.\end{aligned}$$

We can open the expectation of the squared terms defined above as follows:

$$\begin{aligned}\mathbb{E}[\theta_\ell^2] &= \mathbb{E}\Big[\prod_{j=\ell}^{L-1}(\mathcal{N}_\delta^j)^2\Big]\mathbb{E}\Big[\prod_{k=1}^{\ell-1}(\mathcal{N}_x^k)^2\Big] \\ &= \Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\Big(\frac{n_{\ell-1}}{n_0}\Big)^2\prod_{j=\ell}^{L-1}\Big(1+\frac{5}{n_j}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)\prod_{k=1}^{\ell-1}\Big(1+\frac{5}{n_k}\Big),\end{aligned}$$

which simplifies to the following expressions in case of constant width M:

$$\begin{aligned}\mathbb{E}[\theta_\ell^2] &= \Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\Big(\frac{M}{n_0}\Big)^2\Big(1+\frac{5}{M}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)^{L-1}, \quad \ell > 1, \\ \mathbb{E}[\theta_1^2] &= \Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\Big(1+\frac{5}{M}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)^{L-1}.\end{aligned}$$

And the mixed terms with $1 \leq \ell_1 < \ell_2 \leq L$ can be calculated as follows:

$$\mathbb{E}[\theta_{\ell_1}\theta_{\ell_2}] = \mathbb{E}\Big[\prod_{j=\ell_2}^{L-1}(\mathcal{N}_\delta^j)^2\Big]\mathbb{E}\Big[\prod_{p=\ell_1}^{\ell_2-1}\mathcal{N}_\delta^p\mathcal{N}_x^p\Big]\prod_{k=1}^{\ell_1-1}\mathbb{E}[(\mathcal{N}_x^k)^2]$$
$$= \Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\Big(\frac{n_{\ell_2-1}n_{\ell_1-1}}{n_0^2}\Big)\prod_{j=\ell_2}^{L-1}\Big(1+\frac{5}{n_j}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)\prod_{p=\ell_1}^{\ell_2-1}\Big(1+\frac{1}{n_p}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)\prod_{k=1}^{\ell_1-1}\Big(1+\frac{5}{n_k}\Big),$$

which for constant width M simplifies to

$$\mathbb{E}[\theta_{\ell_1}\theta_{\ell_2}] = \Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\Big(\frac{M}{n_0}\Big)^2\Big(1+\frac{5}{M}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)^{L-1-\Delta_\ell}\Big(1+\frac{1}{M}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)^{\Delta_\ell}, \quad \ell_1 > 1,$$
$$\mathbb{E}[\theta_1\theta_{\ell_2}] = \Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\Big(\frac{M}{n_0}\Big)\Big(1+\frac{5}{M}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)^{L-\ell_2}\Big(1+\frac{1}{M}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)^{\ell_2-1}.$$

To make the notation lighter, we will denote $x := 1 + 5/M + O\Big(M^{-3/2}\Big)$, $y := 1 + 1/M + O\Big(M^{-3/2}\Big)$, $a := \sigma_w^2/2$ and $\lambda := L/M$ here and in the following proofs. Then we can rewrite the two sums that comprise the second moment of the NTK as follows:

$$\sum_{\ell=1}^{L}\mathbb{E}[\theta_\ell^2] = a^{2(L-1)}x^{L-1}\Big(\frac{M^2}{n_0^2}(L-1)+1\Big)$$
$$= a^{2(L-1)}\frac{M^2L^2}{n_0^2}\left[x^{L-1}\frac{1}{\lambda M}+O\Big(\frac{1}{M^{3/2}}\Big)\right] = a^{2(L-1)}\frac{M^2L^2}{n_0^2}\left[x^{L-1}\frac{1}{\lambda M}+O\Big(\frac{1}{M^{3/2}}\Big)\right],$$

$$\sum_{1\leq\ell_1<\ell_2\leq L}\mathbb{E}[\theta_{\ell_1}\theta_{\ell_2}] = a^{2(L-1)}\frac{M^2}{n_0^2}\sum_{\Delta_\ell=1}^{L-2}(L-1-\Delta_\ell)\,x^{L-1-\Delta_\ell}y^{\Delta_\ell} + a^{2(L-1)}\frac{M}{n_0}\sum_{\ell_2=2}^{L}x^{L-\ell_2}y^{\ell_2-1}$$
$$= a^{2(L-1)}\frac{M^4}{16n_0^2}\big((L-2)yx^L-(L-1)y^2x^{L-1}+xy^L\big)+a^{2(L-1)}\frac{M^2}{4n_0}(yx^{L-1}-y^L)$$
$$= a^{2(L-1)}\frac{M^2L^2}{n_0^2}\left[x^L\left(\frac{1}{4\lambda}\Big(1-\frac{5}{M}\Big)-\frac{1}{16\lambda^2}\Big(1+\frac{5-4\alpha_0}{M}\Big)+O\Big(\frac{1}{M^{3/2}}\Big)\right)+\right.$$
$$\left.+\,y^L\left(\frac{1}{16\lambda^2}\Big(1+\frac{5-4\alpha_0}{M}\Big)+O\Big(\frac{1}{M^{3/2}}\Big)\right)\right]$$

Therefore, the complete expression for the second moment of $\Theta_W(x,x)$ is given by:

$$\sum_{\ell=1}^{L}\mathbb{E}[\theta_\ell^2]+2\sum_{1\leq\ell_1<\ell_2\leq L}\mathbb{E}[\theta_{\ell_1}\theta_{\ell_2}] = a^{2(L-1)}\frac{M^2L^2}{n_0^2}\left[x^L\left(\frac{1}{2\lambda}\Big(1-\frac{3}{M}\Big)-\frac{1}{8\lambda^2}\Big(1+\frac{5-4\alpha_0}{M}\Big)+O\Big(\frac{1}{M^{3/2}}\Big)\right)+\right.$$
$$\left.+\,y^L\left(\frac{1}{8\lambda^2}\Big(1+\frac{5-4\alpha_0}{M}\Big)+O\Big(\frac{1}{M^{3/2}}\Big)\right)\right]$$
$$a^{2(L-1)}\frac{M^2L^2}{n_0^2}\left[x^L\left(\frac{1}{2\lambda}-\frac{1}{8\lambda^2}+O\Big(\frac{1}{M}\Big)\right)+y^L\frac{1}{8\lambda^2}+O\Big(\frac{1}{M}\Big)\right)\right]$$

One can see that in the limit $L\to\infty,\ M\to\infty,\ L/M\to\lambda\in\mathbb{R}$, we have $x^L\to e^{5\lambda}$ and $y^L\to e^{\lambda}$. Therefore, we can find the limit of the desired ratio:

$$\frac{\mathbb{E}[\Theta_W^2(x,x)]}{\Big(\frac{\sigma_w^2}{2}\Big)^{2(L-1)}\frac{L^2M^2}{n_0^2}} \xrightarrow[\substack{M\to\infty,L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{2\lambda}e^{5\lambda}\left(1-\frac{1}{4\lambda}(1-e^{-4\lambda})\right).$$

□

Lemma A.6 (Dispersion of $\Theta_b(x,x)$)**.** *Consider a fully-connected DNN of depth L defined in (3) initialized as in (6). The input dimension is given by $n_0 = \alpha_0 M$, the output dimension is 1, and the hidden layers have constant width M. The activation function in the hidden layers is ReLU, i.e. $\phi(x) = x\mathbb{1}\{x > 0\}$, and the output layer is linear. Assume also that the biases are initialized to zero, i.e. $\sigma_b = 0$, and the input data is normalized. Then the component of the NTK corresponding to the biases $\Theta_b(x,x) := \sum_{\ell=1}^{L}\sum_i \Big(\frac{\partial f(x)}{\partial \mathbf{b}_i^\ell}\Big)^2$ has the following properties at initialization:*

$$\mathbb{E}[\Theta_b(x,x)] = \begin{cases} \dfrac{\left(\frac{\sigma_w^2}{2}\right)^L - 1}{\frac{\sigma_w^2}{2} - 1} & \text{if } \dfrac{\sigma_w^2}{2} \neq 1 \\ L & \text{if } \dfrac{\sigma_w^2}{2} = 1 \end{cases} \tag{33}$$

$$\frac{\mathbb{E}[\Theta_b^2(x,x)]}{\mathbb{E}^2[\Theta_b(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \begin{cases} 1 & \text{if } \dfrac{\sigma_w^2}{2} < 1 \\ \dfrac{2}{25\lambda^2}(e^{5\lambda} - 1) - \dfrac{2}{5\lambda} & \text{if } \dfrac{\sigma_w^2}{2} = 1 \\ e^{5\lambda} & \text{if } \dfrac{\sigma_w^2}{2} > 1 \end{cases} \tag{34}$$

Proof. Using backpropagation equations (7), we can obtain the following expression for $\Theta_b(x,x)$:

$$\Theta_b(x,x) = \sum_{\ell=1}^{L} \|\boldsymbol{\delta}^\ell\|^2 = \|\boldsymbol{\delta}^L\|^2 \sum_{\ell=1}^{L}\prod_{j=\ell}^{L-1} \frac{\|\boldsymbol{\delta}^j\|^2}{\|\boldsymbol{\delta}^{j+1}\|^2} = \sum_{\ell=1}^{L}\prod_{j=\ell}^{L-1} \mathcal{N}_\delta^j.$$

In this lemma, we will again denote $a := \sigma_w^2/2$ and $x := 1 + 5/M + O(1/M^{3/2})$. And in the following computations, we will need to consider cases with $a \neq 1$ and $a = 1$ separately.

Case 1: $a \neq 1$**.** In this case the expectation is given by a sum of a geometric progression:

$$\mathbb{E}[\Theta_b(x,x)] = \sum_{\ell=1}^{L}\prod_{j=\ell}^{L-1} \mathbb{E}[\mathcal{N}_\delta^j] = \sum_{\ell=1}^{L} a^{L-\ell} = \frac{a^L - 1}{a - 1}$$

And for the second moment we can write:

$$\begin{aligned}\mathbb{E}[\Theta_b(x,x)^2] &= \mathbb{E}\left[\Big(\sum_{\ell=1}^{L}\prod_{j=\ell}^{L-1}\mathcal{N}_\delta^j\Big)^2\right] = \sum_{\ell=1}^{L}\mathbb{E}\Big[\prod_{j=\ell}^{L-1}(\mathcal{N}_\delta^j)^2\Big] + 2\sum_{1\le\ell_1<\ell_2\le L}\mathbb{E}\Big[\prod_{j=\ell_1}^{\ell_2-1}\mathcal{N}_\delta^j\prod_{k=\ell_2}^{L-1}\Big(\mathcal{N}_\delta^k\Big)^2\Big] \\ &= \sum_{\ell=1}^{L} a^{2(L-\ell)}\prod_{j=\ell}^{L-1}\Big(1 + \frac{5}{n_j} + O\Big(\frac{1}{M^{3/2}}\Big)\Big) + 2\sum_{1\le\ell_1<\ell_2\le L} a^{2L-\ell_1-\ell_2}\prod_{k=\ell_2}^{L-1}\Big(1 + \frac{5}{n_k} + O\Big(\frac{1}{M^{3/2}}\Big)\Big)\end{aligned}$$

For constant width M the above expression simplifies to the following sum:

$$\begin{aligned}\mathbb{E}[\Theta_b(x,x)^2] &= \sum_{\ell=1}^{L} a^{2(L-\ell)}x^{L-\ell} + 2\sum_{1\le\ell_1<\ell_2\le L} a^{2L-\ell_1-\ell_2}x^{L-\ell_2} \\ &= \sum_{\ell=1}^{L} a^{2(L-\ell)}x^{L-\ell} + 2\sum_{\ell_1=1}^{L-1} a^{2(L-\ell)}x^{L-\ell}\sum_{\Delta_\ell=1}^{L-\ell} a^{-\Delta_\ell}x^{-\Delta_\ell}\end{aligned}$$

And the involved terms can be further calculated explicitly as follows:

$$\begin{aligned}
\mathbb{E}[\Theta_b(x,x)^2] &= \frac{a^{2L}x^L-1}{a^2x-1} + \frac{2}{ax-1}\sum_{\ell=1}^{L-1} a^{2(L-l)}x^{L-l}(1-a^{l-L}x^{l-L}) \\
&= \frac{a^{2L}x^L-1}{a^2x-1} + \frac{2}{ax-1}\Big(\frac{a^{2L}x^L-1}{a^2x-1} - 1 - \frac{a^L-a}{a-1}\Big) \\
&= \frac{a^{2L}x^L-1}{a^2x-1}\frac{ax+1}{ax-1} - \frac{2}{ax-1}\frac{a^L-1}{a-1} \\
&= \frac{1}{(a-1)^2}\left[a^{2L}x^L\Big(1+O\Big(\frac{1}{M}\Big)\Big) - 2a^L\Big(1+O\Big(\frac{1}{M}\Big)\Big) + 1 + O\Big(\frac{1}{M}\Big)\Big)\right]
\end{aligned}$$

If $a < 1$, the expectation and the second moment have finite limits:

$$\begin{aligned}
\mathbb{E}[\Theta_b(x,x)] &\xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{1-a} \\
\mathbb{E}[\Theta_b(x,x)^2] &\xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{-1}{a^2-1}\frac{a+1}{a-1} - \frac{2}{a-1}\frac{-1}{a-1} = \frac{1}{(a-1)^2}
\end{aligned}$$

Therefore for $a < 1$ we have

$$\frac{\mathbb{E}[\Theta_b^2(x,x)]}{\mathbb{E}^2[\Theta_b(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 1$$

On the other hand, if $a > 1$ then the limits are infinite but there is a finite limit of the ratio:

$$\frac{\mathbb{E}[\Theta_b(x,x)^2]}{a^{2L}/(a-1)^2} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} e^{5\lambda}$$

Case 2: $a = 1$**.** In this case, the expectation is just a sum of ones, so we have

$$\mathbb{E}[\Theta_b(x,x)] = \sum_{\ell=1}^{L} 1 = L$$

And the second moment can be calculated as follows:

$$\begin{aligned}
\mathbb{E}[\Theta_b(x,x)^2] &= \sum_{\ell=1}^{L} x^{L-\ell} + 2\sum_{\ell=1}^{L-1} x^{L-\ell}\sum_{\Delta_\ell=1}^{L-l} x^{-\Delta_\ell} = \frac{x^L-1}{x-1} + \frac{2}{x-1}\Big(\sum_{\ell=1}^{L-1} x^{L-\ell} - \sum_{\ell=1}^{L-1} 1\Big) \\
&= \frac{x^L-1}{x-1} + \frac{2}{x-1}\Big(\frac{x^L-x}{x-1} - L + 1\Big) = M^2\left[x^L\Big(\frac{1}{5M}+\frac{2}{25}\Big) - \frac{1}{5M} - \frac{10\lambda+2}{25}\right] \\
&= \frac{x^L-1}{x-1} + \frac{2}{x-1}\Big(\frac{x^L-x}{x-1} - L + 1\Big) = L^2\left[x^L\Big(\frac{2}{25\lambda^2}+O\Big(\frac{1}{M}\Big)\Big) - \frac{2}{5\lambda} - \frac{2}{25\lambda^2} + O\Big(\frac{1}{M}\Big)\right]
\end{aligned}$$

Then for the desired ratio we have the following result in the limit:

$$\frac{\mathbb{E}[\Theta_b(x,x)^2]}{L^2} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{2}{25\lambda^2}(e^{5\lambda}-1) - \frac{2}{5\lambda},$$

which completes the proof for all the cases. □

Lemma A.7 (Dispersion of $\Theta_W(x,x)\Theta_b(x,x)$)**.** *Consider a fully-connected DNN of depth L defined in (3) initialized as in (6). The input dimension is given by $n_0 = \alpha_0 M$, the output dimension is 1, and the hidden layers have constant width M. The activation function in the hidden layers is ReLU, i.e. $\phi(x) = x\mathbb{1}\{x > 0\}$, and the output layer is linear. Assume also that the biases are initialized to zero, i.e. $\sigma_b = 0$, and the input data is normalized. Then the following statements hold:*

1. *In the chaotic phase, i.e. if $\sigma_w^2/2 > 1$:*

$$\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0 \tag{35}$$

2. *In the ordered phase, i.e. if $\sigma_w^2/2 < 1$:*

$$\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{\mathbb{E}^2[\Theta_b(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0 \tag{36}$$

3. *At the EOC, i.e. if $\sigma_w^2/2 = 1$:*

$$\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{L^2} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{4\alpha_0}\Big(e^{5\lambda}\frac{9}{25\lambda^2} - e^{\lambda}\frac{1}{\lambda^2} - \frac{4}{5\lambda} + \frac{16}{25\lambda^2}\Big) \tag{37}$$

Proof. We can decompose $\Theta_W(x,x)\Theta_b(x,x)$ into telescopic products as follows:

$$\begin{aligned}
\Theta_W(x,x)\Theta_b(x,x) &= \sum_{\ell=1}^{L}\|\boldsymbol{\delta}^{\ell}\|^2\|\mathbf{x}^{\ell-1}\|^2\sum_{\ell'=1}^{L}\|\boldsymbol{\delta}^{\ell'}\|^2 \\
&= \|\boldsymbol{\delta}^{L}\|^4\|\mathbf{x}^{0}\|^2\sum_{\ell=1}^{L}\prod_{j=\ell}^{L-1}\frac{\|\boldsymbol{\delta}^{j}\|^4}{\|\boldsymbol{\delta}^{j+1}\|^4}\prod_{k=1}^{\ell-1}\frac{\|\mathbf{x}^{k}\|^2}{\|\mathbf{x}^{k-1}\|^2} \\
&+ \|\boldsymbol{\delta}^{L}\|^4\|\mathbf{x}^{0}\|^2\sum_{1\le\ell_1<\ell_2\le L}\prod_{p=\ell_1}^{\ell_2-1}\frac{\|\boldsymbol{\delta}^{p}\|^2}{\|\boldsymbol{\delta}^{p+1}\|^2}\prod_{j=\ell_2}^{L-1}\frac{\|\boldsymbol{\delta}^{j}\|^4}{\|\boldsymbol{\delta}^{j+1}\|^4}\prod_{k=1}^{\ell_2-1}\frac{\|\mathbf{x}^{k}\|^2}{\|\mathbf{x}^{k-1}\|^2} \\
&+ \|\boldsymbol{\delta}^{L}\|^4\|\mathbf{x}^{0}\|^2\sum_{1\le\ell_1<\ell_2\le L}\prod_{p=\ell_1}^{\ell_2-1}\frac{\|\boldsymbol{\delta}^{p}\|^2}{\|\boldsymbol{\delta}^{p+1}\|^2}\prod_{j=\ell_2}^{L-1}\frac{\|\boldsymbol{\delta}^{j}\|^4}{\|\boldsymbol{\delta}^{j+1}\|^4}\prod_{k=1}^{\ell_1-1}\frac{\|\mathbf{x}^{k}\|^2}{\|\mathbf{x}^{k-1}\|^2}
\end{aligned}$$

Then, as in the previous lemmas, we can calculate the expectation using the results of Lemmas A.1, A.2 and A.4:

$$\begin{aligned}
\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)] &= \sum_{\ell=1}^{L}\mathbb{E}\Big[\prod_{j=\ell}^{L-1}(\mathcal{N}_\delta^j)^2\Big]\prod_{k=1}^{\ell-1}\mathbb{E}[\mathcal{N}_x^k] + \sum_{1\le\ell_1<\ell_2\le L}\prod_{p=\ell_1}^{\ell_2-1}\mathbb{E}[\mathcal{N}_\delta^p\mathcal{N}_x^p]\mathbb{E}\Big[\prod_{j=\ell_2}^{L-1}(\mathcal{N}_\delta^j)^2\Big]\prod_{k=1}^{\ell_1-1}\mathbb{E}[\mathcal{N}_x^k] \\
&+ \sum_{1\le\ell_1<\ell_2\le L}\prod_{p=\ell_1}^{\ell_2-1}\mathbb{E}[\mathcal{N}_\delta^p]\mathbb{E}\Big[\prod_{j=\ell_2}^{L-1}(\mathcal{N}_\delta^j)^2\Big]\prod_{k=1}^{\ell_1-1}\mathbb{E}[\mathcal{N}_x^k] \\
&= \sum_{\ell=1}^{L}a^{2L-\ell-1}\frac{n_{\ell-1}}{n_0}\prod_{j=\ell}^{L-1}\Big(1+\frac{5}{n_j}+\Big(\frac{1}{M^{3/2}}\Big)\Big) \\
&+ \sum_{1\le\ell_1<\ell_2\le L}a^{2L-\ell_1-1}\frac{n_{\ell_2-1}}{n_0}\prod_{p=\ell_1}^{\ell_2-1}\Big(1+\frac{1}{n_p}+\Big(\frac{1}{M^{3/2}}\Big)\Big)\prod_{j=\ell_2}^{L-1}\Big(1+\frac{5}{n_j}+\Big(\frac{1}{M^{3/2}}\Big)\Big) \\
&+ \sum_{1\le\ell_1<\ell_2\le L}a^{2L-\ell_2-1}\frac{n_{\ell_1-1}}{n_0}\prod_{j=\ell_2}^{L-1}\Big(1+\frac{5}{n_j}+\Big(\frac{1}{M^{3/2}}\Big)\Big),
\end{aligned}$$

where we denoted $a := \sigma_w^2/2$. As in Lemma A.6, we will need to consider the cases with $a \neq 1$ and $a = 1$ separately here.

Case 1: $a \neq 1$**.** For constant width M the above expression for the expectation simplifies to:

$$
\begin{aligned}
\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)] &= \sum_{\ell=1}^{L} \frac{n_{\ell-1}}{n_0} a^{2L-\ell-1} x^{L-\ell} + \sum_{\ell_1=1}^{L-1} a^{2L-\ell_1-1} \sum_{\ell_2=\ell_1+1}^{L} \frac{n_{\ell_2-1}}{n_0} y^{\ell_2-\ell_1} x^{L-\ell_2} \\
&+ \sum_{\ell_1=1}^{L-1} \frac{n_{\ell_1-1}}{n_0} \sum_{\ell_2=\ell_1+1}^{L} a^{2L-\ell_2-1} x^{L-\ell_2} \\
&= a^{2(L-1)} x^{L-1} \Big(1 + \frac{M}{n_0} \frac{1}{ax-1}\Big) - \frac{M}{n_0} \frac{a^{L-1}}{ax-1} \\
&+ a^{2(L-1)} x^{L-1} \frac{M}{n_0} \frac{1}{x-y} \frac{ay}{ax-1} - a^{2(L-1)} y^{L} \frac{M}{n_0} \frac{1}{x-y} \frac{ay}{ay-1} \\
&+ a^{L} \frac{M}{n_0} \frac{1}{x-y} \Big(\frac{-xy}{ax-1} + \frac{y^2}{ay-1} \Big) \\
&+ a^{2(L-1)} x^{L-1} \frac{1}{ax-1} \Big(1 + \frac{M}{n_0} \frac{1}{ax-1}\Big) - \frac{a^{L-1}}{ax-1} - \frac{a^{L-1}}{ax-1} \frac{M}{n_0} (L-2) - \frac{M}{n_0} \frac{a^L x}{(ax-1)^2} \\
&= a^{2(L-1)} x^{L-1} \left[\Big(1 + \frac{M}{n_0} \frac{1}{ax-1}\Big)\Big(1 + \frac{1}{ax-1}\Big) + \frac{M}{n_0} \frac{1}{x-y} \frac{ay}{ax-1} \right] \\
&- a^{2(L-1)} y^{L} \frac{M}{n_0} \frac{1}{x-y} \frac{ay}{ay-1} \\
&+ a^{L-1} \left[\frac{M}{n_0} \frac{a}{x-y} \Big(\frac{-xy}{ax-1} + \frac{y^2}{ay-1} \Big) - \frac{1}{ax-1} \Big(1 + \frac{M(L-1)}{n_0}\Big) - \frac{M}{n_0} \frac{ax}{(ax-1)^2} \right] \\
&= a^{2(L-1)} x^{L-1} \frac{M}{4\alpha_0} \frac{a}{ax-1} \left[y + \frac{4\alpha_0 x}{M} + \frac{4x}{(ax-1)M} + O\Big(\frac{1}{M}\Big) \right] \\
&- a^{2(L-1)} y^{L} \frac{M}{4\alpha_0} \frac{a}{ay-1} \left[y + O\Big(\frac{1}{M}\Big) \right] \\
&+ a^{L-1} \frac{M}{4\alpha_0} \frac{a}{ax-1} \left[\frac{16}{M^2 (ay-1)(ax-1)} - \frac{4\alpha_0}{Ma} \Big(1 + \frac{L-1}{\alpha_0}\Big) \right] \\
&= \frac{M}{4\alpha_0} \frac{a}{a-1} \left[a^{2(L-1)} x^{L-1} - a^{2(L-1)} y^{L} - 4a^{L-1} \lambda + O\Big(\frac{1}{M}\Big) \right],
\end{aligned}
$$

where we also denoted $x := 1 + 5/M + O(1/M^{3/2})$ and $y := 1 + 1/M + O(1/M^{3/2})$. From the last expression, we see that $\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]$ tends to zero if $a < 1$, therefore in this case we get

$$
\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{\mathbb{E}^2[\Theta_b(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty, \\ L/M \to \lambda \in \mathbb{R}}]{} 0, \tag{38}
$$

using the result of Lemma A.6 that $\mathbb{E}[\Theta_b^2(x,x)]$ has a finite limit when $a < 1$.

On the other hand, if $a > 1$, we can see that $\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]$ contains polynomials of M and L of degree not larger than 1. Therefore, we have

$$
\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{a^{2L} M^2 L^2 / n_0^2} \xrightarrow[\substack{M\to\infty, L\to\infty, \\ L/M \to \lambda \in \mathbb{R}}]{} 0, \tag{39}
$$

which completes the proof for the case when $a \neq 1$.

Case 2: $a = 1$**.** In this case, the expression for the expectation with constant width M is given by:

$$\begin{aligned}\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)] &= \sum_{\ell=1}^{L}\frac{n_{\ell-1}}{n_0}x^{L-\ell} + \sum_{\ell_1=1}^{L-1}\sum_{\ell_2=\ell_1+1}^{L}\frac{n_{\ell_2-1}}{n_0}y^{\ell_2-\ell_1}x^{L-\ell_2} \\ &+ \sum_{\ell_1=1}^{L-1}\frac{n_{\ell_1-1}}{n_0}\sum_{\ell_2=\ell_1+1}^{L}x^{L-\ell_2} \\ &= \frac{M^2}{4\alpha_0}\left[x^{L-1}\Big(\frac{9}{25}+O\Big(\frac{1}{M}\Big)\Big) - y^L\Big(1+O\Big(\frac{1}{M}\Big)\Big) - \frac{4\lambda}{5} + \frac{16}{25}\right]\end{aligned}$$

□

Theorem A.8 (Dispersion of the NTK at initialization)**.** *Consider a fully-connected DNN of depth L defined in (3) initialized as in (6). The input dimension is given by $n_0 = \alpha_0 M$, the output dimension is 1, and the hidden layers have constant width M. The activation function in the hidden layers is ReLU, i.e. $\phi(x) = x\mathbb{1}\{x > 0\}$, and the output layer is linear. Assume also that the biases are initialized to zero, i.e. $\sigma_b = 0$, and the input data is normalized. Then the dispersion of the NTK at initialization is given by the following expressions:*

1. *In the **chaotic phase** ($a := \sigma_w^2/2 > 1$), the NTK dispersion grows exponentially with the depth-to-width ratio $\lambda := L/M$ as*
$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{2\lambda}e^{5\lambda}\left(1 - \frac{1}{4\lambda}(1-e^{-4\lambda})\right) \tag{40}$$
2. *At the **EOC** ($a = 1$), the NTK dispersion grows exponentially with the depth-to-width ratio λ as well, but with a slower rate given by*
$$\begin{aligned}\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{(1+\alpha_0)^2\lambda}\Bigg[&e^{5\lambda}\Big(\frac{1}{2} + \frac{16\alpha_0^2+36\alpha_0-25}{200\lambda}\Big) \\ &+ e^{\lambda}\frac{1-4\alpha_0}{8\lambda} + \frac{2\alpha_0(4-\alpha_0)}{25\lambda} - \frac{2\alpha_0(1+\alpha_0)}{5}\Bigg]\end{aligned} \tag{41}$$
3. *In the **ordered phase** ($a < 1$), the NTK variance does not grow with λ and we have*
$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 1 \tag{42}$$

Proof. We will consider the cases of the ordered phase ($a := \sigma_w^2/2 < 1$), the chaotic phase ($a > 1$) and the EOC ($a = 1$) separately.

Case 1: Chaotic phase. Using the results of Lemmas A.5, A.6, and A.7 and taking into account that $a > 1$, we obtain the following limit:

$$\frac{\mathbb{E}[\Theta_b(x,x)]}{\mathbb{E}[\Theta_W(x,x)]} = \frac{\dfrac{a^L-1}{a-1}}{a^{L-1}(1+\dfrac{M}{n_0}(L-1))} = \frac{\dfrac{a-a^{-L+1}}{a-1}}{1+\dfrac{M}{n_0}(L-1)} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0$$

Therefore, recalling that $\Theta(x,x) = \Theta_W(x,x) + \Theta_b(x,x)$, we get the ratio between the complete NTK and its component corresponding to weights:

$$\frac{\mathbb{E}^2[\Theta(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]} = \Big(1 + \frac{\mathbb{E}[\Theta_b(x,x)]}{\mathbb{E}[\Theta_W(x,x)]}\Big)^2 \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 1$$

Similarly, from Lemmas A.5, A.6, A.7, we can also obtain the following limit:

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]} = \frac{\mathbb{E}[\Theta_W^2(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]} + \frac{\mathbb{E}[\Theta_b^2(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]} + \frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{\mathbb{E}[\Theta_W^2(x,x)]}{\mathbb{E}^2[\Theta_W(x,x)]}$$

Therefore, the dispersion of the NTK is determined by $\Theta_W(x,x)$ in the infinite-depth-and-width limit in case of the initialization in the chaotic phase. Then we have the following expression for the dispersion in the limit:

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{2\lambda} e^{5\lambda}\left(1 - \frac{1}{4\lambda}(1 - e^{-4\lambda})\right),$$

which completes the first part of the proof.

Case 2: Ordered phase. In the ordered phase, i.e. if $a < 1$, we have that $a^L \to 0$ as $L \to \infty$, so Lemmas A.5, A.6 and A.7 suggest different relations between the terms of the NTK:

$$\frac{\mathbb{E}[\Theta_W(x,x)]}{\mathbb{E}[\Theta_b(x,x)]} = \frac{a^{L-1}(1 + \frac{M}{n_0}(L-1))}{\frac{a^L - 1}{a - 1}} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0$$

$$\mathbb{E}[\Theta_W^2(x,x)] \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0,$$

$$\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)] \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0$$

$$\mathbb{E}[\Theta_b^2(x,x)] \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{(a-1)^2}$$

Therefore, the dispersion of the NTK is determined by the component corresponding to biases Θ_b in the limit in case of initialization in the ordered phase:

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 1,$$

which completes this part of the proof.

Case 3: EOC. Here we have $a^\ell = 1$ for any $\ell \in \mathbb{N}$. Therefore, we can simplify the expressions for expectations from Lemmas A.5, A.6 and A.7 as follows:

$$\mathbb{E}[\Theta_W(x,x)] = 1 + \frac{1}{\alpha_0}(L-1),$$
$$\mathbb{E}[\Theta_b(x,x)] = L.$$

Then the expectation of the complete NTK is given by:

$$\mathbb{E}[\Theta(x,x)] = \mathbb{E}[\Theta_W(x,x)] + \mathbb{E}[\Theta_b(x,x)] = \frac{L}{\alpha_0}\left(1 + \alpha_0 + \frac{\alpha_0}{L} - \frac{1}{L}\right) \propto \frac{L}{\alpha_0}(1 + \alpha_0).$$

The squared NTK is given by $\Theta^2(x,x) = \Theta_W^2(x,x) + 2\Theta_W(x,x)\Theta_b(x,x) + \Theta_b^2(x,x)$. Then we need to consider the

expectations of all the components of this sum:

$$\frac{\mathbb{E}[\Theta_W^2(x,x)]}{L^2/\alpha_0^2} \xrightarrow[\substack{M\to\infty,L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} e^{5\lambda}\Big(\frac{1}{2\lambda}-\frac{1}{8\lambda^2}\Big)+e^{\lambda}\frac{1}{8\lambda^2},$$

$$\frac{\mathbb{E}[\Theta_b^2(x,x)]}{L^2/\alpha_0^2} \xrightarrow[\substack{M\to\infty,L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \alpha_0^2\Big(\frac{2}{25\lambda^2}e^{5\lambda}-\frac{2}{25\lambda^2}-\frac{2}{5\lambda}\Big),$$

$$\frac{\mathbb{E}[\Theta_W(x,x)\Theta_b(x,x)]}{L^2/\alpha_0^2} \xrightarrow[\substack{M\to\infty,L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{\alpha_0}{4}\Big(\frac{9}{25\lambda^2}e^{5\lambda}-\frac{1}{\lambda^2}e^{\lambda}-\frac{4}{5\lambda}+\frac{16}{25\lambda^2}\Big).$$

Putting the above expressions together, we get the following limit for the desired ratio:

$$\frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty,L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \frac{1}{(1+\alpha_0)^2\lambda}\Bigg[e^{5\lambda}\Big(\frac{1}{2}+\frac{16\alpha_0^2+36\alpha_0-25}{200\lambda}\Big) + e^{\lambda}\frac{1-4\alpha_0}{8\lambda}+\frac{2\alpha_0(4-\alpha_0)}{25\lambda}-\frac{2\alpha_0(1+\alpha_0)}{5}\Bigg],$$

which completes the proof. □

A.2. Non-diagonal elements of the NTK

Theorem A.9 (Non-diagonal elements of the NTK)**.** *Consider a fully-connected DNN of depth L defined in (3) initialized as in (6). The input dimension is given by $n_0=\alpha_0 M$, the output dimension is 1, and the hidden layers have constant width M. The activation function in the hidden layers is ReLU, i.e. $\phi(x)=x\mathbb{1}\{x>0\}$, and the output layer is linear. Assume also that the biases are initialized to zero, i.e. $\sigma_b=0$, and the input data is normalized. Then for the ratio of non-diagonal and diagonal elements of the NTK we have:*

$$1 \geq \lim_{\substack{L\to\infty,M\to\infty\\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta(x,\tilde{x})]}{\mathbb{E}[\Theta(x,x)]} \geq \frac{1}{4}$$

Moreover, for the dispersion of the non-diagonal elements we have:

$$\lim_{\substack{L\to\infty,M\to\infty\\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta^2(x,\tilde{x})]}{\mathbb{E}^2[\Theta(x,\tilde{x})]} \leq 16 \lim_{\substack{L\to\infty,M\to\infty\\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]}$$

Proof. The non-diagonal element of the NTK on point x and $\tilde{x}$ is given by

$$\Theta(x,\tilde{x})=\sum_{\ell=1}^{L}\langle \boldsymbol{\delta}^{\ell},\tilde{\boldsymbol{\delta}}^{\ell}\rangle\langle \mathbf{x}^{\ell-1},\tilde{\mathbf{x}}^{\ell-1}\rangle+\sum_{\ell=1}^{L}\langle \boldsymbol{\delta}^{\ell},\tilde{\boldsymbol{\delta}}^{\ell}\rangle,$$

where the activations and the backpropagated errors with tilde correspond to $\tilde{x}$. Same as in Lemma A.1, we can write the following for the involved dot products:

$$\langle \mathbf{x}^{\ell},\tilde{\mathbf{x}}^{\ell}\rangle=\frac{\sigma_w^2}{n_{\ell-1}}\|\mathbf{x}^{\ell-1}\|\|\tilde{\mathbf{x}}^{\ell-1}\|\sum_{i=1}^{n_\ell}\phi(\mathcal{U}_i^{\ell})\phi(\tilde{\mathcal{U}}_i^{\ell})$$

We notice that in this case $\mathcal{U}_i^{\ell}\sim\mathcal{N}(0,1)$ and $\tilde{\mathcal{U}}_i^{\ell}\sim\mathcal{N}(0,1)$ are correlated variables and the covariance is given by $\rho_x^{\ell-1}:=\frac{\langle \mathbf{x}^{\ell-1},\tilde{\mathbf{x}}^{\ell-1}\rangle}{\|\mathbf{x}^{\ell-1}\|\|\tilde{\mathbf{x}}^{\ell-1}\|}$. The distribution of $\mathcal{U}_i^{\ell}$ and $\tilde{\mathcal{U}}_i^{\ell}$ depends only on the angle between the activations and not on the norms.

Assuming $\rho_x^{\ell-1}$ is given, we can calculate the expectation of $\phi(\mathcal{U}_i^{\ell})\phi(\tilde{\mathcal{U}}_i^{\ell})$:

$$\mathbb{E}[\phi(\mathcal{U}_i^{\ell})\phi(\tilde{\mathcal{U}}_i^{\ell})\mid\rho_x^{\ell-1}]=\frac{1}{2\pi}\big(\sqrt{1-(\rho_x^{\ell-1})^2}+\rho_x^{\ell-1}\pi/2+\rho_x^{\ell-1}\arcsin\rho_x^{\ell-1}\big)$$

Then, denoting $g(x) := \frac{1}{\pi}\big(\sqrt{1-x^2} + x\pi/2 + x\arcsin x\big)$, we have

$$\mathbb{E}\Big[\frac{\langle \mathbf{x}^\ell, \tilde{\mathbf{x}}^\ell\rangle}{\langle \mathbf{x}^{\ell-1}, \tilde{\mathbf{x}}^{\ell-1}\rangle}\Big] = \frac{\sigma_w^2}{2}\frac{n_\ell}{n_{\ell-1}}\mathbb{E}\Big[\frac{g(\rho_x^{\ell-1})}{\rho_x^{\ell-1}}\Big]$$

We can reason in the same way to find expected dot products of the backpropagated errors:

$$\langle \boldsymbol{\delta}^\ell, \tilde{\boldsymbol{\delta}}^\ell\rangle = \frac{\sigma_w^2}{n_\ell}\|\boldsymbol{\delta}^{\ell+1}\|\|\tilde{\boldsymbol{\delta}}^{\ell+1}\|\sum_{i=1}^{n_\ell}\phi'(\mathcal{U}_i^\ell)\phi'(\tilde{\mathcal{U}}_i^\ell)\mathcal{V}_i^{\ell+1}\tilde{\mathcal{V}}_i^{\ell+1}$$

We can also calculate the involved expectations:

$$\mathbb{E}[\phi'(\mathcal{U}_i^\ell)\phi'(\tilde{\mathcal{U}}_i^\ell) \mid \rho_x^{\ell-1}] = \frac{1}{2\pi}\Big(\frac{\pi}{2} + \arcsin\rho_x^{\ell-1}\Big),$$
$$\mathbb{E}[\mathcal{V}_i^\ell\tilde{\mathcal{V}}_i^\ell \mid \rho_\delta^\ell] = \rho_\delta^\ell := \frac{\langle \boldsymbol{\delta}^\ell, \tilde{\boldsymbol{\delta}}^\ell\rangle}{\|\boldsymbol{\delta}^\ell\|\|\tilde{\boldsymbol{\delta}}^\ell\|}$$

And, using the above expressions, we get

$$\mathbb{E}\Big[\frac{\langle \boldsymbol{\delta}^\ell, \tilde{\boldsymbol{\delta}}^\ell\rangle}{\langle \boldsymbol{\delta}^{\ell+1}, \tilde{\boldsymbol{\delta}}^{\ell+1}\rangle}\Big] = \frac{\sigma_w^2}{2}\mathbb{E}\Big[\frac{1}{\pi}\Big(\frac{\pi}{2} + \arcsin\rho_x^{\ell-1}\Big)\Big]$$

We also need to consider the expectation of ρ_x^ℓ:

$$\mathbb{E}[\rho_x^\ell \mid \rho_x^{\ell-1}] = \mathbb{E}\left[\frac{\sum_i \phi(\mathcal{U}_i^\ell)\phi(\tilde{\mathcal{U}}_i^\ell)}{\sqrt{\sum_i \phi^2(\mathcal{U}_i^\ell)}\sqrt{\sum_i \phi^2(\tilde{\mathcal{U}}_i^\ell)}} \mid \rho_x^{\ell-1}\right] \xrightarrow[n_{\ell-1}\to\infty]{} g(\rho_x^{\ell-1}),$$

where the correction to the above expectation for finite width is of order $O(1/M)$ since the components approach normality with this rate. Moreover, the estimator of correlation coefficient has a negative bias, therefore $\mathbb{E}[\rho_x^\ell \mid \rho_x^{\ell-1}]$ approaches $g(\rho_x^{\ell-1})$ from below with $n_{\ell-1}\to\infty$. Then we have

$$\mathbb{E}\Big[\langle \mathbf{x}^\ell, \tilde{\mathbf{x}}^\ell\rangle\Big] = \langle \mathbf{x}^0, \tilde{\mathbf{x}}^0\rangle\mathbb{E}\Big[\prod_{k=1}^{\ell}\frac{\langle \mathbf{x}^k, \tilde{\mathbf{x}}^k\rangle}{\langle \mathbf{x}^{k-1}, \tilde{\mathbf{x}}^{k-1}\rangle}\Big] = \langle \mathbf{x}^0, \tilde{\mathbf{x}}^0\rangle a^\ell\frac{n_\ell}{n_0}\mathbb{E}\Big[\frac{g(\rho_x^{\ell-1})}{\rho_x^0}\prod_{k=0}^{\ell-2}\frac{g(\rho^k)}{\rho^{k+1}}\Big] \geq \|\mathbf{x}^0\|\|\tilde{\mathbf{x}}^0\|a^\ell\frac{n_\ell}{n_0}\mathbb{E}\big[\rho_x^\ell\big]$$

Similarly, denoting $f(x) := \frac{1}{\pi}(\pi/2 + \arcsin x)$, we get the following for the products of backpropagated errors:

$$\mathbb{E}[\langle \boldsymbol{\delta}^\ell, \tilde{\boldsymbol{\delta}}^\ell\rangle] \geq a^{L-\ell}\prod_{k=\ell}^{L-1} f\big(\mathbb{E}[\rho_x^{k-1}]\big)$$

Now we notice that $\mathbb{E}[\rho_x^\ell] \to g^{\circ\ell}(\rho_x^0)$ not only if $M\to\infty$ but also if $\ell\to\infty$ with finite M, where $g^{\circ k}$ denotes composition of the function k times. Indeed $g(x)$ is a monotonically increasing function with $g(x)\geq x$ and a single fixed point at $x=1$, so we have $\mathbb{E}[\rho_x^\ell]\to 1$ and $g^{\circ\ell}(\rho_x^0)\to 1$ if $\ell\to\infty$. In other words, if $\mathbb{E}[\rho_x^\ell]/g^{\circ\ell}(\rho_x^0) = 1 + c_\ell/M$ for some coefficients c_ℓ, then $c_\ell\to 0$ as $\ell\to\infty$. Therefore, we can replace $\mathbb{E}[\rho_x^\ell]$ with $g^{\circ\ell}(\rho_x^0)$ in the above bounds to obtain the infinite-depth-and-width limit.

Putting everything together, we can write the following bound for the expectation of a non-diagonal element of the NTK

$$\lim_{\substack{L\to\infty, M\to\infty\\ L/M\to\lambda\in\mathbb{R}}}[\Theta(x,\tilde{x})] \geq \lim_{\substack{L\to\infty, M\to\infty\\ L/M\to\lambda\in\mathbb{R}}}\Bigg[\|\mathbf{x}^0\|\|\tilde{\mathbf{x}}^0\|a^{L-1}\sum_{\ell=1}^{L}\frac{n_{\ell-1}}{n_0}g^{\circ\ell-1}(\rho_x^0)\prod_{k=\ell}^{L-1}f\big(g^{\circ(k-1)}(\rho_x^0)\big) + \sum_{\ell=1}^{L}a^{L-\ell}\prod_{k=\ell}^{L-1}f\big(g^{\circ(k-1)}(\rho_x^0)\big)\Bigg],$$

Now studying the expressions above we can find the following bounds:

$$1 \geq \lim_{\substack{L\to\infty, M\to\infty \\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta(x,\tilde{x})]}{\mathbb{E}[\Theta(x,x)]} \geq \frac{1}{4},$$

The upper bound is trivial. We obtain the lower bound in case of initialization in the chaotic phase by noticing that $\sum_{\ell=1}^{L} g^{\circ\ell-1}(\rho_x^0) \prod_{k=\ell}^{L-1} f\big(g^{\circ(k-1)}(\rho_x^0)\big) \geq L/4$ for $L \geq 2$, which, by Chebyshev's sum inequality, gives the maximal ratio between diagonal and non-diagonal elements of the NTK, since $\mathbb{E}[\Theta_W(x,x)] = a^{L-1} \sum_{\ell=1}^{L} \frac{n_{\ell-1}}{n_0}$. In the ordered phase, we have $\sum_{\ell=1}^{L} \prod_{k=\ell}^{L-1} f\big(g^{\circ(k-1)}(\rho_x^0)\big) \geq L/4$ and $\mathbb{E}[\Theta_b(x,x)] = \sum_{\ell=1}^{L} a^{L-\ell}$, which gives the same bound.

Moreover, it is easy to see that $\mathbb{E}[\Theta^2(x,\tilde{x})] \leq \mathbb{E}[\Theta^2(x,x)]$, therefore we can write

$$\lim_{\substack{L\to\infty, M\to\infty \\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta^2(x,\tilde{x})]}{\mathbb{E}^2[\Theta(x,\tilde{x})]} \leq 16 \lim_{\substack{L\to\infty, M\to\infty \\ L/M\to\lambda\in\mathbb{R}}} \frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]}$$

□

A.3. Training dynamics of the NTK

Theorem A.10 (GD step of the NTK)**.** *Consider a fully-connected DNN of depth L defined in (3) initialized as in (6). The input dimension is given by $n_0 = \alpha_0 M$, the output dimension is 1, and the hidden layers have constant width M. The activation function in the hidden layers is ReLU, i.e. $\phi(x) = x\mathbb{1}\{x > 0\}$, and the output layer is linear. Assume also that the biases are initialized to zero, i.e. $\sigma_b = 0$, and the input data is normalized. Then, if we perform a GD step on a point $(x, y) \in \mathcal{D}$ with learning rate η, the following holds for the changes of the corresponding element of the NTK:*

1. *In the **chaotic phase** ($a := \sigma_w^2/2 > 1$), the changes to the NTK value are infinite in the limit for a constant learning rate:*

$$\frac{\mathbb{E}[\Delta\Theta(x,x)]}{\mathbb{E}[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty, \\ L/M\to\lambda\in\mathbb{R}}]{} \infty \tag{43}$$

   The scaling of the learning rate needed to avoid the infinite limit is given by $\eta = O(a^{-L})$, which tends to zero with depth.

2. *In the **ordered phase** ($a < 1$), the NTK stays constant in the limit:*

$$\frac{\mathbb{E}[\Delta\Theta(x,x)]}{\mathbb{E}[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty, \\ L/M\to\lambda\in\mathbb{R}}]{} 0 \tag{44}$$

Proof. A derivative of the NTK in gradient flow can be expanded as follows:

$$\dot{\Theta}(x,x) = \sum_{\ell=1}^{L} \Big(\sum_{i,j} \frac{\partial\Theta(x,x)}{\partial \mathbf{W}_{ij}^{\ell}} \dot{\mathbf{W}}_{ij}^{\ell} + \sum_{i} \frac{\partial\Theta(x,x)}{\partial \mathbf{b}_{i}^{\ell}} \dot{\mathbf{b}}_{i}^{\ell} \Big),$$

where the parameters change in the direction of the negative gradient:

$$\dot{\mathbf{W}}_{ij}^{\ell} = -\frac{\partial\mathcal{L}(\mathcal{D})}{\partial \mathbf{W}_{ij}^{\ell}}, \quad \dot{\mathbf{b}}_{i}^{\ell} = -\frac{\partial\mathcal{L}(\mathcal{D})}{\partial \mathbf{b}_{i}^{\ell}}, \quad i = 1, \dots n_\ell,\ j = 1, \dots, n_{\ell-1},\ \ell = 1, \dots L$$

If we now assume that the gradient descent step is performed on a single point of the dataset x, which is the same point for which the NTK is calculated, we have:

$$\dot{\mathbf{W}}_{ij}^{\ell} = -\frac{\partial\mathcal{L}(x)}{\partial \mathbf{W}_{ij}^{\ell}} = -\frac{\partial\mathcal{L}(x)}{\partial f(x)} \frac{\partial f(x)}{\partial \mathbf{W}_{ij}^{\ell}} = -\frac{\partial\mathcal{L}(x)}{\partial f(x)} \boldsymbol{\delta}_i^{\ell} \mathbf{x}_j^{\ell-1},$$

$$\dot{\mathbf{b}}_{i}^{\ell} = -\frac{\partial\mathcal{L}(x)}{\partial \mathbf{b}_{i}^{\ell}} = -\frac{\partial\mathcal{L}(x}{\partial f(x)} \frac{\partial f(x)}{\partial \mathbf{b}_{i}^{\ell}} = -\frac{\partial\mathcal{L}(x)}{\partial f(x)} \boldsymbol{\delta}_i^{\ell}$$

It remains to calculate the derivatives of the NTK with resepect to the parameters. The involved terms are:

$$\frac{\partial\Theta_W(x,x)}{\partial\mathbf{W}^\ell_{ij}} = \sum_{\ell'}\sum_{i',j'}\frac{\partial}{\partial\mathbf{W}^\ell_{ij}}\Big(\frac{\partial f(x)}{\partial\mathbf{W}^{\ell'}_{i'j'}}\Big)^2 = 2\sum_{\ell'}\sum_{i',j'}\frac{\partial f(x)}{\partial\mathbf{W}^{\ell'}_{i'j'}}\frac{\partial^2 f(x)}{\partial\mathbf{W}^\ell_{ij}\mathbf{W}^{\ell'}_{i'j'}},$$

$$\frac{\partial\Theta_W(x,x)}{\partial\mathbf{b}^\ell_{k}} = \sum_{\ell'}\sum_{i',j'}\frac{\partial}{\partial\mathbf{b}^\ell_{k}}\Big(\frac{\partial f(x)}{\partial\mathbf{W}^{\ell'}_{i'j'}}\Big)^2 = 2\sum_{\ell'}\sum_{i',j'}\frac{\partial f(x)}{\partial\mathbf{W}^{\ell'}_{i'j'}}\frac{\partial^2 f(x)}{\partial\mathbf{b}^\ell_{k}\mathbf{W}^{\ell'}_{i'j'}},$$

$$\frac{\partial\Theta_b(x,x)}{\partial\mathbf{b}^\ell_{k}} = \sum_{\ell'}\sum_{i'}\frac{\partial}{\partial\mathbf{b}^\ell_{k}}\Big(\frac{\partial f(x)}{\partial\mathbf{b}^{\ell'}_{i'}}\Big)^2 = 2\sum_{\ell'}\sum_{i'}\frac{\partial f(x)}{\partial\mathbf{b}^{\ell'}_{i'}}\frac{\partial^2 f(x)}{\partial\mathbf{b}^\ell_{k}\mathbf{b}^{\ell'}_{i'}},$$

$$\frac{\partial\Theta_b(x,x)}{\partial\mathbf{W}^\ell_{ij}} = \sum_{\ell'}\sum_{i',j'}\frac{\partial}{\partial\mathbf{W}^\ell_{ij}}\Big(\frac{\partial f(x)}{\partial\mathbf{b}^{\ell'}_{i'}}\Big)^2 = 2\sum_{\ell'}\sum_{i'}\frac{\partial f(x)}{\partial\mathbf{b}^{\ell'}_{i'}}\frac{\partial^2 f(x)}{\partial\mathbf{W}^\ell_{ij}\mathbf{b}^{\ell'}_{i'}}$$

To calculate these terms, we need to find the second derivatives of the DNN's output function.

$$\frac{\partial^2 f(x)}{\partial\mathbf{W}^\ell_{ij}\mathbf{W}^{\ell'}_{i'j'}} = \boldsymbol{\delta}^{\ell'}_{i'}\frac{\partial\mathbf{x}^{\ell'-1}_{j'}}{\partial\mathbf{W}^\ell_{ij}} + \mathbf{x}^{\ell'-1}_{j'}\frac{\partial\boldsymbol{\delta}^{\ell'}_{i'}}{\partial\mathbf{W}^\ell_{ij}} = \mathbb{1}_{\ell<\ell'}\boldsymbol{\delta}^{\ell'}_{i'}\mathbf{x}^{\ell-1}_{j}\frac{\partial\mathbf{x}^{\ell'-1}_{j'}}{\partial\mathbf{h}^\ell_i} + \mathbb{1}_{\ell>\ell'}\boldsymbol{\delta}^{\ell}_{i}\mathbf{x}^{\ell'-1}_{j'}\phi'(\mathbf{h}^{\ell-1}_j)\frac{\partial\boldsymbol{\delta}^{\ell'}_{i'}}{\partial\boldsymbol{\delta}^{\ell-1}_j},$$

In the above equation the first term is non-zero only in case $\ell' > \ell$ and the second term is non-zero only if $\ell' < \ell$. Then we can write the following:

$$\begin{aligned}\sum_{\ell}\sum_{i,j}\frac{\partial\Theta_W(x,x)}{\partial\mathbf{W}^\ell_{ij}}\dot{\mathbf{W}}^\ell_{ij} = &-\frac{\partial\mathcal{L}(x)}{\partial f(x)}\sum_{\ell'>\ell}\|\boldsymbol{\delta}^{\ell'}\|^2\|\mathbf{x}^{\ell-1}\|^2\sum_i\boldsymbol{\delta}^\ell_i\frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}^\ell_i}\\ &-\frac{\partial\mathcal{L}(x)}{\partial f(x)}\sum_{\ell>\ell'}\|\boldsymbol{\delta}^{\ell}\|^2\|\mathbf{x}^{\ell'-1}\|^2\sum_j\mathbf{x}^{\ell-1}_j\frac{\partial\|\boldsymbol{\delta}^{\ell'}\|^2}{\partial\boldsymbol{\delta}^{\ell-1}_j}\end{aligned}$$

Opening the remaining parts of the derivative in the same way, we obtain the following expression:

$$\begin{aligned}\dot\Theta(x,x) = -\frac{\partial\mathcal{L}(x)}{\partial f(x)}\Bigg(&\sum_{\ell'>\ell}\big(\|\boldsymbol{\delta}^{\ell'}\|^2\|\mathbf{x}^{\ell-1}\|^2 + \|\boldsymbol{\delta}^{\ell'}\|^2\big)\sum_i\boldsymbol{\delta}^\ell_i\frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}^\ell_i}\\ &+\sum_{\ell>\ell'}\big(\|\boldsymbol{\delta}^{\ell}\|^2\|\mathbf{x}^{\ell'-1}\|^2 + \|\boldsymbol{\delta}^{\ell}\|^2\big)\sum_j\mathbf{x}^{\ell-1}_j\frac{\partial\|\boldsymbol{\delta}^{\ell'}\|^2}{\partial\boldsymbol{\delta}^{\ell-1}_j}\Bigg)\end{aligned}$$

Case 1. Chaotic phase. Let us bound the change of the NTK by computing only the terms with $\ell' = \ell + 1$. In this case, $\frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}^\ell_i} = 2\mathbf{x}^\ell_i$. We then notice that $\sum_i\mathbf{x}^\ell_i\boldsymbol{\delta}^\ell_i = \sum_k\boldsymbol{\delta}^{\ell+1}_k\sum_i\mathbf{W}^{\ell+1}_{ki}\mathbf{x}^\ell_i = \sum_k\boldsymbol{\delta}^{\ell+1}_k\mathbf{h}^{\ell+1}_k = \sum_k\boldsymbol{\delta}^{\ell+1}_k\mathbf{x}^{\ell+1}_k$, which by induction gives $\sum_i\mathbf{x}^\ell_i\boldsymbol{\delta}^\ell_i = f(x)$. Therefore, taking into account that for quadratic loss we have $\partial\mathcal{L}(x)/\partial f(x) = f(x) - y$, we can write the following bound:

$$\mathbb{E}[|\Delta\Theta(x,x)|] \geq 2\eta\mathbb{E}\Big[f(x)^2\sum_{\ell=1}^{L}\big(\|\boldsymbol{\delta}^{\ell+1}\|^2\|\mathbf{x}^{\ell-1}\|^2 + \|\boldsymbol{\delta}^{\ell+1}\|^2\big)\Big]$$

Then, using the results of Lemmas A.1, A.2 and A.4 again, we obtain the expectation of the first part:

$$\mathbb{E}[|\Delta\Theta(x,x)|] \geq 4\eta\frac{\|\mathbf{x}^0\|^4}{n_0}a^{2L-1}\sum_{\ell=1}^{L}\prod_{j=\ell+1}^{L-1}\Big(1+\frac{1}{n_j}+O\Big(\frac{1}{M^{3/2}}\Big)\Big)\prod_{i=1}^{\ell-1}\Big(1+\frac{5}{n_j}\Big) \propto a^{2L-1}$$

And since in case of the chaotic phase $\mathbb{E}[\Theta(x,x)] \propto \mathbb{E}[\Theta_W(x,x)] \propto a^L LM/n_0$, we have the desired limit:

$$\frac{\mathbb{E}[|\Delta\Theta(x,x)|]}{\mathbb{E}[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} \infty$$

Case 2. Ordered phase. The bound that we used for the chaotic phase above gives zero in the limit in case of the ordered phase. We will now show that the upper bound of the relative change of the NTK is also zero in the limit in this case. We notice that $\sum_i f(x)\boldsymbol{\delta}_i^\ell \frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}_i^\ell} = \sum_i \boldsymbol{\delta}_i^\ell \sum_k \boldsymbol{\delta}_k^\ell \frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}_i^\ell}\mathbf{h}_k^\ell = \sum_i (\boldsymbol{\delta}_i^\ell)^2 \frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}_i^\ell}\mathbf{h}_i^\ell + \sum_{i\neq k}\boldsymbol{\delta}_i^\ell\boldsymbol{\delta}_k^\ell \frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}_i^\ell}\mathbf{h}_k^\ell$. Then we have $\sum_i f(x)\boldsymbol{\delta}_i^\ell \frac{\partial\|\mathbf{x}^{\ell'-1}\|^2}{\partial\mathbf{h}_i^\ell} \leq \|\boldsymbol{\delta}^\ell\|^2\|\mathbf{x}^{\ell'-1}\|^2 + A$, where the expectation of A is zero. Similarly, we have $\sum_j f(x)\mathbf{x}_j^{\ell-1}\frac{\partial\|\boldsymbol{\delta}^{\ell'}\|^2}{\partial\boldsymbol{\delta}_j^{\ell-1}} \leq \|\boldsymbol{\delta}^{\ell'}\|^2\|\mathbf{x}^{\ell-1}\|^2 + B$ with a term B of zero expectation. The we have the following bound for the change of the NTK:

$$\mathbb{E}[|\Delta\Theta(x,x)|] \leq 2\eta\mathbb{E}\Big[\sum_{\ell_1}\|\boldsymbol{\delta}^{\ell_1}\|^2\|\mathbf{x}^{\ell_1-1}\|^2\sum_{\ell_2<\ell_1}\|\boldsymbol{\delta}^{\ell_2}\|^2\big(\|\mathbf{x}^{\ell_2}\|^2+1\big)\Big]$$

The expectation of $\sum_{\ell_2<\ell_1}\theta_W^{\ell_1}\theta_W^{\ell_2}$, where $\theta_W^\ell := \|\boldsymbol{\delta}^\ell\|^2\|\mathbf{x}^{\ell-1}\|^2$, was calculated in Lemma A.5 and the expectation of $\sum_{\ell_2<\ell_2}\theta_W^{\ell_1}\theta_b^{\ell_2}$, where $\theta_b^\ell := \|\boldsymbol{\delta}^\ell\|^2$, was calculated in Lemma A.7. In particular, we have the following results for the two sums:

$$\mathbb{E}[\sum_{\ell_2<\ell_1}\theta_W^{\ell_1}\theta_W^{\ell_2}] \propto a^{2L}\frac{L^2}{\alpha_0^2}\frac{1}{4\lambda}e^{5\lambda}\left(1-\frac{1}{4\lambda}(1-e^{-4\lambda})\right),$$
$$\mathbb{E}[\sum_{\ell_2<\ell_1}\theta_W^{\ell_1}\theta_b^{\ell_2}] \propto \frac{a^{2L}}{a-1}\frac{L}{\alpha_0}\frac{1}{4\lambda}e^{5\lambda}(1-e^{-4\lambda}).$$

Then we see that the upper bound on the changes of the NTK is proportional to $a^{2L}L^2$, which tends to zero with depth in the ordered phase. Given that the expectation of the NTK in the ordered phase has a non-zero limit given by $1/(1-a)$, we can then conclude that

$$\frac{\mathbb{E}[\Delta\Theta(x,x)]}{\mathbb{E}[\Theta(x,x)]} \xrightarrow[\substack{M\to\infty, L\to\infty,\\ L/M\to\lambda\in\mathbb{R}}]{} 0$$

in case of initialization in the ordered phase. □

B. Additional observations

B.1. Effects of $\alpha_0 := n0/M$ at the EOC

The theoretical expression for the NTK dispersion in the infinite-width limit, which we derived in Theorem 3.1, depends on the ratio $\alpha_0 := n_0/M$ at the EOC:

$$V_{\text{EOC}} := \frac{\mathbb{E}[\Theta^2(x,x)]}{\mathbb{E}^2[\Theta(x,x)]} \to \frac{1}{(1+\alpha_0)^2}\left[e^{5\lambda}\Big(\frac{1}{2\lambda}+\frac{2\alpha_0^2-8\alpha_0}{25\lambda^2}\Big)+(e^\lambda-e^{5\lambda})\frac{1-4\alpha_0}{8\lambda^2}+\frac{2\alpha_0}{5\lambda}\Big(\frac{4-\alpha_0}{5\lambda}-1-\alpha_0\Big)\right].$$

Examining this expression, one can see that it tends to the limiting expression for the NTK dispersion in the chaotic phase as the ratio α_0 decreases:

$$V_{\text{EOC}} \xrightarrow[\alpha_0\to 0]{} \frac{1}{2\lambda}e^{5\lambda}\left(1-\frac{1}{4\lambda}(1-e^{-4\lambda})\right).$$

We illustrate this effect in Figure 5. One can see that gradually decreasing the value of α_0 moves the NTK dispersion at the EOC closer to the NTK dispersion in the chaotic phase.

B.2. Effects of the architecture

In Section 3.2, we showed that constant-width DNNs that increase the input dimensionality, i.e. $n_0 < n_1 = \cdots = n_{L-1}$, get more robust with depth in a sense that the dispersion of their NTK decreases. Here we show how the theoretical expressions

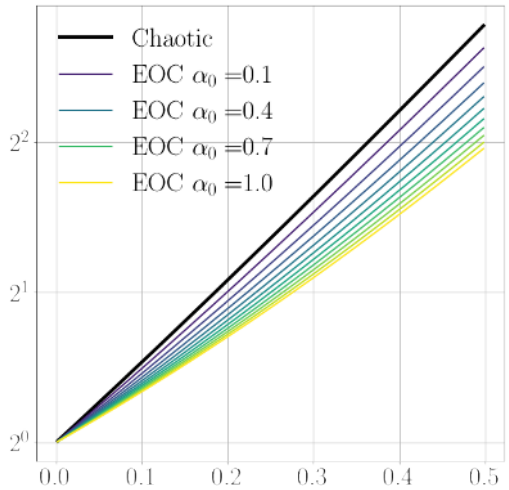


Figure 5. Effects of $\alpha_0 := n0/M$ on the NTK dispersion at the EOC in the infinite-depth-and-width limit. All the lines show the theoretical expressions from Theorem 3.1. The black line (uppermost) corresponds to the NTK dispersion in the chaotic phase, while all the other lines show the NTK dispersion at the EOC with varying α_0 values. The colors spanning from yellow to violet (from lighter to darker tones) indicate the value of α_0 spanning from 1 (yellow) to 0.1 (violet).

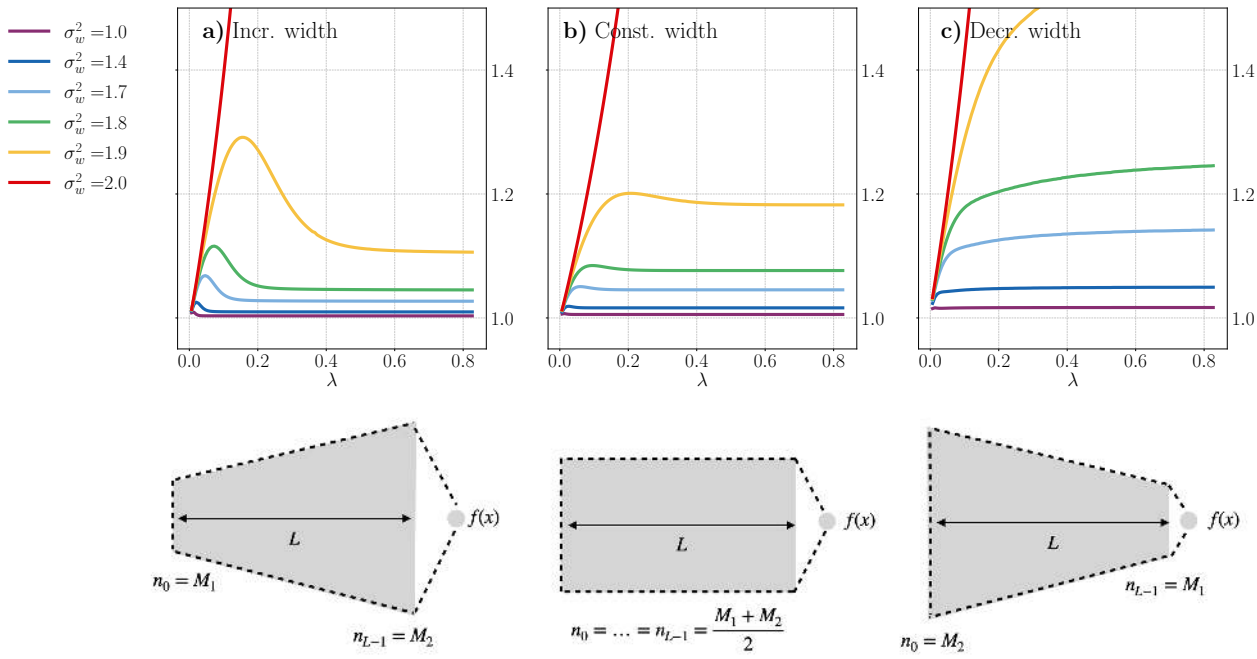


Figure 6. Effects of the architecture on the NTK dispersion ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ as predicted by Theorem 3.2. The subplots show the dispersion for varying values of σ_w^2 for three different architectures. The lower row of the figure illustrates the considered architectures. Formally, the widths for each architecture are given by: **a)** $n_\ell = M_1 + \lceil \ell(M_2 - M_1)/L \rceil$, **b)** $n_\ell = \lceil (M_1 + M_2)/2 \rceil$, **c)** $n_\ell = M_2 + \lceil \ell(M_1 - M_2)/L \rceil$ for $0 \leq \ell \leq L$. The width parameters are given by $M_1 = 100$, $M_2 = 500$ and do not change as the depth grows. The depth-to-width ratio is computed for the average width, i.e. $\lambda = 2L/(M_1 + M_2)$.

for the NTK moments in Theorem 3.2 reveal more effects of the DNN's architecture. In Figure 6, we compute the theoretical prediction of the NTK dispersion for three architectures: the first one gradually increases width over L layers from $n_0 = M_1$ to $n_{L-1} = M_2$, the second one keeps constant width, i.e. $n_0 = \cdots = n_{L-1} = (M_1 + M_2)/2$, and the third one gradually decreases width over L layers from $n_0 = M_2$ to $n_{L-1} = M_1$. We note that all the architectures have the same average width in this setting. We also note that we keep M_1 and M_2 fixed while varying the depth L. Therefore, we compare networks that increase or decrease the dimensionality equally but over a different number of layers. Figure 6 demonstrates that the NTK dispersion is lower for the DNNs that increase the dimensionality. Moreover, for such DNNs the peak of dispersion falls on the relatively shallow networks. Therefore, it may be beneficial to increase dimensionality over more layers if the goal is to decrease the variance of the DNN. On the contrary, the dispersion only increases with depth for DNNs that decrease the dimensionality. Thus, it may be beneficial to keep such networks shallow if one wants to keep the variance minimal.

B.3. Lazy training

The NTK regime of neural networks is often discussed in connection with the so-called lazy training phenomenon (Chizat et al., 2019). In lazy training, a model behaves as its linearization around the initialial parameters due to rescaling given by:

$$\tilde{f}_\alpha(x) = \alpha f(x), \qquad \tilde{\mathcal{L}}_\alpha(\mathcal{D}) = \frac{1}{\alpha^2}\mathcal{L}(\mathcal{D}), \quad \alpha \in \mathbb{R},$$

where $f(\cdot)$ is the original model's output function and $\mathcal{L}(\mathcal{D})$ is the training loss. Chizat et al. (2019) showed that the dynamics of a rescaled model defined by $\tilde{f}_\alpha(\cdot)$ and $\tilde{\mathcal{L}}_\alpha(\mathcal{D})$ is close to its linearization if the scaling factor α is large. Thus, in this section we discuss the effects of the lazy training rescaling on the results presented in our paper.

One can see that the NTK changes trivially if we rescale the output function:

$$\nabla_{\mathbf{w}} \tilde{f}_\alpha(x) = \alpha \nabla_{\mathbf{w}} f(x) \Rightarrow \tilde{\Theta}_\alpha(x_1, x_2) = \alpha^2 \Theta(x_1, x_2),$$

where we denoted the NTK of the rescaled model as $\tilde{\Theta}_\alpha$. Therefore, all our results concerning the NTK dispersion (Theorem 3.1) and the ratios of expectations at initialization (Theorem 3.3) do not change if we rescale the model, since the constants added to the nominator and the denominator cancel each other:

$$\frac{\mathbb{E}[\tilde{\Theta}^2_\alpha(x_1, x_2)]}{\mathbb{E}^2[\tilde{\Theta}_\alpha(x_1, x_2)]} = \frac{\mathbb{E}[\Theta^2(x_1, x_2)]}{\mathbb{E}^2[\Theta(x_1, x_2)]}.$$

On the other hand, the relative change of the NTK in a gradient descent step (Theorem 4.1) is affected by the rescaling. Recall that the NTK derivative is given by:

$$\dot{\Theta}(x, x) = \sum_{\ell=1}^{L} \Big(\sum_{i,j} \frac{\partial \Theta(x, x)}{\partial \mathbf{W}^\ell_{ij}} \dot{\mathbf{W}}^\ell_{ij} + \sum_i \frac{\partial \Theta(x, x)}{\partial \mathbf{b}^\ell_i} \dot{\mathbf{b}}^\ell_i \Big).$$

Terms of the above expression change as follows due to the rescaling:

$$\frac{\partial \tilde{\Theta}_\alpha(x, x)}{\partial \mathbf{W}^\ell_{ij}} = \alpha^2 \frac{\partial \Theta(x, x)}{\partial \mathbf{W}^\ell_{ij}}, \quad \frac{\partial \tilde{\Theta}_\alpha(x, x)}{\partial \mathbf{b}^\ell_i} = \alpha^2 \frac{\partial \Theta(x, x)}{\partial \mathbf{b}^\ell_i},$$

$$\big(\tilde{\dot{\mathbf{W}}}^\ell_{ij}\big)_\alpha = -\frac{\partial \tilde{\mathcal{L}}_\alpha(\mathcal{D})}{\partial \mathbf{W}^\ell_{ij}} = -\frac{1}{\alpha^2}\frac{\partial \mathcal{L}(\mathcal{D})}{\partial \mathbf{W}^\ell_{ij}} = -\frac{1}{\alpha^2}\tilde{\dot{\mathbf{W}}}^\ell_{ij}, \quad \big(\tilde{\dot{\mathbf{b}}}^\ell_i\big)_\alpha = \frac{1}{\alpha^2}\dot{\mathbf{b}}^\ell_i.$$

Therefore, the NTK derivative is not changed by the rescaling and for the relative change of the NTK we have:

$$\frac{\mathbb{E}[\Delta \tilde{\Theta}_\alpha(x, x)]}{\mathbb{E}[\tilde{\Theta}_\alpha(x, x)]} = \frac{1}{\alpha^2}\frac{\mathbb{E}[\Delta \Theta(x, x)]}{\mathbb{E}[\Theta(x, x)]}.$$

For any constant choice of $\alpha \in \mathbb{R}$, this scaling does not change our limiting results in Theorem 4.1. However, if the rescaling parameter is scaled exponentially with L as $\alpha \propto (\sigma_w/\sqrt{2})^L$ then the relative change of the NTK tends to zero in the chaotic phase, as well as in the ordered phase and at the EOC. Thus, it is possible to enforce lazy training in deep networks outside of the ordered phase but the required rescaling parameter grows exponentially with depth.

C. Additional experiments

C.1. Non-diagonal elements of the NTK

We provide empirical results on the dispersion of the non-diagonal elements of the NTK in this subsection. Figure 7 is analogous to Figure 1: it compares the dispersion of the NTK in different phases of initialization. One can see that the non-diagonal elements of the NTK behave similarly to the diagonal ones. In particular, the dispersion of the non-diagonal elements grows exponentially with the depth-to-width ratio and reaches high values for deep networks in the chaotic phase and at the EOC, whereas in the ordered phase the dispersion is low and does not grow with depth. Figure 8 is analogous to Figure 2: it characterizes the behavior of the NTK dispersion close to the EOC, where the finite-width effects are significant. Here the picture is again similar to the one described in Section 3.2 for the diagonal elements of the NTK. In particular, the dispersion gradually increases as σ_w^2 grows and approaches the EOC. One can also see that the dispersion of the non-diagonal elements decreases with depth in the ordered phase for networks with $\alpha_0 < 1$, same as in the case of the diagonal elements. In all the figures, we provide experiments for varying initial angles between the two input samples of the NTK and conclude that the dispersion does not depend significantly on this angle.

C.2. Additional error bars for Figures 1, 2 and 3

To keep all the figures readable, we include error bars only in a subset of points in Figures 1 and 2 in the main text. We also omit error bars in Figure 3. To give the reader a better idea about the variance observed in our experiments, we include additional figures with continuous error bars in this section. Figure 9 is analogous to Figure 1: it shows the estimated dispersion of the NTK along with the theoretical expressions in the infinite-depth-and-width limit from Theorem 3.1. We include fewer lines (values of σ_w^2) in this figure to keep the continuous error bars distinguishable. Similarly, Figure 10 is analogous to Figure 2: it shows the results concerning the NTK dispersion around the EOC. Finally, Figure 11 shows a subset of lines from Figure 3 with their continuous error bars and concerns the ratio between non-diagonal and diagonal elements of the NTK.

C.3. Estimating the NTK dispersion from a sample

In our experiments, we estimate the ratio $r := \mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ at initialization from a sample. To do so, we sample an element of the NTK N times with independently chosen intialization parameters and get a sample $\{\theta_i\}_{i=1}^N$.

In this setting, the standard estimators for the first and the second moments given by $\hat{\mu}_1 := \sum_{i=1}^N \theta_i/N$ and $\hat{\mu}_2 := \sum_{i=1}^N \theta_i^2/N$ are unbiased:

$$\mathbb{E}[\hat{\mu}_1] = \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^N \theta_i\Big] = \frac{1}{N}\sum_{i=1}^N \mathbb{E}[\theta_i] = \mu_1,$$
$$\mathbb{E}[\hat{\mu}_2] = \mathbb{E}\Big[\frac{1}{N}\sum_{i=1}^N \theta_i^2\Big] = \frac{1}{N}\sum_{i=1}^N \mathbb{E}[\theta_i^2] = \mu_2,$$

where we denoted the actual moments as $\mu_1 := \mathbb{E}[\theta_i]$ and $\mu_2 := \mathbb{E}[\theta_i^2]$.

However, $\hat{\mu}_1$ and $\hat{\mu}_2$ computed on the same sample are dependent, so the estimator for the desired ratio given by $\hat{\mu}_2/(\hat{\mu}_1)^2$ is biased and we need to correct it. First, we notice that the estimator for μ_1^2 given by the square of $\hat{\mu}_1$ is biased as follows:

$$\mathbb{E}[\hat{\mu}_1^2] = \mathbb{E}\Big[\frac{1}{N^2}\Big(\sum_{i=1}^N \theta_i\Big)^2\Big] = \frac{\mu_2}{N} + \frac{N-1}{N}\mu_1^2$$

Therefore, an unbiased estimator for μ_1^2 can be computed as

$$\widehat{(\mu_1^2)} = \frac{N}{N-1}(\hat{\mu}_1^2 - \frac{1}{N}\hat{\mu}_2)$$

Second, we want to remove the dependence between the numerator and the denominator, which can be done simply by

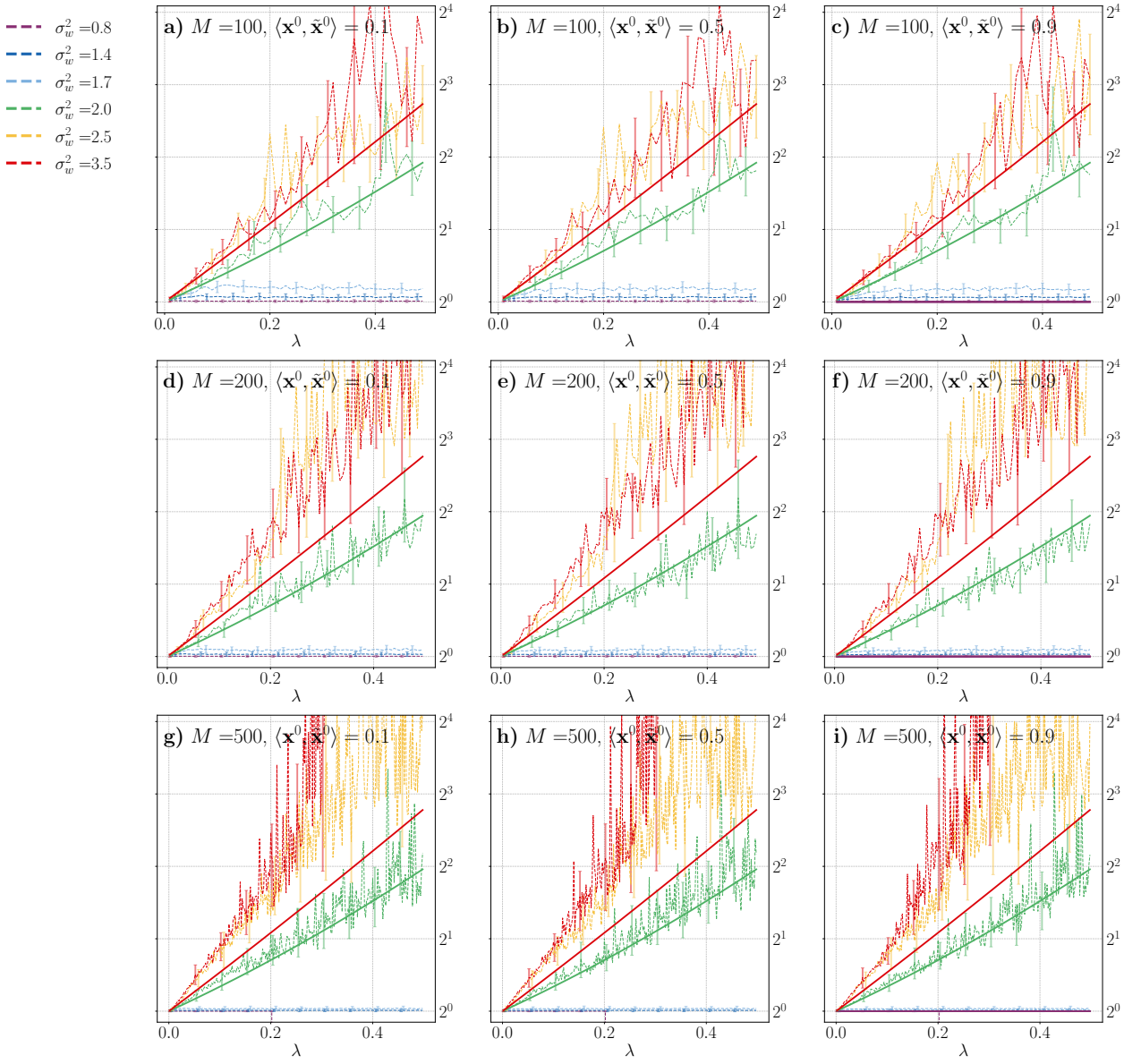


Figure 7. Ratio $\mathbb{E}[\Theta^2(x, \tilde{x})]/\mathbb{E}^2[\Theta(x, \tilde{x})]$ at initialization (on a pair of different input samples, i.e. $x \neq \tilde{x}$) for fully-connected ReLU networks of constant width $M = 200$ with $\alpha_0 \in \{2.0, 0.5, 0.1\}$ and varying initial angle between the input samples $\langle \mathbf{x}^0, \tilde{\mathbf{x}}^0 \rangle \in \{0.1, 0.5, 0.9\}$. The dashed lines show the experimental results and the solid lines show the corresponding theoretical predictions for diagonal elements of the NTK from Theorem 3.1. For each network configuration, we sampled 500 random initializations and computed an unbiased estimator for the ratio (see details in Appendix C.3). The error bars show the bootstrap estimation of the standard error.

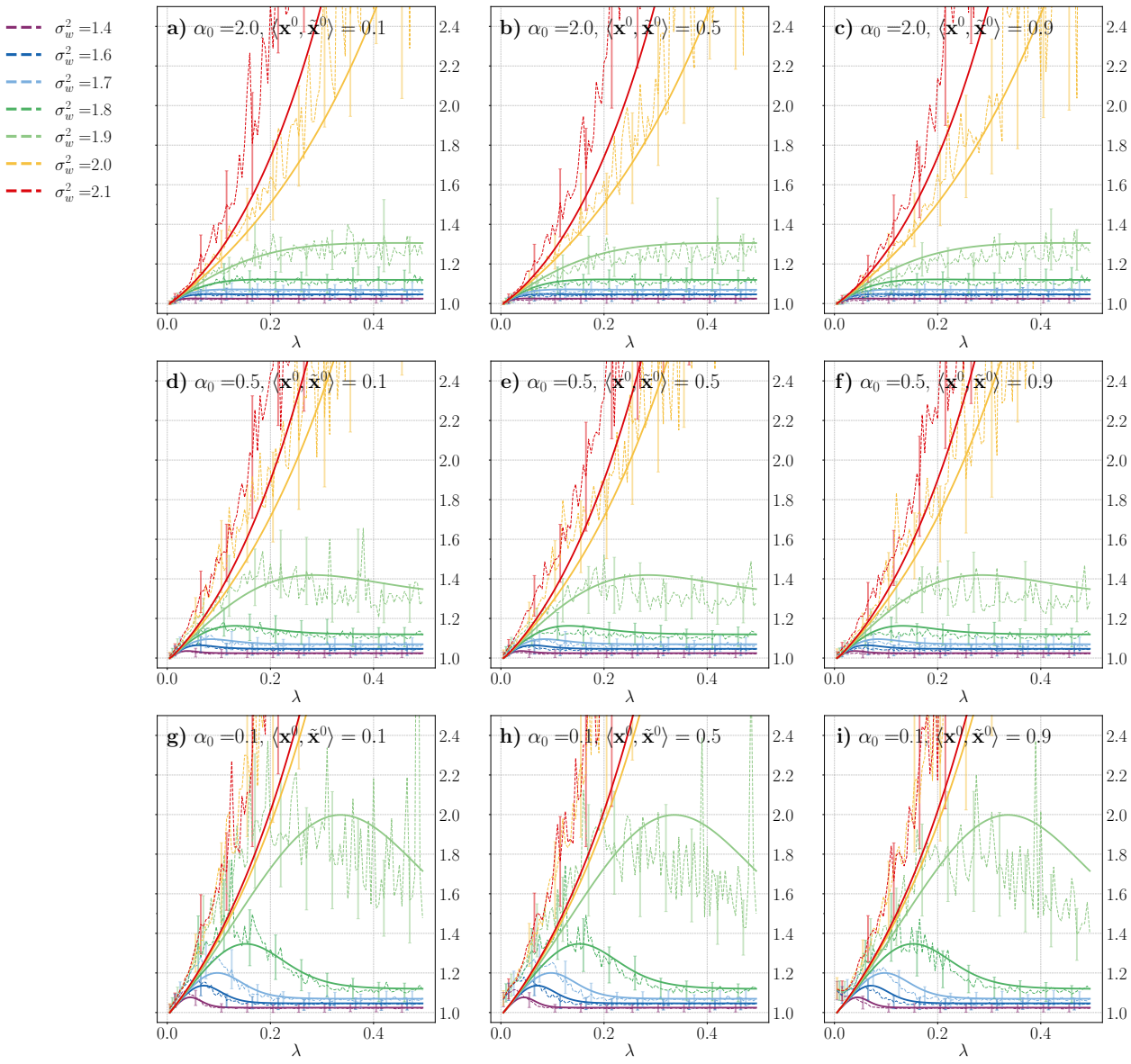


Figure 8. Ratio $\mathbb{E}[\Theta^2(x, \tilde{x})]/\mathbb{E}^2[\Theta(x, \tilde{x})]$ at initialization (on a pair of different input samples, i.e. $x \neq \tilde{x}$) for fully-connected ReLU networks of constant width $M = 200$ with $\alpha_0 \in \{2.0, 0.5, 0.1\}$ and varying initial angle between the input samples $\langle \mathbf{x}^0, \tilde{\mathbf{x}}^0 \rangle \in \{0.1, 0.5, 0.9\}$. The dashed lines show the experimental results and the solid lines show the theoretical predictions given by Theorem 3.2 for the diagonal elements of the NTK. For each network configuration, we sampled 500 random initializations and computed an unbiased estimator for the ratio (see details in Appendix C.3). The error bars show the bootstrap estimation of the standard error.

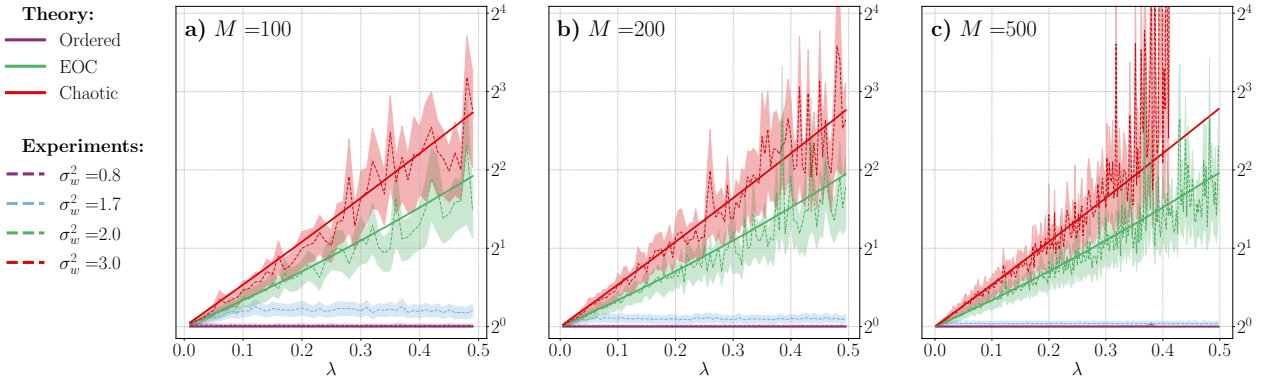


Figure 9. Ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ at initialization for fully-connected ReLU DNNs of constant width $M \in \{100, 200, 500\}$ with $\alpha_0 = 1$. The experiment setup is the same as in Figure 1. Continuous error bars show the bootstrap estimation of the standard error.

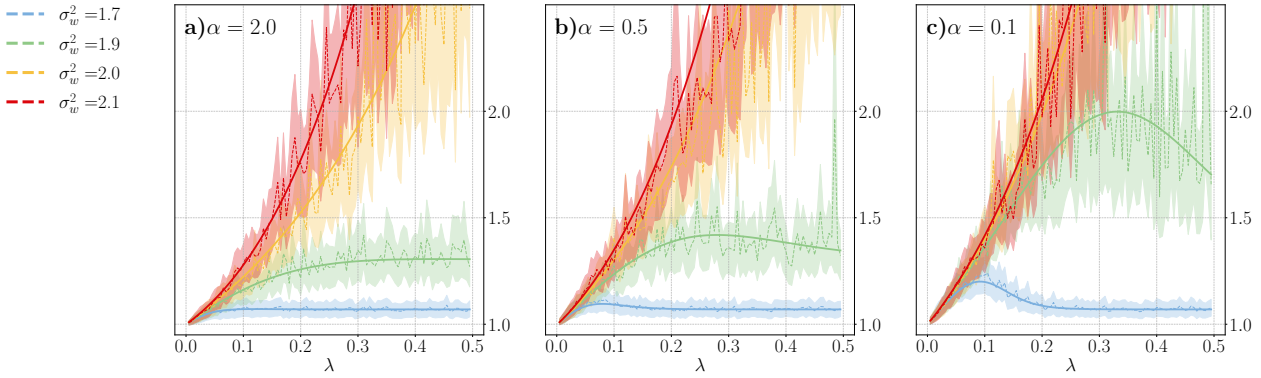


Figure 10. Ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ at initialization for fully-connected ReLU networks of constant width $M = 200$ with the ratio $\alpha_0 := n_0/M \in \{2.0, 0.5, 0.1\}$. The initialization hyperparameter σ_w^2 is close to the EOC for all the lines. The experiment setup is the same as in Figure 2. Continuous error bars show the bootstrap estimation of the standard error.

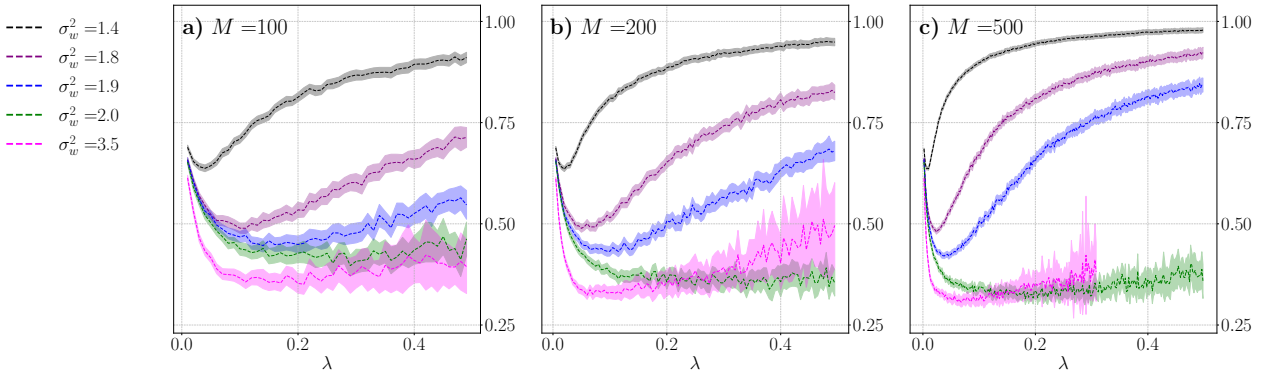


Figure 11. Ratio $\mathbb{E}[\Theta^2(x,x)]/\mathbb{E}^2[\Theta(x,x)]$ at initialization for fully-connected ReLU networks of constant width $M = 200$ with the ratio $\alpha_0 := n_0/M \in \{2.0, 0.5, 0.1\}$. The initialization hyperparameter σ_w^2 is close to the EOC for all the lines. The experiment setup is the same as in Figure 2. Continuous error bars show the bootstrap estimation of the standard error.

using disjoint parts of the sample to compute the two:

$$\hat{r} := \frac{1}{N-2} \sum_{i=1}^{N} \frac{\theta_i^2}{\frac{1}{(N-1)(N-2)} \left[\left(\sum_{j \neq i} \theta_j \right)^2 - \sum_{j \neq i} \theta_j^2 \right]},$$

where we used the unbiased version of the estimator for μ_1^2 computed on a sample without θ_i in the denominator. Then we can compute the expectation of our new estimator for the ratio as follows:

$$\mathbb{E}[\hat{r}] = (N-1) \sum_{i=1}^{N} \frac{\mathbb{E}[\theta_i^2]}{\mathbb{E}\left[\left(\sum_{j \neq i} \theta_j \right)^2 - \sum_{j \neq i} \theta_j^2 \right]} = \frac{\mu_2}{\mu_1^2}.$$

Therefore, $\hat{r}$ is an unbiased estimator of the ratio.

3.3 Neural (Tangent Kernel) Collapse

Contributing article: Seleznova, M., Weitzner, D., Giryes, R., Kutyniok, G., and Chou, H.-H. (2023). Neural (Tangent Kernel) Collapse. In *Advances in Neural Information Processing Systems*, volume 36. Curran Associates, Inc.

Author contributions: Mariia Seleznova developed the original idea to connect NTK aligment and Neural Collapse (NC), formulated the block-structure assumption on the NTK, and derived all the theorems and proofs presented in the paper. Mariia Seleznova designed and programmed all the numerical experiments, and produced all the figures. Mariia Seleznova wrote most of the paper's main text and the appendices, and managed the publication process: paper submission to the conference, writing a rebuttal after the initial reviews, addressing reviewers' concerns, and producing the camera-ready version of the paper. Hung-Hsu Chou proofread and helped to structure the theoretical part of the paper, and took part in the project discussions since the early stages. Dana Weitzner joined the project in the later stages and provided helpful references. Dana Weitzner and Hung-Hsu Chou contributed into writing the paper's main text. Raja Giryes and Gitta Kutyniok provided review and feedback.

Additional resources:

- Paper link: https://openreview.net/pdf?id=fyLvHzEssH
- Slides: https://neurips.cc/media/neurips-2023/Slides/70877.pdf
- Video presentation: https://slideslive.com/39010253
- Source code: https://github.com/mselezniova/ntk_collapse

Neural (Tangent Kernel) Collapse

Mariia Seleznova[1*] **Dana Weitzner**[2] **Raja Giryes**[2] **Gitta Kutyniok**[1] **Hung-Hsu Chou**[1]
[1]Ludwig-Maximilians-Universität München [2]Tel Aviv University

Abstract

This work bridges two important concepts: the Neural Tangent Kernel (NTK), which captures the evolution of deep neural networks (DNNs) during training, and the Neural Collapse (NC) phenomenon, which refers to the emergence of symmetry and structure in the last-layer features of well-trained classification DNNs. We adopt the natural assumption that the empirical NTK develops a block structure aligned with the class labels, i.e., samples within the same class have stronger correlations than samples from different classes. Under this assumption, we derive the dynamics of DNNs trained with mean squared (MSE) loss and break them into interpretable phases. Moreover, we identify an invariant that captures the essence of the dynamics, and use it to prove the emergence of NC in DNNs with block-structured NTK. We provide large-scale numerical experiments on three common DNN architectures and three benchmark datasets to support our theory.

1 Introduction

Deep Neural Networks (DNNs) are advancing the state of the art in many real-life applications, ranging from image classification to machine translation. Yet, there is no comprehensive theory that can explain a multitude of empirical phenomena observed in DNNs. In this work, we provide a theoretical connection between two such empirical phenomena, prominent in modern DNNs: *Neural Collapse (NC)* and *Neural Tangent Kernel (NTK) alignment*.

Neural Collapse. NC [39] emerges while training modern classification DNNs past zero error to further minimize the loss. During NC, the class means of the DNN's last-layer features form a symmetric structure with maximal separation angle, while the features of each individual sample collapse to their class means. This simple structure of the feature vectors appears favourable for generalization and robustness in the literature [12, 31, 40, 47]. Though NC is common in modern DNNs, explaining the mechanisms behind its emergence is challenging, since the complex non-linear training dynamics of DNNs evade analytical treatment.

Neural Tangent Kernel. The NTK [30] describes the gradient descent dynamics of DNNs in the function space, which provides a dual perspective to DNNs' evolution in the parameters space. This perspective allows to study the dynamics of DNNs analytically in the infinite-width limit, where the NTK is constant during training [30]. Hence, theoretical works often rely on the infinite-width NTK to analyze generalization of DNNs [1, 20, 28, 49]. However, multiple authors have argued that the infinite-width limit does not fully reflect the behaviour of realistic DNNs [2, 10, 22, 27, 36, 43], since constant NTK implies that no feature learning occurs during DNNs training.

NTK Alignment. While the infinite-width NTK is label-agnostic and does not change during training, the empirical NTK rapidly aligns with the target function in the early stages of training [5, 7, 44, 45]. In the context of classification, this manifests itself as the emergence of a block structure

*Correspondence to: Mariia Seleznova (selez@math.lmu.de).

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

in the kernel matrix, where the correlations between samples from the same class are stronger than between samples from different classes. The NTK alignment implies the so-called local elasticity of DNNs' training dynamics, i.e., samples from one class have little impact on samples from other classes in Stochastic Gradient Descent (SGD) updates [23]. Several recent works have also linked the local elasticity of training dynamics to the emergence of NC [33, 53]. This brings us to the main question of this paper: *Is there a connection between NTK alignment and neural collapse?*

Contribution. In this work, we consider a model of NTK alignment, where the kernel has a *block structure*, i.e., it takes only three distinct values: an inter-class value, an intra-class value and a diagonal value. We describe this model in Section 3. Within the model, we establish the connection between NTK alignment and NC, and identify the conditions under which NC occurs. Our main contributions are as follows:

- We derive and analyze the training dynamics of DNNs with MSE loss and block-structured NTK in Section 4. We identify three distinct convergence rates in the dynamics, which correspond to three components of the training error: error of the global mean, of the class means, and of each individual sample. These components play a key role in the dynamics.
- We show that NC emerges in DNNs with block-structured NTK under additional assumptions in Section 5.3. To the best of our knowledge, this is the first work to connect NTK alignment and NC. While previous contributions rely on the unconstrained features models [21, 38, 48] or other imitations of DNNs' training dynamics [53] to derive NC (see Appendix A for a detailed discussion of related works), we consider standard gradient flow dynamics of DNNs simplified by our assumption on the NTK structure.
- We analyze when NC does or does not occur in DNNs with NTK alignment, both theoretically and empirically. In particular, we identify an invariant of the training dynamics that provides a necessary condition for the emergence of NC in Section 5.2. Since DNNs with block-structured NTK do not always converge to NC, we conclude that NTK alignment is a more widespread phenomenon than NC.
- We support our theory with large-scale numerical experiments in Section 6.

2 Preliminaries

We consider the classification problem with $C \in \mathbb{N}$ classes, where the goal is to build a classifier that returns a class label for any input $x \in \mathcal{X}$. In this work, the classifier is a DNN trained on a dataset $\{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathcal{X}$ are the inputs and $y_i \in \mathbb{R}^C$ are the one-hot encodings of the class labels. We view the output function of the DNN $f : \mathcal{X} \to \mathbb{R}^C$ as a composition of parametrized last-layer *features* $h : \mathcal{X} \to \mathbb{R}^n$ and a linear *classification* layer parametrized by weights $\mathbf{W} \in \mathbb{R}^{C \times n}$ and biases $\mathbf{b} \in \mathbb{R}^C$. Then the logits of the training data $X = \{x_i\}_{i=1}^N$ can be expressed as follows:

$$f(X) = \mathbf{W}\mathbf{H} + \mathbf{b}\mathbf{1}_N^\top, \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{n \times N}$ are the features of the entire dataset stacked as columns and $\mathbf{1}_N \in \mathbb{R}^N$ is a vector of ones. Though we omit the notion of the data dependence in the text to follow, i.e. we write $\mathbf{H}$ without the explicit dependence on X, we emphasize that the features $\mathbf{H}$ are a function of the data and the DNN's parameters, unlike in the previously studied unconstrained feature models [21, 38, 48].

We assume that the dataset is *balanced*, i.e. there are $m := N/C$ training samples for each class. Without loss of generality, we further assume that the inputs are reordered so that $x_{(c-1)m+1}, \dots, x_{cm}$ belong to class c for all $c \in [C]$. This will make the notation much easier later on. Since the dimension of features n is typically much larger than the number of classes, we also assume $n > C$ in this work.

2.1 Neural Collapse

Neural Collapse (NC) is an empirical behaviour of classifier DNNs trained past zero error [39]. Let $\langle h \rangle := N^{-1} \sum_{i=1}^N h(x_i)$ denote the global features mean and $\langle h \rangle_c := m^{-1} \sum_{x_i \in \text{class } c} h(x_i)$, $c \in [C]$ be the class means. Furthermore, define the matrix of normalized centered class means as $\mathbf{M} := [\langle \overline{h} \rangle_1 / \|\langle \overline{h} \rangle_1\|_2, \dots, \langle \overline{h} \rangle_C / \|\langle \overline{h} \rangle_C\|_2]^\top \in \mathbb{R}^{n \times C}$, where $\langle \overline{h} \rangle_c = \langle h \rangle_c - \langle h \rangle, c \in [C]$. We say that a DNN exhibits NC if the following four behaviours emerge as the training time t increases:

(NC1) Variability collapse: for all samples x_i^c from class $c \in [C]$, where $i \in [m]$, the penultimate layer features converge to their class means, i.e. $\|h(x_i^c) - \langle h \rangle_c\|_2 \to 0$.

(NC2) Convergence to Simplex Equiangular Tight Frame (ETF): for all $c, c' \in [C]$, the class means converge to the following configuration:

$$\|\langle h \rangle_c - \langle h \rangle\|_2 - \|\langle h \rangle_{c'} - \langle h \rangle\|_2 \to 0, \quad \mathbf{M}^\top \mathbf{M} \to \frac{C}{C-1}(\mathbb{I}_C - \frac{1}{C}\mathbf{1}_C \mathbf{1}_C^\top).$$

(NC3) Convergence to self-duality: the class means $\mathbf{M}$ and the final weights $\mathbf{W}^\top$ converge to each other:

$$\left\| \mathbf{M}/\|\mathbf{M}\|_F - \mathbf{W}^\top/\|\mathbf{W}^\top\|_F \right\|_F \to 0.$$

(NC4) Simplification to Nearest Class Center (NCC): the classifier converges to the NCC decision rule behaviour:

$$\operatorname*{argmax}_c (\mathbf{W}h(x) + \mathbf{b})_c \to \operatorname*{argmin}_c \|h(x) - \langle h \rangle_c\|_2.$$

Though NC is observed in practice, there is currently no conclusive theory on the mechanisms of its emergence during DNN training. Most theoretical works on NC adopt the unconstrained features model, where features $\mathbf{H}$ are free variables that can be directly optimized [21, 38, 48]. Training dynamics of such models do not accurately reflect the dynamics of real DNNs, since they ignore the dependence of the features on the input data and the DNN's trainable parameters. In this work, we make a step towards realistic DNN dynamics by means of the Neural Tangent Kernel (NTK).

2.2 Neural Tangent Kernel

The NTK Θ of a DNN with the output function $f : \mathcal{X} \to \mathbb{R}^C$ and trainable parameters $\mathbf{w} \in \mathbb{R}^P$ (stretched into a single vector) is given by

$$\Theta_{k,s}(x_i, x_j) := \left\langle \nabla_{\mathbf{w}} f_k(x_i), \nabla_{\mathbf{w}} f_s(x_j) \right\rangle, \quad x_i, x_j \in \mathcal{X}, \quad k, s \in [C]. \tag{2}$$

We also define the last-layer features kernel Θ^h, which is a component of the NTK corresponding to the parameters up to the penultimate layer, as follows:

$$\Theta^h_{k,s}(x_i, x_j) := \left\langle \nabla_{\mathbf{w}} h_k(x_i), \nabla_{\mathbf{w}} h_s(x_j) \right\rangle, \quad x_i, x_j \in \mathcal{X}, \quad k, s \in [n]. \tag{3}$$

Intuitively, the NTK captures the correlations between the training samples in the DNN dynamics. While most theoretical works consider the infinite-width limit of DNNs [30, 52], where the NTK can be computed theoretically, empirical studies have also extensively explored the NTK of finite-width networks [19, 36, 45, 49]. Unlike the label-agnostic infinite-width NTK, the empirical NTK aligns with the labels during training. We use this observation in our main assumption (Section 3).

2.3 Classification with MSE Loss

We study NC for DNNs with the mean squared error (MSE) loss given by

$$\mathcal{L}(\mathbf{W}, \mathbf{H}, \mathbf{b}) = \frac{1}{2}\|f(X) - \mathbf{Y}\|_F^2, \tag{4}$$

where $\mathbf{Y} \in \mathbb{R}^{C \times N}$ is a matrix of stacked labels y_i. While NC was originally introduced for the cross-entropy (CE) loss [39], which is more common in classification problems, the MSE loss is much easier to analyze theoretically. Moreover, empirical observations suggest that DNNs with MSE loss achieve comparable performance to using CE [14, 29, 41], which motivates the recent line of research on MSE-NC [21, 38, 48].

3 Block Structure of the NTK

Numerous empirical studies have demonstrated that the NTK becomes aligned with the labels $\mathbf{Y}^\top \mathbf{Y}$ during the training process [7, 32, 45]. This alignment constitutes feature learning and is associated with better performance of DNNs [9, 13]. For classification problems, this means that the empirical

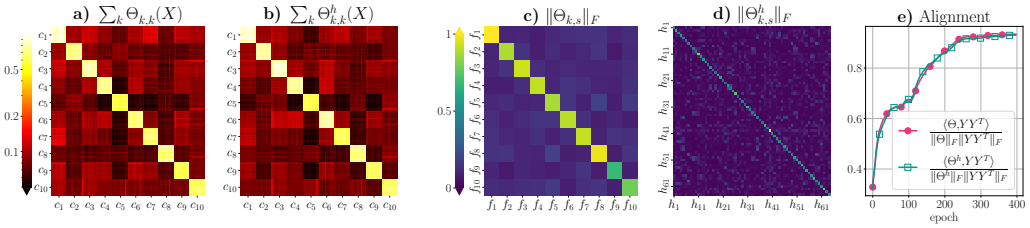


Figure 1: The NTK block structure of ResNet20 trained on MNIST. **a)** Traced kernel $\sum_{k=1}^{C} \Theta_{k,k}(X)$ computed on a random data subset with 12 samples from each class. The samples are ordered as described in Section 2, so that the diagonal blocks correspond to pairs of inputs from the same class. **b)** Traced kernel $\sum_{k=1}^{n} \Theta^h_{k,k}(X)$ computed on the same subset. **c)** Norms of the kernels $\Theta_{k,s}(X)$ for all $k, s \in [C]$. **d)** Norms of the kernels $\Theta^h_{k,s}(X)$ for all $k, s \in [n]$. The color bars show the values in each heatmap as a fraction of the maximal value in the heatmap. **e)** The alignment of the traced kernels from panes **a** and **b** with the class labels.

NTK develops an approximate block structure with larger kernel values corresponding to pairs of samples (x_i^c, x_j^c) from the same class [44]. Figure 1 shows an example of such a structure emergent in the empirical NTK of ResNet20 trained on MNIST.[2] Motivated by these observations, we assume that the NTK and the last-layer features kernel exhibit a block structure, defined as follows:

Definition 3.1 (Block structure of a kernel)**.** *We say a kernel $\Theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{K \times K}$ has a block structure associated with $(\lambda_1, \lambda_2, \lambda_3)$, if $\lambda_1 > \lambda_2 > \lambda_3 \geq 0$ and*

$$\Theta(x, x) = \lambda_1 \mathbb{I}_K, \quad \Theta(x_i^c, x_j^c) = \lambda_2 \mathbb{I}_K, \quad \Theta(x_i^c, x_j^{c'}) = \lambda_3 \mathbb{I}_K, \tag{5}$$

where x_i^c and x_j^c are two distinct inputs from the same class, and $x_j^{c'}$ is an input from class $c' \neq c$.

Assumption 3.2. *The NTK $\Theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{C \times C}$ has a block structure associated with $(\gamma_d, \gamma_c, \gamma_n)$, and the penultimate kernel $\Theta^h : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{n \times n}$ has a block structure associated with $(\kappa_d, \kappa_c, \kappa_n)$.*

This assumption means that every kernel $\Theta_{k,k}(X) := [\Theta_{k,k}(x_i, x_j)]_{i,j \in [N]}$ corresponding to an output neuron $f_k, k \in [C]$ and every kernel $\Theta^h_{p,p}(X)$ corresponding to a last-layer neuron $h_p, p \in [n]$ is aligned with $\mathbf{Y}^\top \mathbf{Y}$ (see Figure 1, panes a-b). Additionally, the "non-diagonal" kernels $\Theta_{k,s}(X)$ and $\Theta^h_{k,s}(X)$, $k \neq s$ are equal to zero (see Figure 1, panes c-d).[3] Moreover, if $\gamma_c \gg \gamma_n$ and $\kappa_c \gg \kappa_n$, Assumption 3.2 can be interpreted as *local elasticity* of DNNs, defined below.

Definition 3.3 (Local elasticity [23])**.** *A classifier is said to be locally elastic (LE) if its prediction or feature representation on point x_i^c from class $c \in [C]$ is not significantly affected by performing SGD updates on data points from classes $c' \neq c$.*

To see the relation between Assumption 3.2 and this definition, consider a Gradient Descent (GD) step of the output neuron $f_k, k \in [C]$ with step size η performed on a single input $x_j^{c'}$ from class $c' \neq c$. By the chain rule, block-structured Θ implies locally-elastic predictions since

$$f^{t+1}(x_i^c) - f^t(x_i^c) = -\eta \Theta(x_i^c, x_j^{c'}) \frac{\partial \mathcal{L}(x_j^{c'})}{\partial f(x_j^{c'})} + O(\eta^2), \tag{6}$$

i.e., the magnitude of the GD step of $f(x_i^c)$ is determined by the value of $\Theta(x_i^c, x_j^{c'})$. Similarly, block-structured kernel Θ^h implies locally-elastic penultimate layer features because

$$h^{t+1}(x_i^c) - h^t(x_i^c) = -\eta \Theta^h(x_i^c, x_j^{c'}) \mathbf{W}^\top \frac{\partial \mathcal{L}(x_j^{c'})}{\partial f(x_j^{c'})} + O(\eta^2). \tag{7}$$

This observation provides a connection between our work and recent contributions suggesting a connection between NC and local elasticity [33, 53].

[2] We provide figures illustrating the NTK block structure on other architectures and datasets in Appendix C.

[3] We discuss possible relaxations to our main assumption, where the "non-diagonal" components of the last-layer kernel $\Theta^h_{k,s}$ are allowed to be non-zero, in Appendix D.

Eigenvalue	Eigenvector	Multiplicity
$\lambda_{\text{single}} = \gamma_d - \gamma_c$	$\mathbf{v}_i^c = \frac{1}{m-1}\big(\underbrace{m-1, -\mathbf{1}_{m-1}^\top}_{\text{index } i>0,\ \text{class } c<0}, \underbrace{\mathbf{0}_{N-m}^\top}_{\text{others } =0}\big)^\top$	$N - C$
$\lambda_{\text{class}} = \lambda_{\text{single}} + m(\gamma_c - \gamma_n)$	$\mathbf{v}_c = \frac{1}{C-1}\big(\underbrace{(C-1)\mathbf{1}_m^\top}_{\text{class } c>0}, \underbrace{-\mathbf{1}_{N-m}^\top}_{\text{others } <0}\big)^\top$	$C - 1$
$\lambda_{\text{global}} = \lambda_{\text{class}} + N\gamma_n$	$\mathbf{v}_0 = \mathbf{1}_N$	1

Table 1: Eigendecomposition of the block-structured NTK.

4 Dynamics of DNNs with NTK Alignment

4.1 Convergence

As a warm up for our main results, we analyze the effects of the NTK block structure on the convergence of DNNs. Consider a GD update of an output neuron $f_k, k \in [C]$ with the step size η:

$$f_k^{t+1}(X) = f_k^t(X) - \eta\Theta_{k,k}(X)(f_k^t(X) - \mathbf{Y}_k) + O(\eta^2), \quad k = 1, \dots, C. \tag{8}$$

Note that we have taken into account that $\Theta_{k,s}$ is zero for $k \neq s$ by our assumption. Denote the residuals corresponding to f_k as $\mathbf{r}_k^\top := f_k^\top(X) - \mathbf{Y}_k \in \mathbb{R}^N$. Then we have the following dynamics for the residuals vector:

$$\mathbf{r}_k^{t+1} = (1 - \eta\Theta_{k,k}(X))\mathbf{r}_k^t + O(\eta^2). \tag{9}$$

The eigendecomposition of the block-structured kernel $\Theta_{k,k}(X)$ provides important insights into this dynamics and is summarized in Table 1. We notice that the NTK has three distinct eigenvalues $\lambda_{\text{global}} \geq \lambda_{\text{class}} \geq \lambda_{\text{single}}$, which imply different convergence rates for certain components of the error. Moreover, the eigenvectors associated with each of these eigenvalues reveal the meaning of the error components corresponding to each convergence rate. Indeed, consider the projected dynamics with respect to eigenvector $\mathbf{v}_0$ and eigenvalue λ_{global} from Table 1:

$$\langle \mathbf{r}_k^{t+1}, \mathbf{v}_0 \rangle = (1 - \eta\lambda_{\text{global}})\langle \mathbf{r}_k^t, \mathbf{v}_0 \rangle, \tag{10}$$

where we omitted $O(\eta^2)$ for clarity. Now notice that the projection of $\mathbf{r}_k^t$ onto the vector $\mathbf{v}_0$ is in fact proportional to the average residual over the training set:

$$\langle \mathbf{r}_k^t, \mathbf{v}_0 \rangle = \langle \mathbf{r}_k^t, \mathbf{1}_N \rangle = N\langle \mathbf{r}_k^t \rangle \tag{11}$$

where $\langle \cdot \rangle$ denotes the average over all the training samples $x_i \in X$. By a similar calculation, for all $c \in [C]$ and $i \in [m]$ we get interpretations of the remaining projections of the residual:

$$\langle \mathbf{r}_k^t, \mathbf{v}_c \rangle = \frac{N}{C-1}(\langle \mathbf{r}_k^t \rangle_c - \langle \mathbf{r}_k^t \rangle), \quad \langle \mathbf{r}_k^t, \mathbf{v}_i^c \rangle = \frac{m}{m-1}(\mathbf{r}_k^t(x_i^c) - \langle \mathbf{r}_k^t \rangle_c), \tag{12}$$

We where $\langle \cdot \rangle_c$ denotes the average over samples x_i^c from class c, and $\mathbf{r}_k^\top(x_i^c)$ is the kth component of $f^\top(x_i^c) - y_i^c$. Combining (10), (11) and (12), we have the following convergence rates:

$$\langle \mathbf{r}_k^{t+1} \rangle = (1 - \eta\lambda_{\text{global}})\langle \mathbf{r}_k^t \rangle, \tag{13}$$

$$\langle \mathbf{r}_k^{t+1} \rangle_c - \langle \mathbf{r}_k^{t+1} \rangle = (1 - \eta\lambda_{\text{class}})(\langle \mathbf{r}_k^t \rangle_c - \langle \mathbf{r}_k^t \rangle), \tag{14}$$

$$\mathbf{r}_k^{t+1}(x_i^c) - \langle \mathbf{r}_k^{t+1} \rangle_c = (1 - \eta\lambda_{\text{single}})(\mathbf{r}_k^t(x_i^c) - \langle \mathbf{r}_k^t \rangle_c). \tag{15}$$

Overall, this means that the global mean $\langle \mathbf{r} \rangle$ of the residual converges first, then the class means, and finally the residual of each sample $\mathbf{r}(x_i^c)$. To simplify the notation, we define the following quantities:

$$\mathbf{R} = f(X) - \mathbf{Y} = [\mathbf{r}(x_1), \dots, \mathbf{r}(x_N)], \tag{16}$$

$$\mathbf{R}_{\text{class}} = \frac{1}{m}\mathbf{R}\mathbf{Y}^\top\mathbf{Y} = \underbrace{[\langle \mathbf{r} \rangle_1, \dots, \langle \mathbf{r} \rangle_C]}_{:=\mathbf{R}_1} \otimes \mathbf{1}_m^\top, \tag{17}$$

$$\mathbf{R}_{\text{global}} = \frac{1}{N}\mathbf{R}\mathbf{1}_N\mathbf{1}_N^\top = \langle \mathbf{r} \rangle \otimes \mathbf{1}_N^\top, \tag{18}$$

where $\mathbf{R} \in \mathbb{R}^{C\times N}$ is the matrix of residuals, $\mathbf{R}_{\text{class}} \in \mathbb{R}^{C\times N}$ are the residuals averaged over each class and stacked m times, and $\mathbf{R}_{\text{global}} \in \mathbb{R}^{C\times N}$ are the residuals averaged over the whole training set stacked N times. According to the previous discussion, $\mathbf{R}_{\text{global}}$ converges to zero at the fastest rate, while $\mathbf{R}$ converges at the slowest rate. The last phase, which we call the *end of training*, is when $\mathbf{R}_{\text{class}}$ and $\mathbf{R}_{\text{global}}$ have nearly vanished and can be treated as zero for the remaining training time. We will use this notion in several remarks, as well as in the proof of Theorem 5.2.

4.2 Gradient Flow Dynamics with Block-Structured NTK

We derive the dynamics of $\mathbf{H}, \mathbf{W}, \mathbf{b}$ under Assumption 3.2 in Theorem 4.1. One can see that the block-structured kernel greatly simplifies the complicated dynamics of DNNs and highlights the role of each of the residual components identified in Section 4.1. We consider gradient flow, which is close to gradient descent for sufficiently small step size [16], to reduce the complications caused by higher order terms. The proof is given in Appendix B.1.

Theorem 4.1. *Suppose Assumption 3.2 holds. Then the gradient flow dynamics of a DNN can be written as*

$$\begin{cases} \dot{\mathbf{H}} = & -\mathbf{W}^\top[(\kappa_d - \kappa_c)\mathbf{R} + (\kappa_c - \kappa_n)m\mathbf{R}_{\text{class}} + \kappa_n N\mathbf{R}_{\text{global}}] \\ \dot{\mathbf{W}} = & -\mathbf{R}\mathbf{H}^\top \\ \dot{\mathbf{b}} = & -\mathbf{R}_{\text{global}}\mathbf{1}_N. \end{cases} \tag{19}$$

We note that at the end of training, where $\mathbf{R}_{\text{class}}$ and $\mathbf{R}_{\text{global}}$ are zero, the system (19) reduces to

$$\dot{\mathbf{H}} = -(\kappa_d - \kappa_c)\nabla_{\mathbf{H}}\tilde{\mathcal{L}}, \qquad \dot{\mathbf{W}} = -\nabla_{\mathbf{W}}\tilde{\mathcal{L}}, \qquad \tilde{\mathcal{L}}(\mathbf{W}, \mathbf{H}) := \frac{1}{2}\|\mathbf{W}\mathbf{H} + \mathbf{b}\mathbf{1}_N^\top - \mathbf{Y}\|_F^2, \tag{20}$$

and $\dot{\mathbf{b}} = 0$. This system differs from the unconstrained features dynamics only by a factor of $\kappa_d - \kappa_c$ before $\mathbf{H}$. Moreover, such a form of the loss function also appears in the literature of implicit regularization [4, 6, 11], where the authors show that $\mathbf{W}\mathbf{H}$ converges to a low rank matrix.

5 NTK Alignment Drives Neural Collapse

The main goal of this work is to demonstrate how NC results from the NTK block structure. To this end, in Section 5.1 we further analyze the dynamics presented in Theorem 4.1, in Section 5.2 we derive the invariant of this training dynamics, and in Section 5.3 we finally derive NC.

5.1 Features Decomposition

We first decompose the features dynamics presented in Theorem 4.1 into two parts: $\mathbf{H}_1$, which lies in the subspace of the labels $\mathbf{Y}$, and $\mathbf{H}_2$, which is orthogonal to the labels and eventually vanishes. To achieve this, note that the SVD of $\mathbf{Y}$ has the following form:

$$\mathbf{P}^\top\mathbf{Y}\mathbf{Q} = \big[\sqrt{m}\mathbb{I}_C, \mathbb{O}\big], \tag{21}$$

where $\mathbb{O} \in \mathbb{R}^{C\times(N-C)}$ is a matrix of zeros, and $\mathbf{P} \in \mathbb{R}^{C\times C}$ and $\mathbf{Q} \in \mathbb{R}^{N\times N}$ are orthogonal matrices. Moreover, we can choose $\mathbf{P}$ and $\mathbf{Q}$ such that $\mathbf{P} = \mathbb{I}_C$ and

$$\mathbf{Q} = \big[\mathbf{Q}_1, \mathbf{Q}_2\big], \quad \mathbf{Q}_1 = \frac{1}{\sqrt{m}}\mathbb{I}_C \otimes \mathbf{1}_m \in \mathbb{R}^{N\times C}, \quad \mathbf{Q}_2 = \mathbb{I}_C \otimes \tilde{\mathbf{Q}}_2 \in \mathbb{R}^{N\times(N-C)}, \tag{22}$$

where $\otimes$ is the Kronecker product. Note that by orthogonality, $\tilde{\mathbf{Q}}_2 \in \mathbb{R}^{m\times(m-1)}$ has full rank and $\mathbf{1}_m^\top\tilde{\mathbf{Q}}_2 = \mathbb{O}$. We can now decompose $\mathbf{H}\mathbf{Q}$ into two components as follows:

$$\mathbf{H}\mathbf{Q} = \sqrt{m}[\mathbf{H}_1, \mathbf{H}_2], \quad \mathbf{H}_1 = \frac{1}{\sqrt{m}}\mathbf{H}\mathbf{Q}_1, \quad \mathbf{H}_2 = \frac{1}{\sqrt{m}}\mathbf{H}\mathbf{Q}_2. \tag{23}$$

The following equations reveal the meaning of these two components:

$$\mathbf{H}_1 = \big[\langle h\rangle_1, \ldots, \langle h\rangle_C\big], \quad \mathbf{H}_2 = \frac{1}{\sqrt{m}}\big[\mathbf{H}^{(1)}\tilde{\mathbf{Q}}_2, \ldots, \mathbf{H}^{(C)}\tilde{\mathbf{Q}}_2\big], \tag{24}$$

where $\langle h\rangle_c \in \mathbb{R}^n$ is the mean of h over inputs x_i^c from class $c \in [C]$, and $\mathbf{H}^{(c)} \in \mathbb{R}^{n\times m}$ is the submatrix of $\mathbf{H}$ corresponding to samples of class c, i.e., $\mathbf{H} = \big[\mathbf{H}^{(1)}, \ldots, \mathbf{H}^{(C)}\big]$. We see that $\mathbf{H}_1$

is simply the matrix of the last-layer features' class means, which is prominent in the NC literature. We also see that the columns of $\mathbf{H}^{(c)}\tilde{\mathbf{Q}}_2$ are $m-1$ different linear combinations of m vectors $h(x_i^c)$, $i \in [m]$. Moreover, the coefficients of each of these linear combinations sum to zero by the choice of $\tilde{\mathbf{Q}}_2$. Therefore, $\mathbf{H}_2$ must reduce to zero in case of variability collapse (NC1), when all the feature vectors within the same class become equal. We prove that $\mathbf{H}_2$ indeed vanishes in DNNs with block-structured NTK as part of our main result (Theorem 5.2).

5.2 Invariant

We now use the former decomposition of the last-layer features to further simplify the dynamics and deduce a training invariant in Theorem 5.1. The proof is given in Appendix B.2.

Theorem 5.1. *Suppose Assumption 3.2 holds. Define $\mathbf{H}_1$ and $\mathbf{H}_2$ as in* (23)*. Then the class-means of the residuals (defined in* (17)*) are given by $\mathbf{R}_1 = \mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top - \mathbb{I}_C$, and the training dynamics of the DNN can be written as*

$$\begin{cases} \dot{\mathbf{H}}_1 &= -\mathbf{W}^\top \mathbf{R}_1(\mu_{\text{class}}\mathbb{I}_C + \kappa_n m \mathbf{1}_C\mathbf{1}_C^\top) \\ \dot{\mathbf{H}}_2 &= -\mu_{\text{single}}\mathbf{W}^\top\mathbf{W}\mathbf{H}_2 \\ \dot{\mathbf{W}} &= -m(\mathbf{R}_1\mathbf{H}_1^\top + \mathbf{W}\mathbf{H}_2\mathbf{H}_2^\top) \\ \dot{\mathbf{b}} &= -m\mathbf{R}_1\mathbf{1}_C, \end{cases} \tag{25}$$

where $\mu_{\text{single}} := \kappa_d - \kappa_c$ and $\mu_{\text{class}} := \mu_{\text{single}} + m(\kappa_c - \kappa_n)$ are the two smallest eigenvalues of the kernel $\Theta^h_{k,k}(X)$ for any $k \in [n]$. Moreover, the quantity

$$\mathbf{E} := \frac{1}{m}\mathbf{W}^\top\mathbf{W} - \frac{1}{\mu_{\text{class}}}\mathbf{H}_1(\mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top)\mathbf{H}_1^\top - \frac{1}{\mu_{\text{single}}}\mathbf{H}_2\mathbf{H}_2^\top \tag{26}$$

is invariant in time. Here $\alpha := \frac{\kappa_n m}{\mu_{\text{class}} + C\kappa_n m}$.

We note that the invariant $\mathbf{E}$ derived here resembles the conservation laws of *hyperbolic* dynamics that take the form $\mathbf{E}_{\text{hyp}} := a^2 - b^2 = const$ for time-dependent quantities a and b. Such dynamics arise when gradient flow is applied to a loss function of the form $\mathcal{L}(a,b) := (ab-q)^2$ for some q. Since the solutions of such minimization problems, given by $ab = q$, exhibit symmetry under scaling $a \to \gamma a, b \to b/\gamma$, the value of the invariant $\mathbf{E}_{\text{hyp}}$ uniquely specifies the hyperbola followed by the solution. In machine learning theory, hyperbolic dynamics arise as the gradient flow dynamics of linear DNNs [42], or in matrix factorization problems [3, 15]. Moreover, the end of training dynamics defined in (20) has a hyperbolic invariant given by

$$\mathbf{E}_{\text{eot}} := \mathbf{W}^\top\mathbf{W} - \frac{1}{\mu_{\text{single}}}\mathbf{H}\mathbf{H}^\top. \tag{27}$$

Therefore, the final phase of training exhibits a typical behavior for the hyperbolic dynamics, which is also characteristic for the unconstrained features models [21, 38]. Namely, "scaling" $\mathbf{W}$ and $\mathbf{H}$ by an invertible matrix does not affect the loss value but changes the dynamic's invariant. On the other hand, minimizing the invariant $\mathbf{E}_{\text{eot}}$ has the same effect as joint regularization of $\mathbf{W}$ and $\mathbf{H}$ [48].

However, we also note that our invariant $\mathbf{E}$ provides a new, more comprehensive look at the DNNs' dynamics. While unconstrained features models effectively make assumptions on the end-of-training invariant $\mathbf{E}_{\text{eot}}$ to derive NC [21, 38, 48], our dynamics control the value of $\mathbf{E}_{\text{eot}}$ through the more general invariant $\mathbf{E}$. This way we connect the properties of end-of-training hyperbolic dynamics with the previous stages of training.

5.3 Neural Collapse

We are finally ready to state and prove our main result in Theorem 5.2 about the emergence of NC in DNNs with NTK alignment. We include the proof in Appendix B.3.

Theorem 5.2. *Assume that the NTK has a block structure as defined in Assumption 3.2. Then the DNN's training dynamics are given by the system of equations in* (25)*. Assume further that the last-layer features are centralized, i.e $\langle h \rangle = 0$, and the dynamics invariant* (26) *is zero, i.e., $\mathbf{E} = \mathbb{O}$. Then the DNN's dynamics exhibit neural collapse as defined in (NC1)-(NC4).*

Below we provide several important remarks and discuss the implications of this result:

(1) Zero invariant assumption: We assume that the invariant (26) is zero in Theorem 5.2 for simplicity and consistency with the literature. Indeed, similar assumptions arise in matrix decomposition papers, where zero invariant guarantees "balance" of the problem [3, 15]. However, our proofs in fact only require a weaker assumption that the invariant terms containing features $\mathbf{H}$ are aligned with the weights $\mathbf{W}^\top\mathbf{W}$, i.e.

$$\mathbf{W}^\top\mathbf{W} \propto \frac{1}{\mu_{\text{class}}}\mathbf{H}_1\mathbf{H}_1^\top - \frac{1}{\mu_{\text{single}}}\mathbf{H}_2\mathbf{H}_2^\top, \tag{28}$$

where we have taken into account our assumption on the zero global mean $\langle h\rangle = 0$.

(2) Necessity of the invariant assumption: The relaxed assumption on the invariant (28) is necessary for the emergence of NC in DNNs with block-structured NTK. Indeed, NC1 implies $\mathbf{H}_2 = \mathbb{O}$, and NC3 implies $\mathbf{H}_1\mathbf{H}_1^\top \propto \mathbf{W}^\top\mathbf{W}$. Therefore, DNNs that do not satisfy this assumption do not display NC. Our numerical experiments described in Section 6 strongly support this insight (see Figure 2, panes a-e). Thus, we believe that the invariant derived in this work characterizes the difference between models that do and do not exhibit NC.

(3) Zero global mean assumption: We note that the zero global mean assumption $\langle h\rangle = 0$ in Theorem 5.2 ensures that the biases are equal to $\mathbf{b} = \frac{1}{C}\mathbf{1}_C$ at the end of training. This assumption is common in the NC literature [21, 38] and is well-supported by our numerical experiments (see figures in Appendix C, pane i). Indeed, modern DNNs typically include certain normalization (e.g. through batch normalization layers) to improve numerical stability, and closeness of the global mean to zero is a by-product of such normalization.

(4) General biases case: Discarding the zero global mean assumption allows the biases $\mathbf{b}$ to take an arbitrary form. In this general case, the following holds for the matrix of weights:

$$(\mathbf{W}\mathbf{W}^\top)^2 = \frac{m}{\mu_{\text{class}}}\Big(\mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top + (1-\alpha C)(C\mathbf{b}\mathbf{b}^\top - \mathbf{b}\mathbf{1}_C^\top - \mathbf{1}_C\mathbf{b}^\top)\Big). \tag{29}$$

For optimal biases $\mathbf{b} = \frac{1}{C}\mathbf{1}_C$, this reduces to the ETF structure that emerges in NC. Moreover, if biases are all equal, i.e. $\mathbf{b} = \beta\mathbf{1}_C$ for some $\beta \in \mathbb{R}$, the centralized class means still form an ETF (i.e., NC2 holds), and the weights exhibit a certain symmetric structure given by

$$\mathbf{W}\mathbf{W}^\top \propto \Big(\mathbb{I}_C - \gamma\mathbf{1}_C\mathbf{1}_C^\top\Big), \quad \mathbf{M}^\top\mathbf{M} \propto \Big(\mathbb{I}_C - \frac{1}{C}\,\mathbf{1}_C\mathbf{1}_C^\top\Big), \tag{30}$$

where $\gamma := \frac{1}{C}(1 - |1-\beta C|\sqrt{1-\alpha C}) < \frac{1}{C}$. The proof and a discussion of this result are given in Appendix B.4. In general, the angles of these two frames are different, and thus NC3 does not hold. This insight leads us to believe that normalization is an important factor in the emergence of NC.

(5) Partial NC: Our proofs and the discussion suggest that all the four phenomena that form NC do not have to always coincide. In particular, our proof of NC1 only requires the block-structured NTK and the invariant to be P.S.D, which is much weaker than the total set of assumptions in Theorem 5.2. Therefore, variability collapse can occur in models that do not exhibit the ETF structure of the class-means or the duality of the weights and the class means. Moreover, as shown above, NC2 can occur when NC3 does not, i.e., the ETF structure of the class means does not imply duality.

6 Experiments

We conducted large-scale numerical experiments to support our theory. While we only showcase our results on a single dataset-architecture pair in the main text (see Figure 2) and refer the rest to the appendix, the following discussion covers all our experiments.

Datasets and models. Following the seminal NC paper [39], we use three canonical DNN architectures: VGG [46], ResNet [24] and DenseNet [26]. Our datasets are MNIST [35], FashionMNIST [51] and CIFAR10 [34]. We choose VGG11 for MNIST and FashionMNIST, and VGG16 for CIFAR10. We add batch normalization after every layer in the VGG architecture, set dropout to zero and choose the dimensions of the two fully-connected layers on the top of the network as 512 and 256. We use ResNet20 architecture described in the original ResNet paper [24], and DenseNet40 with bottleneck layers, growth $k = 12$, and zero dropout for all the datasets.

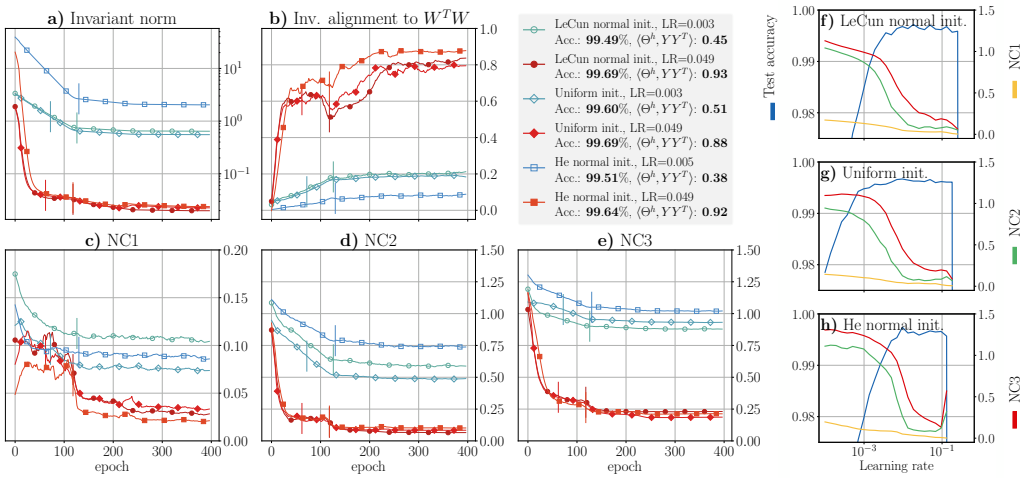


Figure 2: ResNet20 trained on MNIST with three initialization settings and varying learning rates (see Section 6 for details). We chose a model that exhibits NC (red lines, filled markers) and a model that does not exhibit NC (blue lines, empty markers) for each initialization. The vertical lines indicate the epoch when the training accuracy reaches 99.9% (over the last 10 batches). **a)** Frobenious norm of the invariant $\|\mathbf{E}\|_F$. **b)** Alignment of the invariant terms as defined in (28). **c)** NC1: standard deviation of $h(x_i^c)$ averaged over classes. **d)** NC2: $\|\mathbf{M}^\top\mathbf{M}/\|\mathbf{M}^\top\mathbf{M}\|_F - \Phi\|_F$, where Φ is an ETF. **e)** NC3: $\|\mathbf{W}^\top/\|\mathbf{W}\|_F - \mathbf{M}/\|\mathbf{M}\|_F\|_F$. The legend displays the test accuracy achieved by each model and the last-layer features kernel alignment given by $\langle\Theta^h/\|\Theta^h\|_F, \mathbf{Y}^\top\mathbf{Y}/\|\mathbf{Y}^\top\mathbf{Y}\|_F\rangle_F$. The curves in panes a-e are smoothed by Savitzky–Golay filter with polynomial degree 1 over window of size 10. Panes **f**, **g** and **h** show the NC metrics and the test accuracy as functions of the learning rate.

Optimization and initialization. We use SGD with Nesterov momentum 0.9 and weight decay 5×10^{-4}. Every model is trained for 400 epochs with batches of size 120. To be consistent with the theory, we balance the batches exactly. We train every model with a set of initial learning rates spaced logarithmically in the range $\eta \in [10^{-4}, 10^{0.25}]$. The learning rate is divided by 10 every 120 epochs. On top of the varying learning rates, we try three different initialization settings for every model: **(a)** LeCun normal initialization (default in Flax), **(b)** uniform initialization on $[-\sqrt{k}, \sqrt{k}]$, where $k = 1/n_{\ell-1}$ for a linear layer, and $k = 1/(Kn_{\ell-1})$ for a convolutional layer, where K is the convolutional kernel size (default in PyTorch), **(c)** He normal initialization in `fan_out` mode.

Results. Our experiments confirm the validity of our assumptions and the emergence of NC as their result. Specifically, we make the following observations:

- While most of the DNNs that achieve high test performance exhibit NC, we are able to identify DNNs with comparable performance that do not exhibit NC (see Figure 2, panes f-h). We note that such models still achieve near-zero error on the training set in our setup.
- Comparing DNNs that do and do not exhibit NC, we find that our assumption on the invariant (see Theorem 5.2 and (28)) holds only for the models with NC (see Figure 2, panes a-e). This confirms our reasoning about the necessity of the invariant assumption for NC emergence.
- The kernels Θ and Θ^h are strongly aligned with the labels $\mathbf{Y}^\top\mathbf{Y}$ in the models with the best performance, which is in agreement with the NTK alignment literature and justifies our assumption on the NTK block structure.

We include the full range of experiments along with the implementation details and the discussion of required computational resources in Appendix C. Specifically, we present a figure analogous to Figure 2 for every considered dataset-architecture pair. Additionally, we report the norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h\rangle\langle h\rangle^\top$, as well as the alignment of both the NTK Θ and the last-layer features kernel Θ^h in the end of training, to further justify our assumptions.

7 Conclusions and Broad Impact

This work establishes the connection between NTK alignment and NC, and thus provides a mechanistic explanation for the emergence of NC within realistic DNNs' training dynamics. It also contributes to the underexplored line of research connecting NC and local elasticity of DNNs' training dynamics.

The primary implication of this research is that it exposes the potential to study NC through the lens of NTK alignment. Indeed, previous works on NC focus on the top-down approach (layer-peeled models) [18, 21, 38, 48], and fundamentally cannot explain how NC develops through earlier layers of a DNN and what are the effects of depth. On the other hand, NTK alignment literature focuses on the alignment of individual layers [7], and recent theoretical results even quantify the role of each hidden layer in the final alignment [37]. Therefore, we believe that the connection between NTK alignment and NC established in this work provides a conceptually new method to study NC.

Moreover, this work introduces a novel approach to facilitate theoretical analysis of DNNs' training dynamics. While most theoretical works consider the NTK in the infinite-width limit to simplify the dynamics [1, 20, 28, 49], our analysis shows that making reasonable assumptions on the empirical NTK can also lead to tractable dynamics equations and new theoretical results. Thus, we believe that the analysis of DNNs' training dynamics based on the properties of the empirical NTK is a promising approach also beyond NC research.

8 Limitations and Future Work

The main limitation of this work is the simplifying Assumption 3.2 on the kernel structure. While the NTK of well-trained DNNs indeed has an approximate block structure (as we discuss in detail in Section 3), the NTK values also tend to display high variance in real DNNs [22, 44]. Thus, we believe that adding stochasticity to the dynamics considered in this paper is a promising direction for the future work. Moreover, the empirical NTK exhibits so-called specialization, i.e., the kernel matrix corresponding to a certain output neurons aligns more with the labels of the corresponding class [45]. In block-structured kernels, specialization implies different values in blocks corresponding to different classes. Thus, generalizing our theory to block-structured kernels with specialization is another promising short-term research goal. In addition, our theory relies on the assumption that the dataset (or the training batch) is balanced, i.e., all the classes have the same number of samples. Accounting for the effects of non-balanced datasets within the dynamics of DNNs with block-structured NTK is another possible future work direction.

More generally, we believe that empirical observations are essential to demistify the DNNs' training dynamics, and there are still many unknown and interesting connections between seemingly unrelated empirical phenomena. Establishing new theoretical connections between such phenomena is an important objective, since it provides a more coherent picture of the deep learning theory as a whole.

Acknowledgments and Disclosure of Funding

R. Giryes and G. Kutyniok acknowledge support from the LMU-TAU - International Key Cooperation Tel Aviv University 2023. R. Giryes is also grateful for partial support by ERC-StG SPADE grant no. 757497. G. Kutyniok is grateful for partial support by the Konrad Zuse School of Excellence in Reliable AI (DAAD), the Munich Center for Machine Learning (BMBF) as well as the German Research Foundation under Grants DFG-SPP-2298, KU 1446/31-1 and KU 1446/32-1 and under Grant DFG-SFB/TR 109, Project C09 and the Federal Ministry of Education and Research under Grant MaGriDo.

References

[1] Ben Adlam and Jeffrey Pennington. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 74–84. PMLR, 2020.

[2] Laurence Aitchison. Why bigger is not always better: on finite and infinite neural networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 156–164. PMLR, 2020.

[3] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 244–253. PMLR, 2018.

[4] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7411–7422, 2019.

[5] Alexander Atanasov, Blake Bordelon, and Cengiz Pehlevan. Neural networks as kernel learners: The silent alignment effect. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[6] Bubacarr Bah, Holger Rauhut, Ulrich Terstiege, and Michael Westdickenberg. Learning deep linear neural networks: Riemannian gradient flows and convergence to global minimizers. *Information and Inference: A Journal of the IMA*, 11(1):307–353, 02 2021.

[7] Aristide Baratin, Thomas George, César Laurent, R. Devon Hjelm, Guillaume Lajoie, Pascal Vincent, and Simon Lacoste-Julien. Implicit regularization via neural feature alignment. In *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, volume 130 of *Proceedings of Machine Learning Research*, pages 2269–2277. PMLR, 2021.

[8] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[9] Shuxiao Chen, Hangfeng He, and Weijie J. Su. Label-aware neural tangent kernel: Toward better generalization and local elasticity. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[10] Lénaïc Chizat, Edouard Oyallon, and Francis R. Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2933–2943, 2019.

[11] Hung-Hsu Chou, Carsten Gieshoff, Johannes Maly, and Holger Rauhut. Gradient descent for deep matrix factorization: Dynamics and implicit bias towards low rank. *arXiv preprint: 2011.13772*, 2020.

[12] Moustapha Cissé, Piotr Bojanowski, Edouard Grave, Yann N. Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th*

International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017, volume 70 of *Proceedings of Machine Learning Research*, pages 854–863. PMLR, 2017.

[13] Nello Cristianini, John Shawe-Taylor, André Elisseeff, and Jaz S. Kandola. On kernel-target alignment. In *Advances in Neural Information Processing Systems 14: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada*, pages 367–373. MIT Press, 2001.

[14] Ahmet Demirkaya, Jiasi Chen, and Samet Oymak. Exploring the role of loss functions in multiclass classification. In *54th Annual Conference on Information Sciences and Systems, CISS 2020, Princeton, NJ, USA, March 18-20, 2020*, pages 1–5. IEEE, 2020.

[15] Simon S. Du, Wei Hu, and Jason D. Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 382–393, 2018.

[16] Omer Elkabetz and Nadav Cohen. Continuous vs. discrete optimization of deep neural networks. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 4947–4960, 2021.

[17] Tolga Ergen and Mert Pilanci. Revealing the structure of deep neural networks via convex duality. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 3004–3014. PMLR, 2021.

[18] C Fang, H He, Q Long, and WJ Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences of the United States of America*, 118(43), 2021.

[19] Stanislav Fort, Gintare Karolina Dziugaite, Mansheej Paul, Sepideh Kharaghani, Daniel M. Roy, and Surya Ganguli. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[20] Mario Geiger, Arthur Jacot, Stefano Spigler, Franck Gabriel, Levent Sagun, Stéphane d'Ascoli, Giulio Biroli, Clément Hongler, and Matthieu Wyart. Scaling description of generalization with number of parameters in deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(2):023401, 2020.

[21] X. Y. Han, Vardan Papyan, and David L. Donoho. Neural collapse under MSE loss: Proximity to and dynamics on the central path. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[22] Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[23] Hangfeng He and Weijie J. Su. The local elasticity of neural networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[25] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2020.

[26] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society, 2017.

[27] Jiaoyang Huang and Horng-Tzer Yau. Dynamics of deep neural networks and neural tangent hierarchy. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4542–4551. PMLR, 2020.

[28] Kaixuan Huang, Yuqing Wang, Molei Tao, and Tuo Zhao. Why do deep residual networks generalize better than deep feedforward networks? - A neural tangent kernel perspective. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[29] Like Hui and Mikhail Belkin. Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[30] Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8580–8589, 2018.

[31] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[32] Dmitry Kopitkov and Vadim Indelman. Neural spectrum alignment: Empirical study. In *Artificial Neural Networks and Machine Learning - ICANN 2020 - 29th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 15-18, 2020, Proceedings, Part II*, volume 12397 of *Lecture Notes in Computer Science*, pages 168–179. Springer, 2020.

[33] Vignesh Kothapalli, Ebrahim Rasromani, and Vasudev Awatramani. Neural collapse: A review on modelling principles and generalization. *arXiv preprint arXiv:2206.04041*, 2022.

[34] Alex Krizhevsky et al. Learning multiple layers of features from tiny images, 2009.

[35] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[36] Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. *Advances in Neural Information Processing Systems*, 33:15156–15172, 2020.

[37] Yizhang Lou, Chris E Mingard, and Soufiane Hayou. Feature learning and signal propagation in deep neural networks. In *International Conference on Machine Learning*, pages 14248–14282. PMLR, 2022.

[38] Dustin G Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *arXiv preprint arXiv:2011.11619*, 2020.

[39] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

[40] Federico Pernici, Matteo Bruni, Claudio Baecchi, and Alberto Del Bimbo. Fix your features: Stationary and maximally discriminative embeddings using regular polytope (fixed classifier) networks. *arXiv preprint arXiv:1902.10441*, 2019.

[41] Tomaso A. Poggio and Qianli Liao. Explicit regularization and implicit bias in deep network classifiers trained with the square loss. *arXiv preprint arXiv:2101.00072*, 2021.

[42] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[43] Mariia Seleznova and Gitta Kutyniok. Analyzing finite neural networks: Can we trust neural tangent kernel theory? In *Mathematical and Scientific Machine Learning, 16-19 August 2021, Virtual Conference / Lausanne, Switzerland*, volume 145 of *Proceedings of Machine Learning Research*, pages 868–895. PMLR, 2021.

[44] Mariia Seleznova and Gitta Kutyniok. Neural tangent kernel beyond the infinite-width limit: Effects of depth and initialization. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 19522–19560. PMLR, 2022.

[45] Haozhe Shan and Blake Bordelon. A theory of neural tangent kernel alignment and its influence on training. *arXiv preprint arXiv:2105.14301*, 2021.

[46] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[47] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6398–6407, 2020.

[48] Tom Tirer and Joan Bruna. Extended unconstrained features model for exploring deep neural collapse. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 21478–21505, 2022.

[49] Tom Tirer, Joan Bruna, and Raja Giryes. Kernel-based smoothness analysis of residual networks. In *Mathematical and Scientific Machine Learning*, volume 145, pages 921–954, 2021.

[50] Tom Tirer, Haoxiang Huang, and Jonathan Niles-Weed. Perturbation analysis of neural collapse. *arXiv preprint arXiv:2210.16658*, 2022.

[51] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[52] Greg Yang. Tensor programs II: neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*, 2020.

[53] Jiayao Zhang, Hua Wang, and Weijie J. Su. Imitating deep learning dynamics via locally elastic stochastic differential equations. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 6392–6403, 2021.

A Related works

NC with MSE loss. NC was first introduced for DNNs with cross-entropy (CE) loss, which is commonly used in classification problems [39]. Since then, numerous papers discussed NC with MSE loss, which provides more opportunities for theoretical analysis, especially after the MSE loss was shown to perform on par with CE loss for classification tasks [14, 29].

Most previous works on MSE-NC adopt the so-called unconstrained features model [21, 38, 48]. In this model, the last-layer features $\mathbf{H}$ are free variables that are directly optimized during training, i.e., the features do not depend on the input data or the DNN's trainable parameters. Fang *et al.* [18] also introduced a generalization of this approach called N-layer-peeled model, where features of the N-th-to-last layer are free variables, and studied the 1-layer-peeled model (equivalent to the unconstrained features model) with CE loss as a special case.

One line of research on MSE-NC in unconstrained/layer-peeled models aims to derive global minimizers of optimization problems associated with DNNs [17, 18, 48]. In particular, Tirer *et al.* [48] showed that global minimizers of the MSE loss with regularization of both $\mathbf{H}$ and $\mathbf{W}$ exhibit NC. Moreover, Ergen & Pilanci [17] showed that NC emerges in global minimizers of optimization problems with general convex loss in the context of the 2-layer-peeled model. In comparison to our work, these contributions do not consider the training dynamics of DNNs, i.e., they do not discuss whether and how the model converges to the optimal solution.

Another line of research on MSE-NC explicitly considers the dynamics of the unconstrained features models [21, 38]. In particular, Han *et al.* [21] considered the gradient flow of the unconstrained renormalized features along the "central path", where the classifier is assumed to take the form of the optimal least squares (OLS) solution for given features $\mathbf{H}$. Under this assumption, they derive a closed-form dynamics that implies NC. While they empirically show that DNNs are close to the central path in certain scenarios, they do not provide a theoretical justification for this assumption. The dynamics considered in their work is also distinct from the standard gradient flow dynamics of DNNs considered in our work. On the other hand, an earlier work by Mixon *et al.* [38] considered the gradient flow dynamics of the unconstrained features model, which is equivalent (up to rescaling) to the end-of-training dynamics (20) that we discuss in Sections 4.2 and 5.2. Their work relies on the linearization of these dynamics to derive a certain subspace, which appears to be an invariant subspace of the non-linearized unconstrained features model dynamics. Then they show that minimizers of the loss from this subspace exhibit NC. We note that, in terms of our paper, assuming that the unconstrained features model dynamics follow a certain invariant subspace means making assumptions on the end-of-training invariant (27). In comparison to these works, we make a step towards realistic DNNs dynamics by considering the standard gradient flow of DNNs simplified by Assumption 3.2 on the NTK structure, which is supported by the extensive research on NTK alignment [7, 9, 44, 45]. In our setting, the NTK captures the dependence of the features on the training data, which is missing in the unconstrained features model. Moreover, while other works focus only on the dynamics that converge to NC, we show that DNNs with MSE loss may not exhibit NC in certain settings, and the invariant of the dynamics (26) characterizes the difference between models that do and do not converge to NC.

Notably, works by Poggio & Liao [41] adopt a model different from the unconstrained features model to analyze gradient flow of DNNs. They consider the dynamics of homogeneous DNNs, in particular ReLU networks without biases, with normalization of the weights matrices and weights regularization. The goal of weights normalization in their model is to imitate the effects of batch normalization in DNNs training. In this model, certain fixed points of the gradient flow exhibit NC. While the approach taken in their work captures the dependence of the features on the data and the DNN's parameters, it fundamentally relies on the homogeneity of the DNN's output function. However, most DNNs that exhibit NC in practice are not homogeneous due to biases and skip-connections.

NC and local elasticity. A recent extensive survey of NC literature [33] discussed local elasticity as a possible mechanism behind the emergence of NC, which has not been sufficiently explored up until now. One of the few works in this research direction is by Zhang *et al.* [53], who analyzed the so-called locally-elastic stochastic differential equations (SDEs) and showed the emergence of NC in their solutions. They model local elasticity of the dynamics through an effect matrix, which has only two distinct values: a larger intra-class value and a smaller inter-class value. These values characterize how much influence samples from one class have on samples from other classes in the SDEs. While the aim of their work is to imitate DNNs' training dynamics through SDEs, the authors

do not provide any explicit connection between their dynamics and real gradient flow dynamics of DNNs. On the other hand, we derive our dynamics directly from the gradient flow equations and connect local elasticity to the NTK, which is a well-studied object in the deep learning theory.

Another work by Tirer *et al.* [50] provided a perturbation analysis of NC to study "inexact collapse". They considered a minimization problem with MSE loss, regularization of $\mathbf{H}$ and $\mathbf{W}$, and additional regularization of the distance between $\mathbf{H}$ and a given matrix of initial features. In the "near-collapse" setting, i.e., when the initial features are already close to collapse, they showed that the optimal features can be obtained from the initial features by a certain linear transformation with a block structure, where the intra-class effects are stronger than the inter-class ones. While this transformation matrix resembles the block-structured effect matrices in locally-elastic training dynamics, it does not originate from the gradient flow dynamics of DNNs and is not related to the NTK.

B Proofs

B.1 Proof of Theorem 4.1

Proof of Theorem 4.1. We will first derive the dynamics of $h_s(x_i^c)$, which is the s-th component of the last-layer features vector on sample $x_i^c \in X$ from class $c \in [C]$. Let $\mathbf{w} \in \mathbb{R}^P$ be the trainable parameters of the network stretched into a single vector. Then its gradient flow dynamics is given by

$$\dot{\mathbf{w}} = -\nabla_{\mathbf{w}} \mathcal{L}(f) = -\sum_{k=1}^{C} \sum_{i'=1}^{N} (f(X)_{ki'} - \mathbf{Y}_{ki'}) \nabla_{\mathbf{w}} f(X)_{ki'}, \tag{31}$$

where $\nabla_{\mathbf{w}} f(X)_{ki'} \in \mathbb{R}^P$ is the component of the DNN's Jacobian corresponding to output neuron k and the input sample $x_{i'}^{c'}$. Since entries of $f(X)$ can be written as

$$f(X)_{ki'} = \sum_{s'=1}^{n} \mathbf{W}_{ks'} \mathbf{H}_{s'i'} + \mathbf{b}_k = \sum_{s'=1}^{n} \mathbf{W}_{ks'} h_{s'}(x_{i'}^{c'}) + \mathbf{b}_k, \tag{32}$$

we obtain

$$\dot{\mathbf{w}} = -\sum_{k=1}^{C} \sum_{i'=1}^{N} \sum_{s'=1}^{n} (f(X)_{ki'} - \mathbf{Y}_{ki'}) \nabla_{\mathbf{w}} (\mathbf{W}_{ks'} h_{s'}(x_{i'}^{c'}) + \mathbf{b}_k). \tag{33}$$

By chain rule, we have $\dot{h}_s(x_i^c) = \langle \nabla_{\mathbf{w}} h_s(x_i^c), \dot{\mathbf{w}} \rangle$. Then, taking into account that

$$\langle \nabla_{\mathbf{w}} h_s(x_i^c), \nabla_{\mathbf{w}} (\mathbf{W}_{ks'} h_{s'}(x_{i'}^{c'}) + \mathbf{b}_k) \rangle = \mathbf{W}_{ks'} \langle \nabla_{\mathbf{w}} h_s(x_i^c), \nabla_{\mathbf{w}} h_{s'}(x_{i'}^{c'}) \rangle, \tag{34}$$

and that $\langle \nabla_{\mathbf{w}} h_s(x_i^c), \nabla_{\mathbf{w}} h_{s'}(x_{i'}^{c'}) \rangle = \Theta^h_{s,s'}(x_i^c, x_{i'}^{c'})$ by definition of Θ^h, we have

$$\dot{h}_s(x_i^c) = -\sum_{k=1}^{C} \sum_{i'=1}^{N} \sum_{s'=1}^{n} (f(X)_{ki'} - \mathbf{Y}_{ki'}) \mathbf{W}_{ks'} \Theta^h_{s,s'}(x_i^c, x_{i'}^{c'}). \tag{35}$$

Now by Assumption 3.2 we have $\Theta^h_{s,s'} = 0$ if $s \neq s'$. Therefore, the above expression simplifies to

$$\begin{aligned} \dot{h}_s(x_i^c) &= -\sum_{i'=1}^{N} \Theta^h_{s,s}(x_i^c, x_{i'}^{c'}) \sum_{k=1}^{C} (f(X)_{ki'} - \mathbf{Y}_{ki'}) \mathbf{W}_{ks} \\ &= -\sum_{i'=1}^{N} [\mathbf{W}^\top (\mathbf{W}\mathbf{H} + \mathbf{b}\mathbf{1}_N^\top - \mathbf{Y})]_{si'} \Theta^h_{s,s}(x_i^c, x_{i'}^{c'}). \end{aligned}$$

To express $\dot{\mathbf{H}} = \left[\dot{h}_s(x_i^c)\right]_{s,i} \in \mathbb{R}^{n \times N}$ in matrix form, it remains to express $\Theta^h_{s,s}(x_i^c, x_{i'}^{c'})$ as the (i', i)-th entry of some matrix. We will separate the sum into three cases: 1) $i = i'$, 2) $i \neq i'$ and $c = c'$, and 3) $c \neq c'$. According to Assumption 3.2, the first case corresponds to the multiple of identity $\kappa_d \mathbb{I}_N$. The second corresponds to the block matrix of size m with zeros on the diagonal,

which can be written as $\kappa_c(\mathbf{Y}^\top\mathbf{Y} - \mathbb{I}_N)$. The third matrix equals to $\kappa_n(\mathbf{1}_N\mathbf{1}_N^\top - \mathbf{Y}^\top\mathbf{Y})$. Therefore we can express the dynamics of $\mathbf{H}$ as follows:

$$\begin{aligned}\dot{\mathbf{H}} =& -[\mathbf{W}^\top(\mathbf{W}\mathbf{H} + \mathbf{b}\mathbf{1}_N^\top - \mathbf{Y})][\kappa_d\mathbb{I} + \kappa_c(\mathbf{Y}^\top\mathbf{Y} - \mathbb{I}) + \kappa_n(\mathbf{1}_N\mathbf{1}_N^\top - \mathbf{Y}^\top\mathbf{Y})] \\ =& -(\kappa_d - \kappa_c)\mathbf{W}^\top(\mathbf{W}\mathbf{H} + \mathbf{b}\mathbf{1}_N^\top - \mathbf{Y}) \\ & -(\kappa_c - \kappa_n)\mathbf{W}^\top(\mathbf{W}\mathbf{H}\mathbf{Y}^\top\mathbf{Y} + m\mathbf{b}\mathbf{1}_N^\top - m\mathbf{Y}) \\ & -\kappa_n\mathbf{W}^\top(\mathbf{W}\mathbf{H}\mathbf{1}_N\mathbf{1}_N^\top + N\mathbf{b}\mathbf{1}_N^\top - \frac{N}{C}\mathbf{1}_C\mathbf{1}_N^\top).\end{aligned}$$

Now we notice that $\mathbf{H}\mathbf{Y}^\top\mathbf{Y}/m$ is the matrix of stacked class means repeated m times each and $\mathbf{H}\mathbf{1}_N\mathbf{1}_N^\top/N$ is a matrix of the global mean repeated N times. Therefore, we have

$$\mathbf{W}\mathbf{H}\mathbf{Y}^\top\mathbf{Y} + m\mathbf{b}\mathbf{1}_N^\top - m\mathbf{Y} = m\mathbf{R}_{\text{class}},$$
$$\mathbf{W}\mathbf{H}\mathbf{1}_N\mathbf{1}_N^\top + N\mathbf{b}\mathbf{1}_N^\top - \frac{N}{C}\mathbf{1}_C\mathbf{1}_N^\top = N\mathbf{R}_{\text{global}}$$

according to the definitions of global and class-mean residuals in (18) and (17).

The expressions for the gradient flow dynamics of $\mathbf{W}$ and $\mathbf{b}$ follow directly from the derivatives of $f(X)$ w.r.t. $\mathbf{W}$ and $\mathbf{b}$. This completes the proof. □

B.2 Proof of Theorem 5.1

Proof of Theorem 5.1. Recall from (23) in Section 5.1 that we have the following decomposition

$$\mathbf{H}\mathbf{Q} = \sqrt{m}[\mathbf{H}_1, \mathbf{H}_2], \quad \mathbf{H}_1 = \frac{1}{\sqrt{m}}\mathbf{H}\mathbf{Q}_1, \quad \mathbf{H}_2 = \frac{1}{\sqrt{m}}\mathbf{H}\mathbf{Q}_2$$

with orthogonal $\mathbf{Q} = [\mathbf{Q}_1, \mathbf{Q}_2] \in \mathbb{R}^{N\times N}$. We now artificially add $\mathbf{Q}\mathbf{Q}^\top(= \mathbb{I}_N)$ to the dynamics (19) in Theorem 4.1 and obtain

$$\begin{cases}\dot{\mathbf{H}}\mathbf{Q} = & -(\kappa_d - \kappa_c)\mathbf{W}^\top(\mathbf{W}\mathbf{H}\mathbf{Q} + \mathbf{b}\mathbf{1}_N^\top\mathbf{Q} - Y\mathbf{Q}) \\ & -(\kappa_c - \kappa_n)m\mathbf{W}^\top(\frac{1}{m}\mathbf{W}\mathbf{H}\mathbf{Q}\mathbf{Q}^\top\mathbf{Y}^\top\mathbf{Y}\mathbf{Q} + \mathbf{b}\mathbf{1}_N^\top\mathbf{Q} - Y\mathbf{Q}) \\ & -\kappa_n N\mathbf{W}^\top(\frac{1}{N}\mathbf{W}\mathbf{H}\mathbf{Q}\mathbf{Q}^\top\mathbf{1}_N\mathbf{1}_N^\top\mathbf{Q} + \mathbf{b}\mathbf{1}_N^\top\mathbf{Q} - \frac{1}{C}\mathbf{1}_C\mathbf{1}_N^\top\mathbf{Q}) \\ \dot{\mathbf{W}} = & -(\mathbf{W}\mathbf{H}\mathbf{Q} + \mathbf{b}\mathbf{1}_N^\top\mathbf{Q} - Y\mathbf{Q})\mathbf{Q}^\top\mathbf{H}^\top \\ \dot{\mathbf{b}} = & -(\mathbf{W}\mathbf{H}\mathbf{Q} + \mathbf{b}\mathbf{1}_N^\top\mathbf{Q} - Y\mathbf{Q})\mathbf{Q}^\top\mathbf{1}_N.\end{cases} \tag{36}$$

Let us simplify the expression. Since $\mathbf{Q}_1 = \frac{1}{\sqrt{m}}\mathbb{I}_C \otimes \mathbf{1}_m$ and $\mathbf{Q}_2 = \mathbb{I}_C \otimes \tilde{\mathbf{Q}}_2$, we have

$$\mathbf{1}_N^\top\mathbf{Q} = \sqrt{m}[\mathbf{1}_C^\top, \mathbb{O}], \quad \mathbf{Y}\mathbf{Q} = \sqrt{m}[\mathbb{I}_C, \mathbb{O}]. \tag{37}$$

Plugging (37) into (36), we see the dynamics can be decomposed into

$$\begin{cases}\dot{\mathbf{H}}_1 = & -(\kappa_d - \kappa_c)\mathbf{W}^\top(\mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top - \mathbb{I}_C) \\ & -(\kappa_c - \kappa_n)m\mathbf{W}^\top(\mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top - \mathbb{I}_C) \\ & -\kappa_n N\mathbf{W}^\top(\frac{1}{C}\mathbf{W}\mathbf{H}_1\mathbf{1}_C\mathbf{1}_C^\top + \mathbf{b}\mathbf{1}_C^\top - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top) \\ \dot{\mathbf{H}}_2 = & -(\kappa_d - \kappa_c)\mathbf{W}^\top\mathbf{W}\mathbf{H}_2 \\ \dot{\mathbf{W}} = & -m(\mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top - \mathbb{I}_C)\mathbf{H}_1^\top - m\mathbf{W}\mathbf{H}_2\mathbf{H}_2^\top \\ \dot{\mathbf{b}} = & -m(\mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top - \mathbb{I}_C)\mathbf{1}_C.\end{cases} \tag{38}$$

To further simplify (38), we define the following quantities

$$\mu_{\text{single}} := \kappa_d - \kappa_c, \quad \mu_{\text{class}} := \mu_{\text{single}} + m(\kappa_c - \kappa_n), \quad \mathbf{R}_1 := \mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top - \mathbb{I}_C. \tag{39}$$

Notice that μ_{single} and μ_{class} are the two largest eigenvalues of the block-structured kernel $\Theta^h_{s,s}(X)$ (see Table 1 for the eigndecomposition of a block-structured matrix), and $\mathbf{R}_1$ is a matrix of the stacked class-mean residuals, which is also defined in (17). The the dynamics (38) simplifies to

$$\begin{cases}\dot{\mathbf{H}}_1 = & -\mathbf{W}^\top(\mu_{\text{class}}\mathbf{R}_1 + \kappa_n N(\frac{1}{C}\mathbf{W}\mathbf{H}_1\mathbf{1}_C\mathbf{1}_C^\top + \mathbf{b}\mathbf{1}_C^\top - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top)) \\ \dot{\mathbf{H}}_2 = & -\mu_{\text{single}}\mathbf{W}^\top\mathbf{W}\mathbf{H}_2 \\ \dot{\mathbf{W}} = & -m(\mathbf{R}_1\mathbf{H}_1^\top - \mathbf{W}\mathbf{H}_2\mathbf{H}_2^\top) \\ \dot{\mathbf{b}} = & -m\mathbf{R}_1\mathbf{1}_C.\end{cases} \tag{40}$$

It remains to simplify the expression for $\dot{\mathbf{H}}_1$. By using the relation

$$\frac{1}{C}\mathbf{W}\mathbf{H}_1\mathbf{1}_C\mathbf{1}_C^\top + \mathbf{b}\mathbf{1}_C^\top - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top = \frac{1}{C}\mathbf{R}_1\mathbf{1}_C\mathbf{1}_C^\top, \tag{41}$$

we can deduce that the dynamics for $\dot{\mathbf{H}}_1$ in (40) can be expressed as (recalling that $N = mC$)

$$\dot{\mathbf{H}}_1 = -\mathbf{W}^\top\mathbf{R}_1(\mu_{\text{class}}\mathbb{I} + \kappa_n m\mathbf{1}_C\mathbf{1}_C^\top). \tag{42}$$

We notice that $(\mathbb{I}_C + \frac{\kappa_n m}{\mu_{\text{class}}}\mathbf{1}_C\mathbf{1}_C^\top)^{-1} = \mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top$, where $\alpha := \frac{\kappa_n m}{\mu_{\text{class}} + C\kappa_n m}$. Then we can derive the invariant of the training dynamics by direct computation of the time-derivative $\dot{\mathbf{E}}$, where

$$\mathbf{E} := \frac{1}{m}\mathbf{W}^\top\mathbf{W} - \frac{1}{\mu_{\text{class}}}\mathbf{H}_1(\mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top)\mathbf{H}_1^\top - \frac{1}{\mu_{\text{single}}}\mathbf{H}_2\mathbf{H}_2^\top \tag{43}$$

Since $\dot{\mathbf{E}} = \mathbb{O}$, we get that the quantity $\mathbf{E}$ remains constant in time. This completes the proof.

□

B.3 Proof of Theorem 5.2

We divide the proof into two main parts: the first one shows the emergence of NC1, and the second one shows NC2-4.

(NC1). Following the analysis in Section 3, the dynamics eventually enters the end of training phase (see Section 4.1). Then the dynamics in Theorem 5.1 simplifies to the following form:

$$\begin{cases} \dot{\mathbf{H}}_1 = \mathbb{O} \\ \dot{\mathbf{H}}_2 = -\mu_{\text{single}}\mathbf{W}^\top\mathbf{W}\mathbf{H}_2 \\ \dot{\mathbf{W}} = -m\mathbf{W}\mathbf{H}_2\mathbf{H}_2^\top \\ \dot{\mathbf{b}} = \mathbb{O} \end{cases} \tag{44}$$

As we note in Section 4, this dynamics is similar to the gradient flow of the unconstrained features models and is an instance of the class of hyperbolic dynamics, which is discussed in Section 5.2. During this phase the quantity

$$\tilde{\mathbf{E}} := \mu_{\text{single}}\mathbf{W}^\top\mathbf{W} - m\mathbf{H}_2\mathbf{H}_2^\top = m\mu_{\text{single}}(\mathbf{E} + \frac{1}{\mu_{\text{class}}}\mathbf{H}_1(\mathbb{I} - \alpha\mathbf{1}_C\mathbf{1}_C^\top)\mathbf{H}_1^\top) \tag{45}$$

does not change in time. Hence we can decouple the dynamic using the invariant as follows:

$$\begin{cases} \dot{\mathbf{H}}_2 = -\mu_{\text{single}}(\tilde{\mathbf{E}} + m\mathbf{H}_2\mathbf{H}_2^\top)\mathbf{H}_2 \\ \dot{\mathbf{W}} = -\mathbf{W}(\mu_{\text{single}}\mathbf{W}^\top\mathbf{W} - \tilde{\mathbf{E}}) \end{cases} \tag{46}$$

Since $\mathbf{E}$ is p.s.d (or zero, as a special case), $\tilde{\mathbf{E}}$ is p.s.d as well, and the eigendecomposition of the invariant is given by $\tilde{\mathbf{E}} = \sum_k c_k v_k v_k^\top$ for some coefficients $c_k \geq 0$ and a set of orthonormal vectors $v_k \in \mathbb{R}^n$. Then we also have $\mathbf{H}_2\mathbf{H}_2^\top = \sum_{k,l}\alpha_{kl}v_k v_l^\top$, where α_{kl} are symmetric (i.e. $\alpha_{kl} = \alpha_{lk}$) and $\alpha_{kk} \geq 0$ for all $k = 1, \dots n$ (since $\mathbf{H}_2\mathbf{H}_2^\top$ is symmetric and p.s.d.). Note that coefficients c_k here are constant while coefficients α_{kl} are time-dependent. Let us then write the dynamics for α_{kl} using the dynamics of $\mathbf{H}_2\mathbf{H}_2^\top$:

$$(\dot{\mathbf{H}_2\mathbf{H}_2^\top}) = -\tilde{\mathbf{E}}\mathbf{H}_2\mathbf{H}_2^\top - \mathbf{H}_2\mathbf{H}_2^\top\tilde{\mathbf{E}} - 2(\mathbf{H}_2\mathbf{H}_2^\top)^2 \tag{47}$$

Then for the elements of α we have:

$$\dot{\alpha}_{kl} = -\alpha_{kl}(c_k + c_l) - 2\sum_j \alpha_{kj}\alpha_{jl} \tag{48}$$

For the diagonal elements α_{kk}, this gives:

$$\dot{\alpha}_{kk} = -2c_k\alpha_{kk} - 2\sum_j \alpha_{kj}^2 \tag{49}$$

Since $c_k \geq 0$, $\alpha_{kk} \geq 0$ and $\alpha_{kj}^2 \geq 0$, we get that

$$\alpha_{kk} \xrightarrow[t\to\infty]{} 0 \quad \forall k \tag{50}$$

And, therefore, all the non-diagonal elements also tend to zero. Thus, we get that

$$\mathbf{H}_2\mathbf{H}_2^\top \xrightarrow[t\to\infty]{} \mathbb{O} \tag{51}$$

and thus

$$\mathbf{H}_2 \xrightarrow[t\to\infty]{} \mathbb{O} \tag{52}$$

Now we notice that from the expression for $\mathbf{H}_2$ in (24) it follows that $\mathbf{H}_2 = \mathbb{O}$ implies variability collapse, since it means that all the feature vectors within the same class are equal. Indeed, $\mathbf{H}^{(c)}\tilde{\mathbf{Q}}_2 = \mathbb{O} \in \mathbb{R}^{n\times(m-1)}$ means that there is a set of $m-1$ orthogonal vectors, which are all also orthogonal to $[h_i(x_1^c), \ldots, h_i(x_m^c)]$ for any $i = 1, \ldots, n$, where x_i^c are inputs from class c. However, there is only one vector (up to a constant) orthogonal to all the columns of $\tilde{\mathbf{Q}}_2$ in $\mathbb{R}^m$ and this vector is $\mathbf{1}_m$. Therefore, $[h_i(x_1^c), \ldots h_i(x_m^c)] = \gamma\mathbf{1}_m$ for some constant γ for any $i = 1, \ldots, n$. Thus, we indeed have $h(x_1^c) = \cdots = h(x_m^c)$, which constitutes variability collapse within classes. □

(NC2-4). Set $\beta = \frac{1}{C}$. We first show that zero global feature mean implies $\mathbf{b} = \beta\mathbf{1}_C$. At the end of training, since $\mathbf{R}_1 = \mathbb{O}$, we have

$$\mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top = \mathbb{I}_C \tag{53}$$

On the other hand, zero global mean implies $\mathbf{H}_1\mathbf{1}_C = C\langle h\rangle = \mathbb{O}$. Then multiplying (53) by $\mathbf{1}_C$ on the right, we get the desired expression for the biases. Given the zero global mean, we have

$$\frac{1}{m}\mathbf{W}^\top\mathbf{W} - \frac{1}{\mu_{\text{class}}}\mathbf{H}_1\mathbf{H}_1^\top - \frac{1}{\mu_{\text{single}}}\mathbf{H}_2\mathbf{H}_2^\top = \mathbf{E} - \frac{\alpha m C^2}{\mu_{\text{class}}}\langle h\rangle\langle h\rangle^\top = \mathbf{E} \tag{54}$$

By the proof of NC1, $\mathbf{H}_2 \to \mathbb{O}$. Together with the assumption that $\mathbf{E}$ is proportional to the limit of $\mathbf{W}^\top\mathbf{W}$ (or zero, as a special case), we obtain

$$\mu_{\text{class}}\mathbf{W}^\top\mathbf{W} - m\mathbf{H}_1\mathbf{H}_1^\top \to \gamma\mathbf{W}^\top\mathbf{W} \tag{55}$$

for some $\gamma \geq 0$. Note that since $\mathbf{H}_1\mathbf{H}_1^\top$ is p.s.d. this implies $\tilde{\lambda}_c := \mu_{\text{class}} - \gamma \geq 0$. By multiplying the left and right with appropriate factors, we have

$$\begin{cases} \mathbf{H}_1^\top(\tilde{\lambda}_c\mathbf{W}^\top\mathbf{W} - m\mathbf{H}_1\mathbf{H}_1^\top)\mathbf{H}_1 \to \mathbb{O} \\ \mathbf{W}(\tilde{\lambda}_c\mathbf{W}^\top\mathbf{W} - m\mathbf{H}_1\mathbf{H}_1^\top)\mathbf{W}^\top \to \mathbb{O}. \end{cases} \tag{56}$$

Consequently (according to (53))

$$\begin{cases} \tilde{\lambda}_c(\mathbb{I}_C - \beta\mathbf{1}_C\mathbf{1}_C^\top)^2 - m(\mathbf{H}_1^\top\mathbf{H}_1)^2 \to \mathbb{O} \\ \tilde{\lambda}_c(\mathbf{W}\mathbf{W}^\top)^2 - (\mathbb{I}_C - \beta\mathbf{1}_C\mathbf{1}_C^\top)^2 \to \mathbb{O} \end{cases} \tag{57}$$

Since both $\mathbf{W}\mathbf{W}^\top$ and $\mathbf{H}_1^\top\mathbf{H}_1$ are p.s.d., we have

$$\begin{cases} \mathbf{H}_1^\top\mathbf{H}_1 \to \sqrt{\frac{\tilde{\lambda}_c}{m}}(\mathbb{I}_C - \beta\mathbf{1}_C\mathbf{1}_C^\top) \\ \mathbf{W}\mathbf{W}^\top \to \sqrt{\frac{m}{\tilde{\lambda}_c}}(\mathbb{I}_C - \beta\mathbf{1}_C\mathbf{1}_C^\top). \end{cases} \tag{58}$$

To establish NC2, recall that $\mathbf{H}_1 = \left[\langle h\rangle_1, \ldots, \langle h\rangle_C\right]$ and that $\mathbf{M}$, as a normalized version of $\mathbf{H}_1$, satisfies

$$\mathbf{M}^\top\mathbf{M} \to \frac{1}{1-\beta}(\mathbb{I}_C - \beta\mathbf{1}_C\mathbf{1}_C^\top) = \frac{C}{C-1}(\mathbb{I}_C - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top).$$

To establish NC3, note that from (55) and (58) together, it follows that the limits of $\mathbf{M}$ and $\mathbf{W}^\top$ only differ by a constant multiplier.

To establish NC4, note that using NC3 we can write

$$\begin{aligned} \operatorname*{argmax}_c (\mathbf{W}h(x) + \mathbf{b})_c &= \operatorname*{argmax}_c (\mathbf{W}h(x))_c && (\mathbf{b} = \beta\mathbf{1}_C) \\ &\to \operatorname*{argmax}_c (\mathbf{M}^\top h(x))_c && \text{(NC3)} \\ &= \operatorname*{argmin}_c \|h(x) - \langle h\rangle_c\|_2. \end{aligned}$$

This completes the proof. □

B.4 General biases case

Proof. As in the proof of Theorem 5.2, at the end of training we have $\mathbf{W}\mathbf{H}_1 + \mathbf{b}\mathbf{1}_C^\top = \mathbb{I}_C$. Moreover, since $\mathbf{E} = \mathbb{O}$ and $\mathbf{H}_2 \to \mathbb{O}$, we have

$$\frac{1}{m}\mathbf{W}^\top\mathbf{W} - \frac{1}{\mu_{\text{class}}}\mathbf{H}_1(\mathbb{I} - \alpha\mathbf{1}_C\mathbf{1}_C^\top)\mathbf{H}_1^\top \to \mathbb{O}. \tag{59}$$

Multyplying the above expression to the left by $\mathbf{W}$ and to the right by $\mathbf{W}^\top$, we obtain the general expression (29) for the matrix $(\mathbf{W}\mathbf{W}^\top)^2$ mentioned in the main text:

$$(\mathbf{W}\mathbf{W}^\top)^2 \to \frac{m}{\mu_{\text{class}}}\Big(\mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top + (1-\alpha C)(C\mathbf{b}\mathbf{b}^\top - \mathbf{b}\mathbf{1}_C^\top - \mathbf{1}_C\mathbf{b}^\top)\Big). \tag{60}$$

This expression implies that the rows of the weights matrix may have varying separation angles in the general biases case, i.e., there is no symmetric structure is general. However, for constant biases $\mathbf{b} = \beta\mathbf{1}_C$, the above expression simplifies to

$$(\mathbf{W}\mathbf{W}^\top)^2 \to \frac{m}{\mu_{\text{class}}}\Big(\mathbb{I}_C - \frac{1}{C}\big(1-(1-\alpha C)(1-\beta C)^2\big)\mathbf{1}_C\mathbf{1}_C^\top\Big). \tag{61}$$

Since $\alpha < 1/C$ and $(1-\beta C)^2 \geq 0$, we have that $(1-(1-\alpha C)(1-\beta C)^2)/C \leq 1/C$. Therefore, the RHS of (61) is always p.s.d. and has a unique p.s.d square root proportional to $\mathbb{I}_C - \gamma\mathbf{1}_C\mathbf{1}_C^\top$ for some constant $\gamma < 1/C$. Denote $\rho := (1-(1-\alpha C)(1-\beta C)^2)/C$, then we have $\gamma = (1-\sqrt{1-C\rho})/C$. Note that $\rho < 1/C$ ensures that γ is well defined. Then the configuration of the final weights is given by

$$\mathbf{W}\mathbf{W}^\top \to \sqrt{\frac{m}{\mu_{\text{class}}}}\Big(\mathbb{I}_C - \gamma\mathbf{1}_C\mathbf{1}_C^\top\Big). \tag{62}$$

This means that the norms of all the weights rows are still equal, as in NC2. However, since $\gamma < 1/C$ if $\beta \neq 1/C$, the angle between these rows is smaller than in the ETF structure.

We can derive the configuration of the class means similarly by multyplying (59) to the left by $\mathbf{H}_1^\top$ and to the right by $\mathbf{H}_1$. In the general biases case, we get

$$\mathbf{H}_1^\top\mathbf{H}_1(\mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top)\mathbf{H}_1^\top\mathbf{H}_1 \to \frac{\mu_{\text{class}}}{m}\Big(\mathbb{I}_C - \mathbf{b}\mathbf{1}_C^\top - \mathbf{1}_C\mathbf{b}^\top + \|\mathbf{b}\|_2^2\mathbf{1}_C\mathbf{1}_C^\top\Big). \tag{63}$$

As with the weights, we see that this is not a symmetric structure in general. Thus, NC2 does not hold in the general biases case. However, for the constant biases $\mathbf{b} = \beta\mathbf{1}_C$, the above expression simplifies to

$$\mathbf{H}_1^\top\mathbf{H}_1(\mathbb{I}_C - \alpha\mathbf{1}_C\mathbf{1}_C^\top)\mathbf{H}_1^\top\mathbf{H}_1 \to \frac{\mu_{\text{class}}}{m}(\mathbb{I}_C - \beta\mathbf{1}_C\mathbf{1}_C^\top)^2. \tag{64}$$

Analogously to the previous derivations, we get that the unique p.s.d. square root of the RHS is given by $\mathbb{I}_C - \tilde{\rho}\mathbf{1}_C\mathbf{1}_C^\top$, where $\tilde{\rho} := (1-|1-\beta C|)/C < 1/C$ for $\beta \neq 1/C$. On the other hand, the unique p.s.d root of $\mathbb{I} - \alpha\mathbf{1}_C\mathbf{1}_C^\top$ is given by $\mathbb{I}_C - \phi\mathbf{1}_C\mathbf{1}_C^\top$, where $\phi := (1-\sqrt{1-\alpha C})/C$. Thus, we have the following

$$\sqrt{\frac{m}{\mu_{\text{class}}}}\mathbf{H}_1^\top\mathbf{H}_1(\mathbb{I}_C - \phi\mathbf{1}_C\mathbf{1}_C^\top) \to \mathbb{I}_C - \tilde{\rho}\mathbf{1}_C\mathbf{1}_C^\top. \tag{65}$$

Therefore, the structure of the last-layer features class means is given by

$$\mathbf{H}_1^\top\mathbf{H}_1 \to \sqrt{\frac{\mu_{\text{class}}}{m}}\Big(\mathbb{I}_C - \tilde{\rho}\mathbf{1}_C\mathbf{1}_C^\top\Big)\Big(\mathbb{I}_C - \frac{\phi}{1+\phi C}\mathbf{1}_C\mathbf{1}_C^\top\Big) = \sqrt{\frac{\mu_{\text{class}}}{m}}\Big(\mathbb{I}_C - \theta\mathbf{1}_C\mathbf{1}_C^\top\Big), \tag{66}$$

where $\theta := \tilde{\rho} + \phi/(1+\phi C) - C\tilde{\rho}\phi/(1+\phi C) < 1/C$ for $\beta \neq 1/C$. Thus, similarly to the classifier weights $\mathbf{W}$, the last-layer features class means form a symmetric structure with equal lengths and a separation angle smaller than in the ETF. However, the centralized class means given by $\mathbf{M} = \mathbf{H}_1(\mathbb{I}_C - \mathbf{1}_C\mathbf{1}_C^T/C)$ still form the ETF structure:

$$\mathbf{M}^\top\mathbf{M} \to \sqrt{\frac{\mu_{\text{class}}}{m}}\Big(\mathbb{I}_C - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top\Big). \tag{67}$$

This holds since the component proportional to $\mathbf{1}_C\mathbf{1}_C^\top$ on the RHS of equation (66) lies in the kernel of the ETF matrix $(\mathbb{I}_C - \mathbf{1}_C\mathbf{1}_C^\top/C)$. Thus, we conclude that NC2 holds in case of equal biases, while NC3 does not. □

Remark on $\alpha \to 0$ case: Simplifying the expressions for constants γ and θ, which define the angles in the configurations of the weights and the class means above, we get the following:

$$\gamma = \frac{1}{C}\big(1 - |1 - \beta C|\sqrt{1 - \alpha C}\big), \quad \theta = \frac{1}{C}\Big(1 - \frac{|1 - \beta C|}{2 - \sqrt{1 - \alpha C}}\Big). \tag{68}$$

Analyzing these expressions, we find that they are equal only if $1 - \alpha C = 1$, i.e. $\alpha = 0$. However, this can only hold if $\kappa_n = 0$ by definition of α, i.e., when the kernel Θ^h is zero on pairs of samples from different classes. While $\alpha \neq 0$ in general, there are certain settings where α approaches zero. Simplifying the expression for α, we can get the following

$$\alpha = \frac{1}{\frac{\kappa_c}{\kappa_n}(1 - \frac{1}{m}) + \frac{\kappa_d}{\kappa_n}\frac{1}{m} + (C - 1)}. \tag{69}$$

One can see that $\alpha \to 0$ if $C \to \infty$ or when $\kappa_c/\kappa_n \to \infty$. Since the kernel Θ^h is strongly aligned with the labels in our numerical experiments, the value of κ_c/κ_n is large in practice. Thus, α is not zero but indeed significantly smaller than $1/C$. Thus, in our numerical experiments the angles θ and γ are close to each other. However, we note that the equality of these two angles does not imply NC3, since the value of θ characterizes the angles between the non-centralized class means.

Remark on $\alpha \to 1/C$ case: If $\alpha = 1/C$, the equation (63) for the structure of the features class means with general (not equal) biases simplifies to

$$\mathbf{M}^\top \mathbf{M} \to \frac{\mu_{\text{class}}}{m}\Big(\mathbb{I}_C - \frac{1}{C}\mathbf{1}_C\mathbf{1}_C^\top\Big), \tag{70}$$

i.e., in this case the class means always exhibit the ETF structure, even without the assumption that all the biases are equal. Moreover, in this case $\gamma = 1/C$ as well. Thus, both NC2 and NC3 hold. While by definition $\alpha < 1/C$, we can analyze the cases when it approaches $1/C$ using the expression (69) again. One can see that when $m \to \infty$ and $\kappa_c/\kappa_n \to 1$, we have $\alpha \to 1/C$. However, the requirement $\kappa_c/\kappa_n \to 1$ implies that the kernel Θ^h does not distinguish between pairs of samples from the same class and from different classes. Such a property of the kernel is associated with poor generalization performance and does not occur in our numerical experiments.

C Numerical experiments

Implementation details We use JAX [8] and Flax (neural network library for JAX) [25] to implement all the DNN architectures and the training routines. This choice of the software allows to compute the empirical NTK of any DNN architecture effortlessly and efficiently. We compute the values of kernels Θ and Θ^h on the whole training batch ($m = 12$ samples per class, 120 samples in total) in case of ResNet20 and DenseNet40 to approximate the values $(\gamma_d, \gamma_c, \gamma_n)$ and $(\kappa_d, \kappa_c, \kappa_n)$, as well as the NTK alignment metrics, and compute the invariant $\mathbf{E}$ using these values. Since VGG11 and VGG16 architectures are much larger (over 10 million parameters) and computing their Jacobians is very memory-intensive, we use $m = 4$ samples per class (i.e., 40 samples in total) to approximate the kernels of these models. We compute all the other training metrics displayed in panes a-e of Figures 3, 4, 5, 6, 7, 8, 9, 10, 11 on the whole last batch of every second training epoch for all the architectures. The test accuracy is computed on the whole test set. To produce panes f-h of the same figures, we only compute the NC metrics and the test accuracy one time after 400 epochs of training for every learning rate. We use 30 logarithmically spaced learning rates in the range $\eta \in [10^{-4}, 10^{0.25}]$ for ResNet20 trained on MNIST and VGG11 trained on MNIST. For all the other architecture-dataset pairs we only compute the last 20 of these learning rates to reduce the computational costs, since the smallest learning rates do not yield models with acceptable performance.

Compute We executed the numerical experiments mainly on NVIDIA GeForce RTX 3090 Ti GPUs, each model was trained on a single GPU. In this setup, a single training run displayed in panes a-e of Figures 3, 4, 5, 6, 7, 8, 9, 10, 11 took approximately 3 hours for ResNet20, 6 hours for DenseNet40, 7 hours for VGG11, and 11 hours for VGG16. This adds up to a total of 312 hours to compute panes a-e of the figures. The computation time is mostly dedicated not to the training routine itself but to the large number of computationally-heavy metrics, which are computed every second epoch of a training run. Indeed, to approximate the values of Θ and Θ^h, one needs to compute $C(C + 1) + n(n + 1)$ kernels on a sample of size mC from the dataset, and each of the kernels requires computing a

gradient with respect to numerous parameters of a DNN. Additionally, the graphs in panes f-h of the same figures take around 1.5 hours for each learning rate value for ResNet20, 3 hours for DenseNet40, and 4 hours for VGG11 and VGG16, which adds up to approximately 1350 computational hours.

Results We include experiments on the following architecture-dataset pairs:

- Figure 3: VGG11 trained on MNIST
- Figure 4: VGG11 trained on FashionMNIST
- Figure 5: VGG16 trained on CIFAR10
- Figure 6: ResNet20 trained on MNIST
- Figure 7: ResNet20 trained on FashionMNIST
- Figure 8: ResNet20 trained on CIFAR10
- Figure 9: DenseNet40 trained on MNIST
- Figure 10: DenseNet40 trained on FashionMNIST
- Figure 11: DenseNet40 trained on CIFAR10

The experiments setup is described in Section 6. Panes a-h of Figures 3, 4, 5, 6, 7, 8, 9, 10, 11 are analogous to the same panes of Figure 2. We include additional pane i here, which displays the norms of the invariant terms corresponding to the feature matrix components $\mathbf{H}_1$ and $\mathbf{H}_2$, and the global features mean $\langle h \rangle$ at the end of training. One can see that the global features mean is relatively small in comparison with the class-means in every setup, and the "variance" term $\mathbf{H}_2$ is small for models that exhibit NC. We also add pane j, which displays the alignment of kernels Θ and Θ^h for every model at the end of training. One can see that the kernel alignments is typically stronger in models that exhibit NC.

C.1 Additional examples of the NTK block structure

We include the following additional illustrative figures (analogous to Figure 1 in the main text) that show the NTK block structure in dataset-architecture pairs covered in our experiments:

- Figure 13: VGG11 trained on MNIST
- Figure 14: VGG11 trained on FashionMNIST
- Figure 15: VGG16 trained on CIFAR10
- Figure 16: ResNet20 trained on FashionMNIST
- Figure 17: ResNet20 trained on CIFAR10
- Figure 18: DenseNet40 trained on MNIST
- Figure 19: DenseNet40 trained on FashionMNIST
- Figure 11: DenseNet40 trained on CIFAR10

Overall, the block structure pattern is visible in the traced kernels in all the figures. As expected, the block structure is more pronounced in the kernels where the final alignment values are higher. While the norms of the "non-diagonal" components of the kernels are generally smaller than the "diagonal" components in panes c) and d), we notice that there is a large variability in the norms of the "diagonal" components in some settings. This means that different neurons of the penultimate layer and different classification heads may contribute to the kernel unequally in some settings. Moreover, certain "non-diagonal" components of the last-layer kernel may have non-negligible effect in some settings. We discuss how one could generalize our analysis to account for these properties of the NTK in Appendix D.

C.2 Preliminary experiments with CE loss

While CE loss is a common choice for training DNN classifiers, our theoretical analysis and the experimental results only cover DNNs trained with MSE loss. For completeness, we provide experimental results for ResNet20 trained on MNIST with CE loss in Figure 12. One can see that

smaller invariant norm and higher invariant alignment correlate with NC in the figure. However, DNNs trained with CE loss overall reach better NC metrics but have much larger norm of the invariant in comparison with DNNs trained with MSE loss.

D Relaxation of the NTK Block-Structure Assumption

In this section, we first derive the dynamics equations of DNNs with a general block structure assumption on the last-layer kernel Θ^h (analogous to the equations presented in Theorem 4.1 and Theorem 5.1). Then we discuss a possible relaxation of Assumption 3.2, under which our main result regarding NC in Theorem 5.2 still holds.

D.1 Dynamics under General Block Structure Assumption

We first formulate the most general form of the block structure assumption on Θ^h as follows:

Assumption D.1. *Assume that $\Theta^h : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{n\times n}$ has the following block structure*

$$\Theta^h(x,x) = \mathbf{A}_d + \mathbf{A}_c + \mathbf{A}_n, \quad \Theta^h(x_i^c, x_j^c) = \mathbf{A}_c + \mathbf{A}_n, \quad \Theta^h(x_i^c, x_j^{c'}) = \mathbf{A}_n, \tag{71}$$

where $\mathbf{A}_{d,c,n} \in \mathbb{R}^{n\times n}$ are arbitrary p.s.d. matrices. Here x_i^c and x_j^c are two distinct inputs from the same class, and $x_j^{c'}$ is an input from class $c' \neq c$.

This assumption means that every kernel matrix $\Theta^h_{k,s}(X), k, s \in [1, n]$ still has at most three distinct values, corresponding to the inter-class, intra-class, and the diagonal values of the kernel. However, these values are arbitrary and may depend on the choice of $k, s \in [1, n]$.

Under the general block structure assumption, the gradient flow dynamics of DNNs with MSE loss takes the following form:

$$\begin{cases} \dot{\mathbf{H}} = & -\mathbf{A}_d\mathbf{W}^\top\mathbf{R} + m\mathbf{A}_c\mathbf{W}^\top\mathbf{R}_{\text{class}} + N\mathbf{A}_n\mathbf{W}^\top\mathbf{R}_{\text{global}} \\ \dot{\mathbf{W}} = & -\mathbf{R}\mathbf{H}^\top \\ \dot{\mathbf{b}} = & -\mathbf{R}_{\text{global}}\mathbf{1}_N. \end{cases} \tag{72}$$

This is the generalized version of the dynamics presented in Theorem 4.1. Consequently, the decomposed dynamics presented in Theorem 5.1 takes the following form under the general block structure assumption:

$$\begin{cases} \dot{\mathbf{H}}_1 &= -(\mathbf{A}_d + m\mathbf{A}_c)\mathbf{W}^\top\mathbf{R}_1 - m\mathbf{A}_n\mathbf{W}^\top\mathbf{R}_1\mathbf{1}_C\mathbf{1}_C^\top \\ \dot{\mathbf{H}}_2 &= -\mathbf{A}_d\mathbf{W}^\top\mathbf{W}\mathbf{H}_2 \\ \dot{\mathbf{W}} &= -m(\mathbf{R}_1\mathbf{H}_1^\top + \mathbf{W}\mathbf{H}_2\mathbf{H}_2^\top) \\ \dot{\mathbf{b}} &= -m\mathbf{R}_1\mathbf{1}_C. \end{cases} \tag{73}$$

The derivation of the above dynamics equations are identical to the proofs of Theorem 4.1 and Theorem 5.1 presented in Appendix B.

Rotation invariance We notice that the dynamics of $(\mathbf{W}, \mathbf{H})$ in (72) has to be rotation invariant, i.e., the equations should not be affected by a change of variables $\mathbf{W} \to \mathbf{W}\mathbf{Q}$, $\mathbf{H} \to \mathbf{Q}^\top\mathbf{H}$ for any orthogonal matrix $\mathbf{Q}$. This holds since the loss function only depends on the product $\mathbf{W}\mathbf{H}$, which does not change under rotation. This requirement puts conditions on the behavior of $\mathbf{A}_{d,c,n}$ under rotation. Indeed, assume that the rotation $\mathbf{W} \to \mathbf{W}\mathbf{Q}$, $\mathbf{H} \to \mathbf{Q}^\top\mathbf{H}$ for some $\mathbf{Q}$ corresponds to the following change of the kernel:

$$\mathbf{A}_{d,c,n} \to \tilde{\mathbf{A}}_{d,c,n}(\mathbf{Q}), \tag{74}$$

then the rotation invariance of the dynamics implies the following equality for any $\mathbf{Q}$:

$$\mathbf{Q}\tilde{\mathbf{A}}_d(\mathbf{Q})\mathbf{Q}^\top\mathbf{W}^\top\mathbf{R} + m\mathbf{Q}\tilde{\mathbf{A}}_c(\mathbf{Q})\mathbf{Q}^\top\mathbf{W}^\top\mathbf{R}_{\text{class}} + N\mathbf{Q}\tilde{\mathbf{A}}_n(\mathbf{Q})\mathbf{Q}^\top\mathbf{W}^\top\mathbf{R}_{\text{global}} \tag{75}$$

$$= \mathbf{A}_d\mathbf{W}^\top\mathbf{R} + m\mathbf{A}_c\mathbf{W}^\top\mathbf{R}_{\text{class}} + N\mathbf{A}_n\mathbf{W}^\top\mathbf{R}_{\text{global}}. \tag{76}$$

These equations are satisfied trivially with our initial assumption, where $\mathbf{A}_{d,c,n} = \tilde{\mathbf{A}}_{d,c,n}(\mathbf{Q}) \propto \mathbb{I}_n$. However, as we can see, any generalized assumption should specify the behavior of the kernel under rotation, and satisfy the above equation.

For general $\mathbf{A}_{d,c,n}$, the following behavior under rotation trivially satisfies the above condition: $\tilde{\mathbf{A}}_{d,c,n}(\mathbf{Q}) = \mathbf{Q}^\top \mathbf{A}_{d,c,n}\mathbf{Q}$. This behaviour of the kernel under rotation is intuitive, since it implies that the gradients of the last-layer features h are rotated in the same way as the features. However, we note that gradients of parametrized functions do not in general behave this way, since the rotation of the function has to be realized by a certain change of parameters. Consider, for instance, a one-hidden-layer linear network with weights $\mathbf{V}$ in the first layer. Then we have $\mathbf{H} = \mathbf{V}X$, and a rotation $\mathbf{H} \to \mathbf{Q}^\top \mathbf{H}$ corresponds to the change of parameters $\mathbf{V} \to \mathbf{Q}^\top \mathbf{V}$. In this case, the kernel does not change under rotation, i.e., $\tilde{\mathbf{A}}_{d,c,n}(\mathbf{Q}) = \mathbf{A}_{d,c,n}$.

Dynamics invariant We note that the dynamics in 73 does not in general have an invariant analogous to the one we identified in Theorem 5.1. Indeed, if we define a quantity $\mathbf{E} := \mathbf{W}^\top \mathbf{W} - c_1 \mathbf{H}_1 \mathbf{H}_1^\top - c_2 \mathbf{H}_2 \mathbf{H}_2^\top$ for some constants $c_{1,2} \in \mathbb{R}$, and additionally assume centered global means $\mathbf{H}_1 \mathbf{1}_C = 0$, we get the following expression for the derivative of $\mathbf{E}$:

$$\dot{\mathbf{E}} = \Big(c_1(\mathbf{A}_d + m\mathbf{A}_c) - m\mathbb{I}_n\Big)\mathbf{W}^\top \mathbf{R}_1 \mathbf{H}_1^\top - \mathbf{H}_1 \mathbf{R}_1^\top \mathbf{W}\Big(c_1(\mathbf{A}_d + m\mathbf{A}_c)^\top - m\mathbb{I}_n\Big) \tag{77}$$

$$+ \Big(c_2 \mathbf{A}_d - m\mathbb{I}_n\Big)\mathbf{W}^\top \mathbf{W}\mathbf{H}_2 \mathbf{H}_2^\top - \mathbf{H}_2 \mathbf{H}_2^\top \mathbf{W}^\top \mathbf{W}\Big(c_2 \mathbf{A}_d^\top - m\mathbb{I}_n\Big), \tag{78}$$

which is not equal to zero with arbitrary matrices $\mathbf{A}_{d,c}$.

D.2 Neural Collapse under Relaxed Block Structure Assumption

We now propose a relaxation of our main assumption, under which our main result regarding NC in Theorem 5.2 still holds. In terms of Assumption D.1 on the general block structure of Θ^h, our initial Assumption 3.2 in the main text is the special case with $\mathbf{A}_n = \kappa_n \mathbb{I}_n$, $\mathbf{A}_c = (\kappa_c - \kappa_n)\mathbb{I}_n$, $\mathbf{A}_d = (\kappa_d - \kappa_c)\mathbb{I}_n$. The relaxed assumption can be formulated as follows in terms of matrices $\mathbf{A}_{d,c,n}$:

Assumption D.2. *Assume that $\mathbf{A}_n$ is an arbitrary p.s.d. matrix and $(\mathbf{A}_c, \mathbf{A}_d)$ satisfy the following conditions:*

$$\mathbf{A}_c = \kappa_c \mathbb{I}_n + \mathbf{N}_c, \mathbf{A}_d = \kappa_d \mathbb{I}_n + \mathbf{N}_d, \tag{79}$$

where $\mathbf{N}_{c,d}^\top \in \ker(\mathbf{R}^\top \mathbf{W})$, i.e., $\mathbf{N}_{c,d}\mathbf{W}^\top \mathbf{R} = \mathbb{O}$. Further, assume that the kernel changes under rotation with an orthogonal matrix $\mathbf{Q}$ as follows:

$$\tilde{\mathbf{A}}_{d,c,n}(\mathbf{Q}) = \mathbf{Q}^\top \mathbf{A}_{d,c,n}\mathbf{Q}. \tag{80}$$

Since $\mathbf{A}_n$ is arbitrary, this relaxation allows arbitrary non-zero values of non-diagonal kernels $\Theta^h_{k,s}$ with $k \neq s$. The following observations justify the consistency of the above assumption:

- Since $\mathbf{W} \in \mathbb{R}^{C \times n}$, $\mathbf{R} \in \mathbb{R}^{C \times N}$ and $N > n > C$, $\mathbf{R}^\top \mathbf{W}$ has a non-empty kernel (possibly time-dependent).
- The dynamics is rotation invariant under the assumption, i.e., the equation (75) holds.
- The expression of the assumption is rotation invariant, in a sense that $\tilde{\mathbf{A}}_{d,c}(\mathbf{Q}) = \kappa_{d,c}\mathbb{I}_n + \tilde{\mathbf{N}}_{d,c}(\mathbf{Q})$, where $\tilde{\mathbf{N}}_{d,c}^\top \in \ker(\mathbf{R}^\top \mathbf{W}\mathbf{Q})$ for any orthogonal $\mathbf{Q}$.

Under the above assumption, the derivative in 77 becomes zero, so the dynamics has an invariant of the form $\mathbf{E} := \mathbf{W}^\top \mathbf{W} - c_1 \mathbf{H}_1 \mathbf{H}_1^\top - c_2 \mathbf{H}_2 \mathbf{H}_2^\top$. Moreover, the statement and the proof of our main Theorem 5.2 remains unchanged. Thus, DNNs satisfying the conditions of Theorem 5.2 display NC under Assumption D.2.

D.3 Discussion

The analysis of the DNNs dynamics is simplified significantly by assuming that Θ^h has a block structure. However, formulating a reasonable and consistent assumption on the NTK and its components is non-trivial. The Assumption 3.2 that we used in the main text is justified by the empirical results but may not capture all the relevant properties of the NTK. We believe that studying DNNs' dynamics under a more general or a more reasonable assumption on the NTK is a promising future work direction. The relaxed block structure assumption proposed in this section is the first step into this direction.

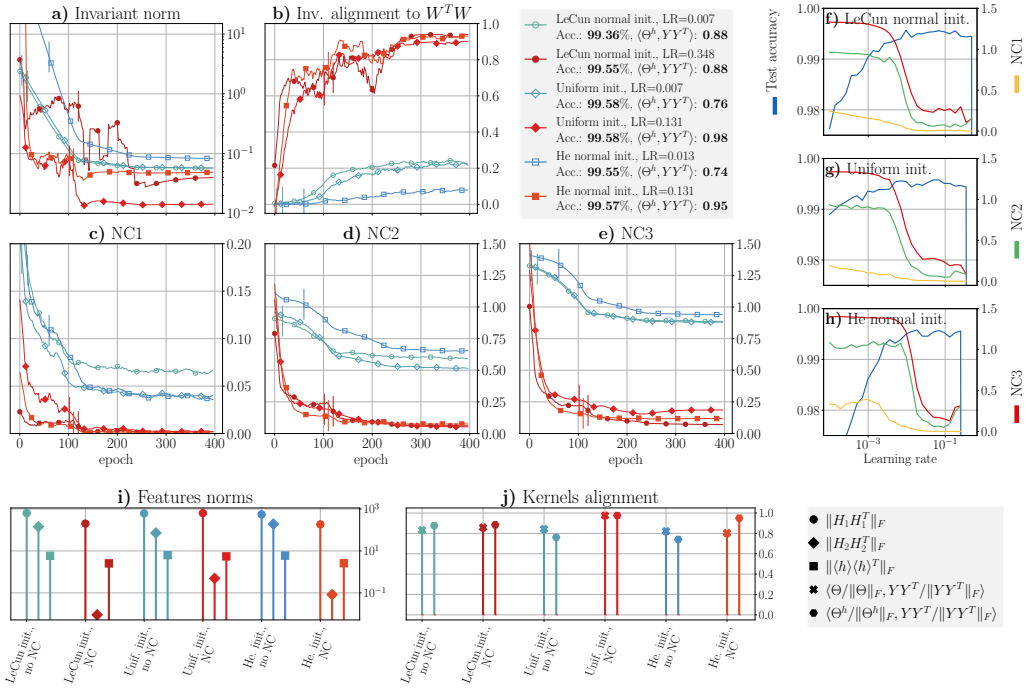


Figure 3: VGG11 trained on MNIST. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h\rangle\langle h\rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

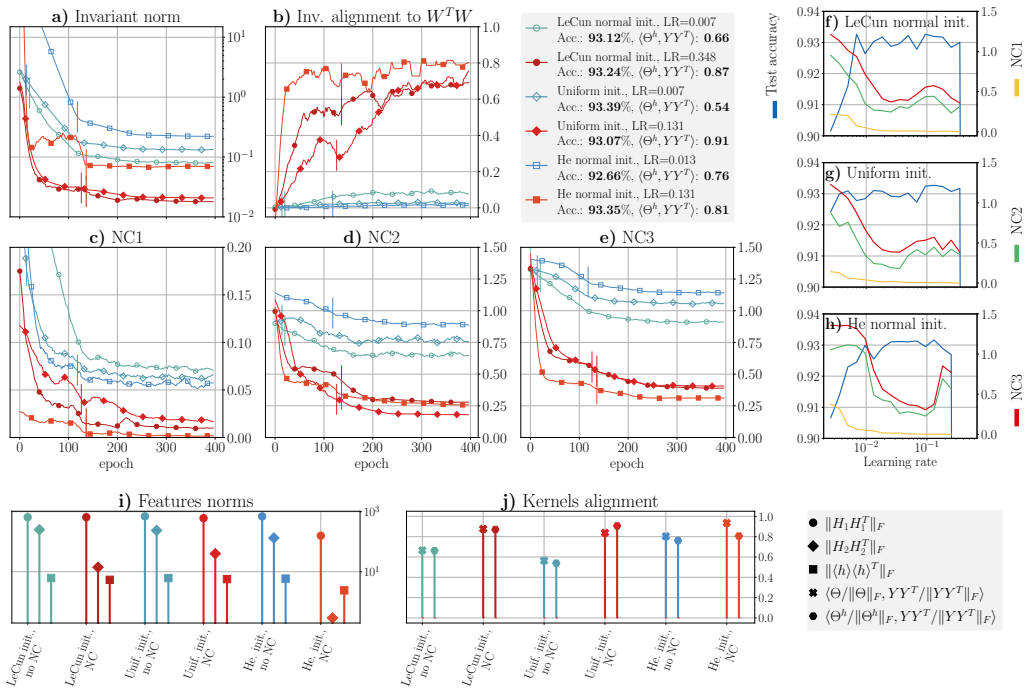


Figure 4: VGG11 trained on FashionMNIST. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h\rangle\langle h\rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

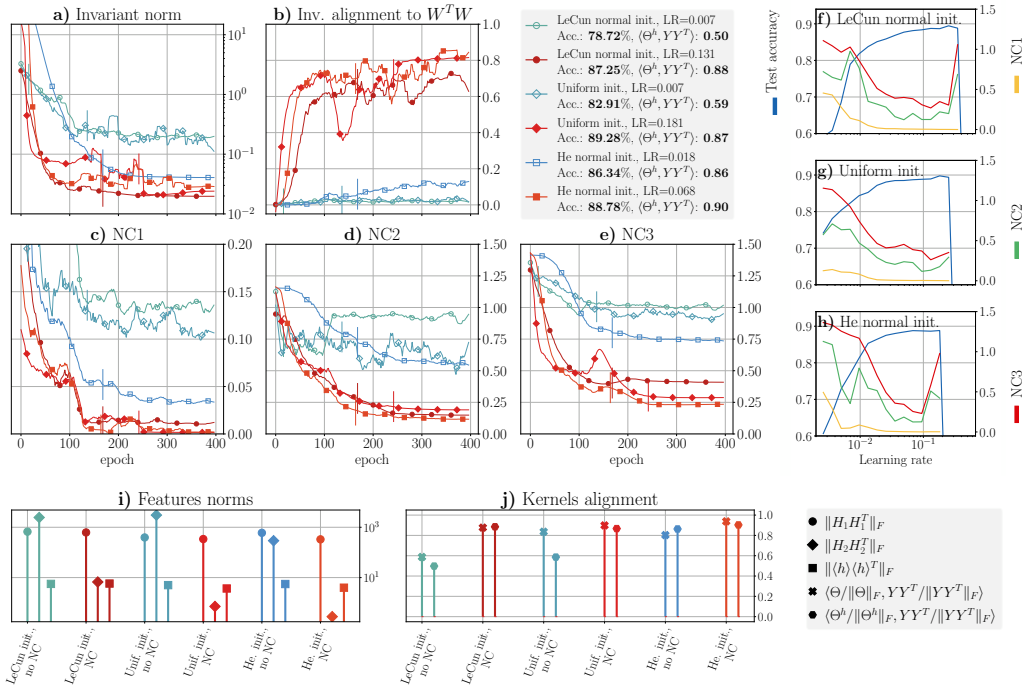


Figure 5: VGG16 trained on CIFAR10. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

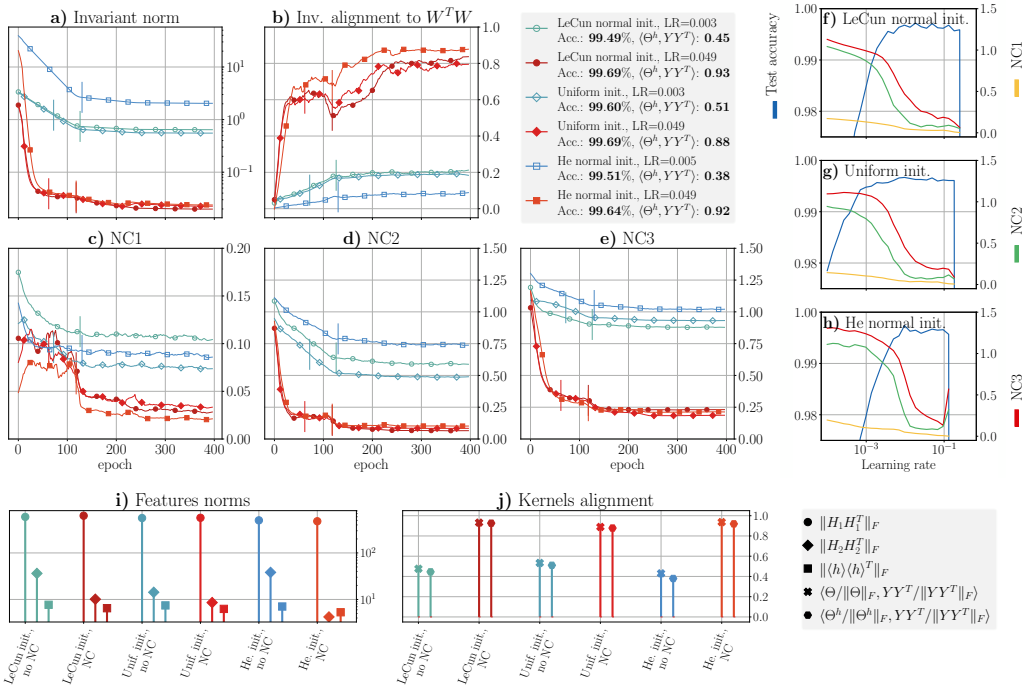


Figure 6: ResNet20 trained on MNIST. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

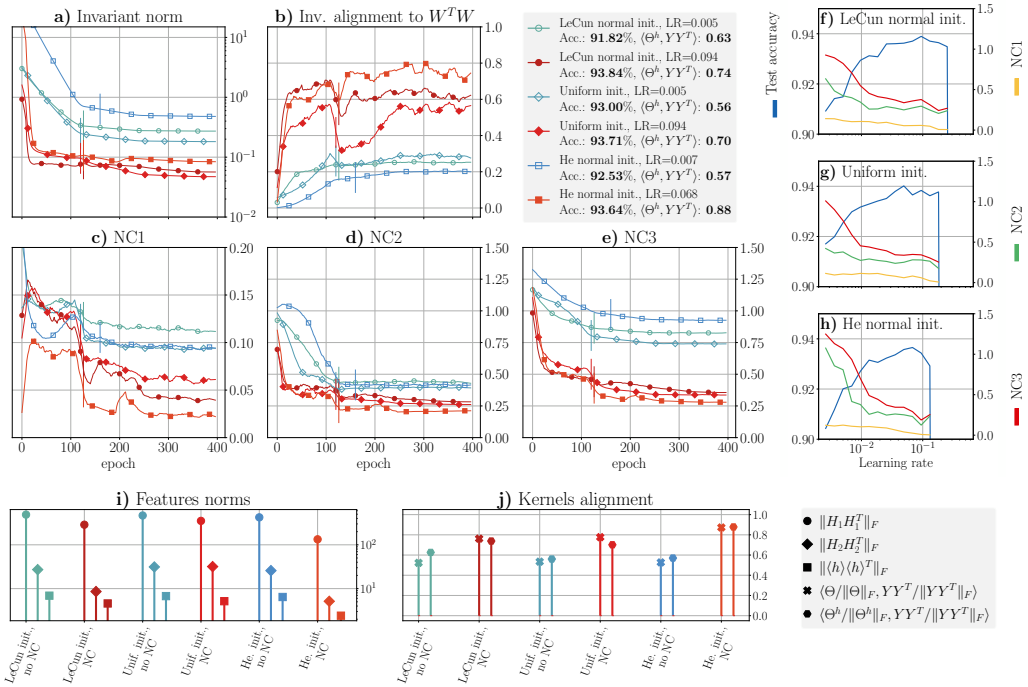


Figure 7: ResNet20 trained on FashionMNIST. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

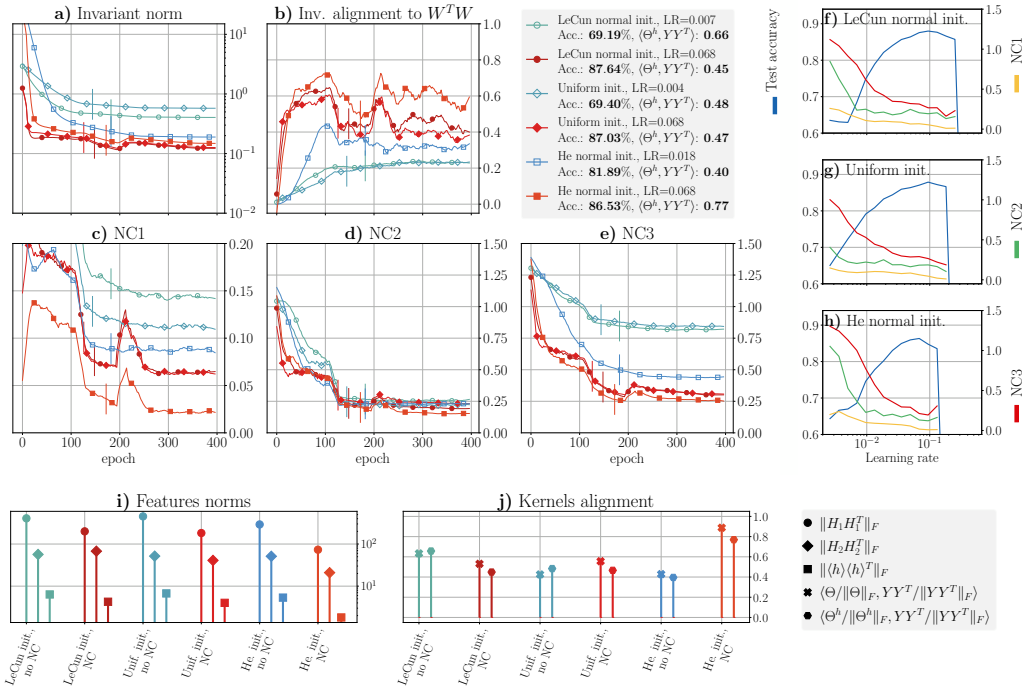


Figure 8: ResNet20 trained on CIFAR10. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

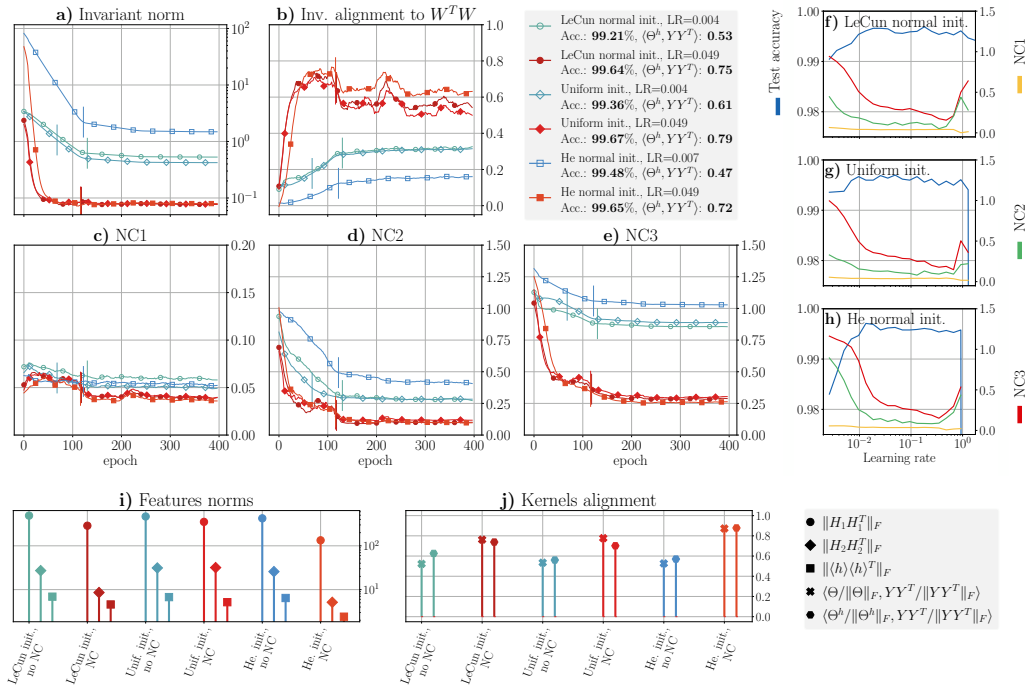


Figure 9: DenseNet40 trained on MNIST. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

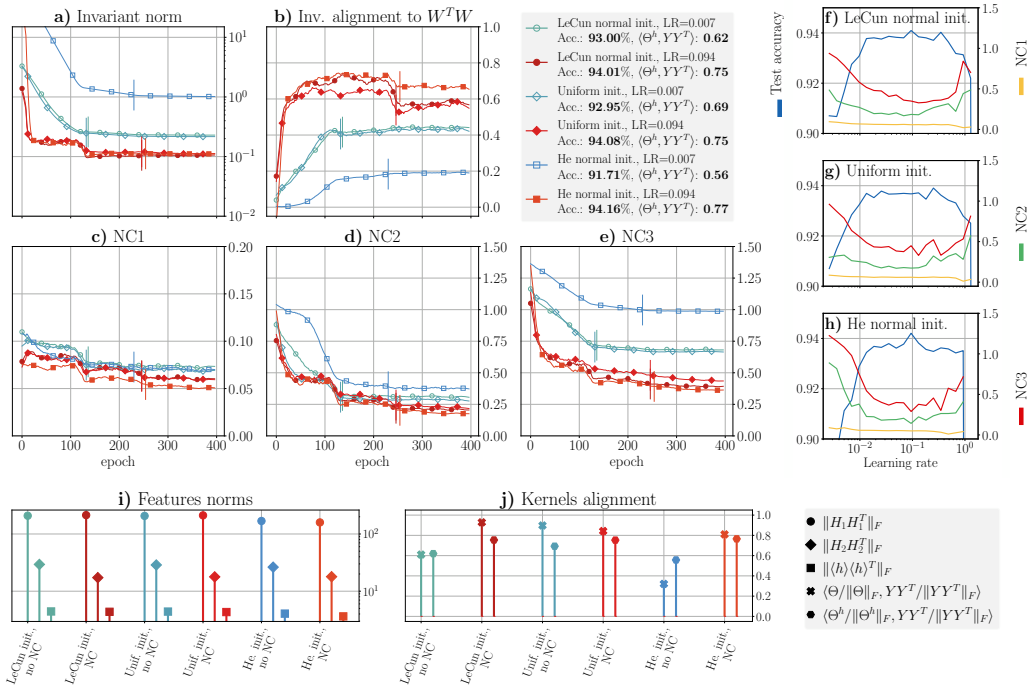


Figure 10: DenseNet40 trained on FashionMNIST. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

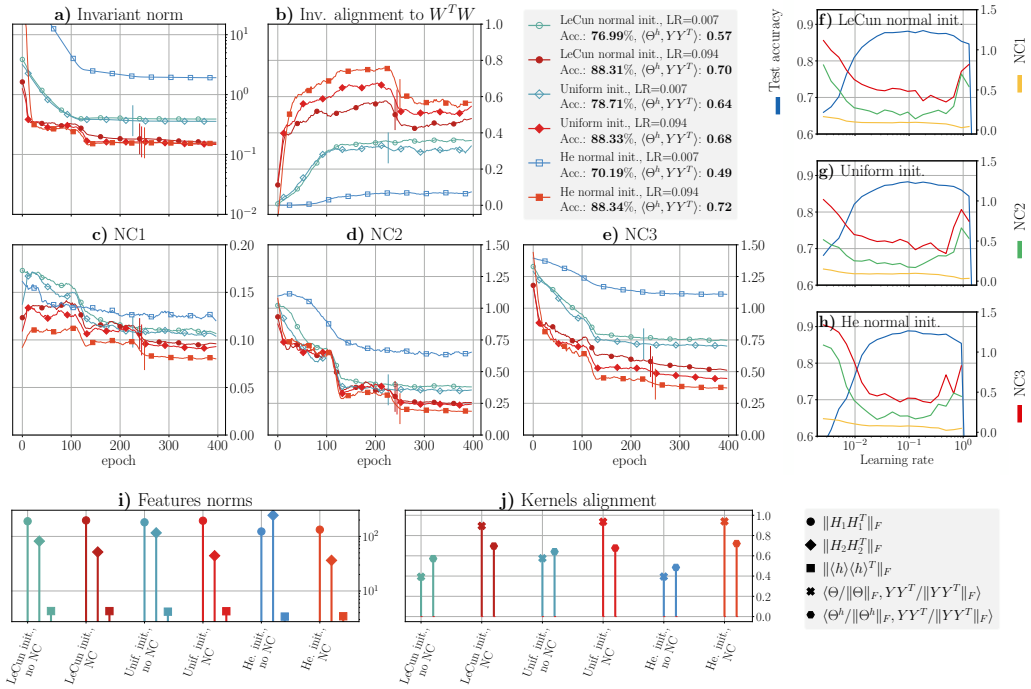


Figure 11: DenseNet40 trained on CIFAR10. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

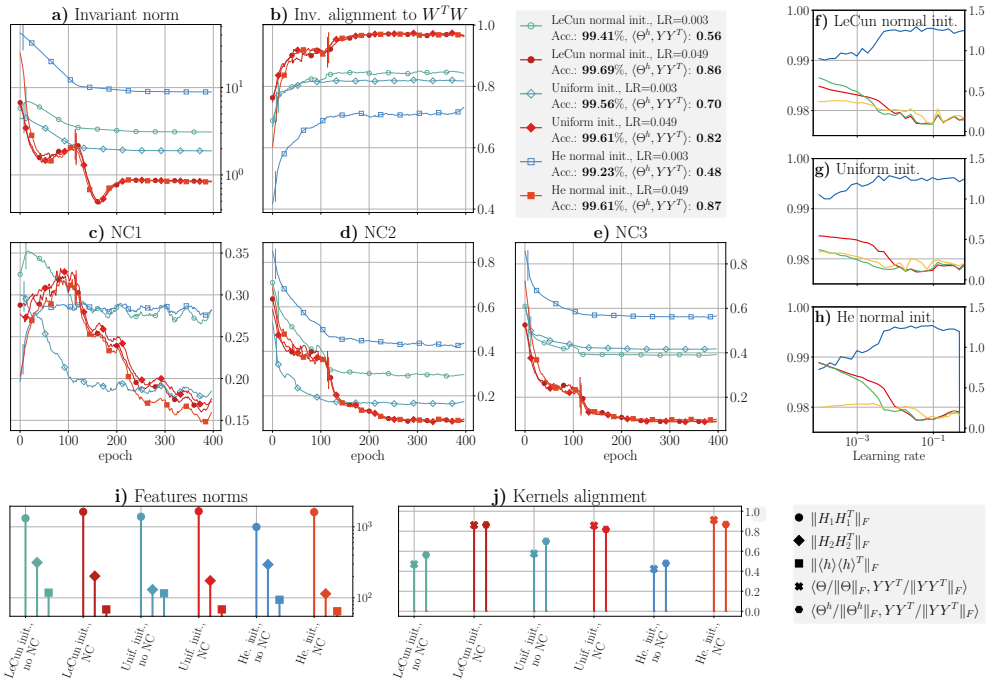


Figure 12: ResNet20 trained on MNIST with *CE loss*. See Figure 2 for the description of panes a-h. **i)** Norms of matrices $\mathbf{H}_1\mathbf{H}_1^\top$, $\mathbf{H}_2\mathbf{H}_2^\top$, and $\langle h \rangle \langle h \rangle^\top$ at the end of training. **j)** Alignment of kernels Θ and Θ^h at the end of training. The color in panes i-j is the color of the same model in panes a-e.

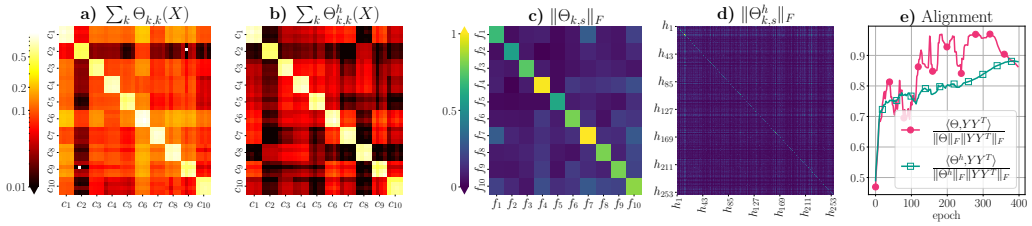


Figure 13: NTK block structure of VGG11 trained on MNIST. LeCun normal initialization, initial learning rate 0.131. The kernel is computed on a random data subset with 4 samples from each class. See Figure 1 for the description of panes.

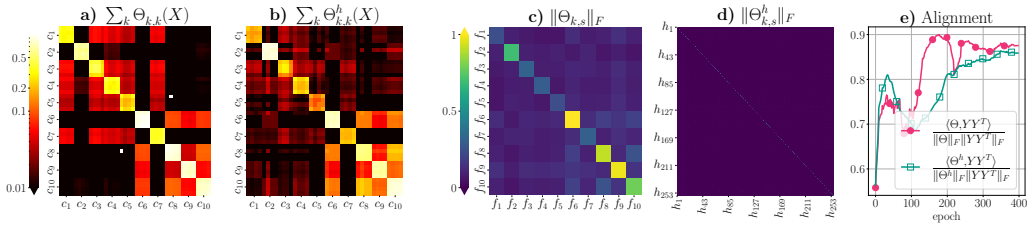


Figure 14: NTK block structure of VGG11 trained on FashionMNIST. LeCun normal initialization, initial learning rate 0.049. The kernel is computed on a random data subset with 4 samples from each class. See Figure 1 for the description of panes.

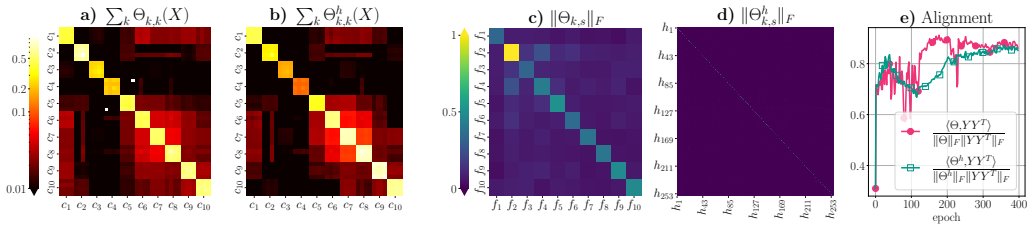


Figure 15: NTK block structure of VGG11 trained on CIFAR10. LeCun normal initialization, initial learning rate 0.131. The kernel is computed on a random data subset with 4 samples from each class. See Figure 1 for the description of panes.

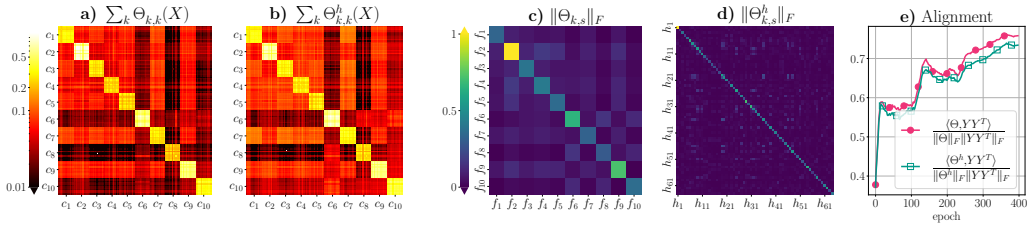


Figure 16: NTK block structure of ResNet20 trained on FashionMNIST. LeCun normal initialization, initial learning rate 0.094. The kernel is computed on a random data subset with 12 samples from each class. See Figure 1 for the description of panes.

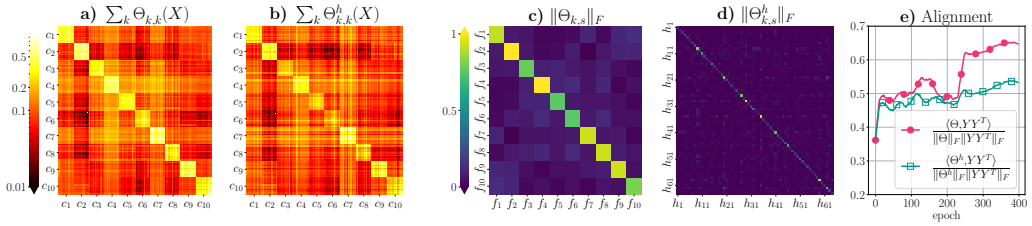


Figure 17: NTK block structure of ResNet20 trained on CIFAR10. LeCun normal initialization, initial learning rate 0.068. The kernel is computed on a random data subset with 12 samples from each class. See Figure 1 for the description of panes.

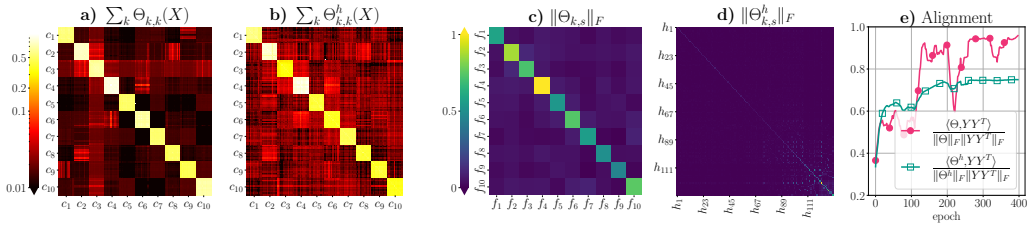


Figure 18: NTK block structure of DenseNet40 trained on MNIST. LeCun normal initialization, initial learning rate 0.049. The kernel is computed on a random data subset with 12 samples from each class. See Figure 1 for the description of panes.

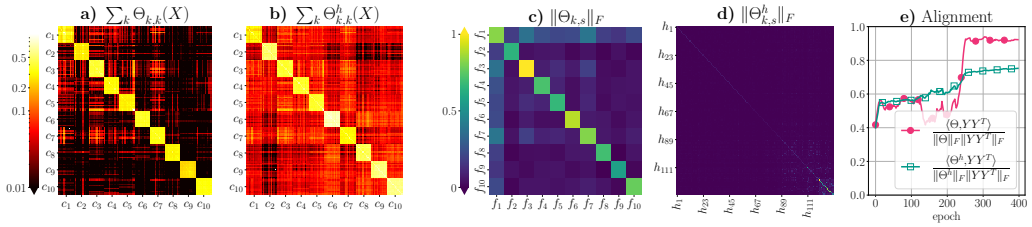


Figure 19: NTK block structure of DenseNet40 trained on FashionMNIST. LeCun normal initialization, initial learning rate 0.094. The kernel is computed on a random data subset with 12 samples from each class. See Figure 1 for the description of panes.

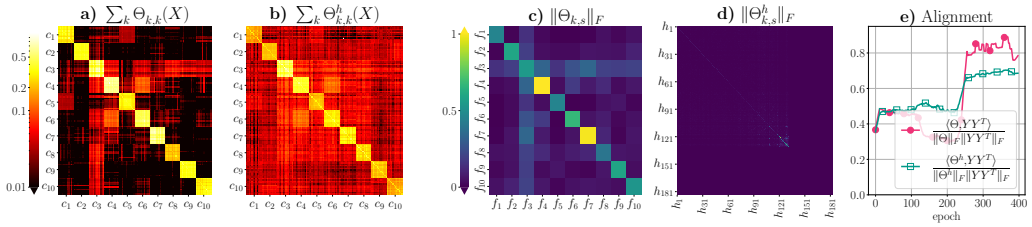


Figure 20: NTK block structure of DenseNet40 trained on CIFAR10. LeCun normal initialization, initial learning rate 0.094. The kernel is computed on a random data subset with 12 samples from each class. See Figure 1 for the description of panes.

Chapter 4

Conclusions and Future Work

In this thesis, we explored insights and limitations of the NTK regime for the analysis of DNNs' training dynamics. While we acknowledge that the introduction of the NTK regime was a breakthrough in deep learning theory, our contributions underscore that this regime frequently fails to accurately capture the intricacies of real-world DNNs' dynamics.

Our contributions regarding the limitations of the NTK regime focused specifically on deep networks, i.e., those with depth comparable to width. Our theoretical analysis concerned fully-connected DNNs with ReLU activation, while our empirical analysis additionally covered DNNs with sigmoid activation. We demonstrated that properties of the NTK at initialization and during training significantly depend on the initialization setup. Namely, we showed that the NTK of a deep network is random at initialization and changes during training if the network is initialized in the chaotic phase or at the EOC. While in case of initialization in the ordered phase the NTK is approximately deterministic and changes insignificantly during the first GD step, its structural changes during the entire training process are still non-trivial in the empirical studies. This critical examination contributes to a nuanced understanding of the limitations of relying solely on the NTK regime to characterize the training dynamics of DNNs.

While our analysis of the NTK behaviour answers many questions regarding the NTK regime applicability for fully-connected DNNs, numerous open questions remain, suggesting several avenues for future research:

- **Architectures with weights sharing:** While the literature extensively covers the infinite-width limit of the NTK for various architectures, the consideration of the infinite-depth-and-width regime, to our knowledge, remains confined to fully-connected DNNs. The challenge in generalizing our analysis to different architectures lies in addressing weights sharing, which introduces additional dependencies among neurons within the same layer. Although our theoretical approach may potentially extend to architectures like residual NNs, novel conceptual frameworks are likely required for architectures incorporating weight sharing. Importantly, even if theoretical analysis

proves challenging, we believe that empirical results on the NTK statistics and structure across diverse architectures would constitute a valuable contribution to the literature.

- **Dynamics beyond the first GD step:** Describing the evolution of the NTK beyond the initial GD step presents a significant theoretical challenge. Some studies on the NTK alignment considered the NTK evolution in toy models to show that alignment with the labels matrix is in some sense optimal. However, capturing the complete dynamics of the NTK for realistic DNNs currently seems infeasible due to the complex nature of its non-linear dynamics.

Our contributions regarding the kernel regime of DNNs with block-structured NTK proposed a new perspective on the NTK regime. Instead of considering the dynamics with the infinite-width NTK computed theoretically at initialization, we proposed to make assumptions on the NTK at the end of training, motivated by our understanding of the empirical NTK's properties. Given the approximate block structure observed in the empirical NTK of well-trained classification DNNs, our NTK block structure assumption provides an approximation of the end-of-training NTK for such networks. This assumption allowed us to analyze the dynamics of the last two layers of a DNN at the end of training, and derive conditions for convergence to NC. We believe that numerous future work directions exist in this field as well:

- **Relaxing the NTK block structure assumption:** While the NTK of well-trained DNNs indeed demonstrates an approximate block structure, it is also evident that the NTK values often exhibit considerable variance in real-world DNNs. Consequently, incorporating stochasticity into the dynamics with block-structured NTK is a promising avenue for future research. Additionally, empirical observations reveal the phenomenon of specialization within the NTK, where the kernel matrix associated with specific output neurons aligns more closely with the labels of their respective classes. In the context of block-structured kernels, specialization entails distinct values in blocks corresponding to different classes. Therefore, extending our theory to encompass block-structured kernels with specialization is another possible direction for future work.
- **Other empirical phenomena of DNNs:** Our work employed assumptions on the NTK structure to analyze specifically the NC phenomenon. However, similar assumptions could potentially be applied to investigate other empirical phenomena of DNNs, particularly towards the end of training. One such phenomenon, which may be related to the NTK dynamics, is the Edge of Stability behavior observed in DNNs during training. Exploring these connections could provide further insights into the underlying mechanisms governing DNN training dynamics.

Overall, we believe that empirical observations should play a crucial role in unveiling theory of DNNs' dynamics and deep learning in general.

Bibliography

Adcock, B. and Dexter, N. (2021). The Gap between Theory and Practice in Function Approximation with Deep Neural Networks. *SIAM Journal on Mathematics of Data Science*, 3(2):624–655.

Adlam, B. and Pennington, J. (2020). The Neural Tangent Kernel in High Dimensions: Triple Descent and a Multi-Scale Theory of Generalization. In *Proceedings of the 37th International Conference on Machine Learning*, pages 74–84. PMLR.

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*, pages 156–164. PMLR.

Alemohammad, S., Wang, Z., Balestriero, R., and Baraniuk, R. (2020). The Recurrent Neural Tangent Kernel. In *International Conference on Learning Representations*.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. (2019a). Fine-Grained Analysis of Optimization and Generalization for Overparameterized Two-Layer Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 322–332. PMLR.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. R., and Wang, R. (2019b). On Exact Computation with an Infinitely Wide Neural Net. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Atanasov, A., Bordelon, B., and Pehlevan, C. (2021). Neural Networks as Kernel Learners: The Silent Alignment Effect. In *International Conference on Learning Representations*.

Azulay, S., Moroshko, E., Nacson, M. S., Woodworth, B. E., Srebro, N., Globerson, A., and Soudry, D. (2021). On the Implicit Bias of Initialization Shape: Beyond Infinitesimal Mirror Descent. In *Proceedings of the 38th International Conference on Machine Learning*, pages 468–477. PMLR.

Bah, B., Rauhut, H., Terstiege, U., and Westdickenberg, M. (2022). Learning deep linear neural networks: Riemannian gradient flows and convergence to global minimizers. *Information and Inference: A Journal of the IMA*, 11(1):307–353.

Baratin, A., George, T., Laurent, C., Hjelm, R. D., Lajoie, G., Vincent, P., and Lacoste-Julien, S. (2021). Implicit Regularization via Neural Feature Alignment. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 2269–2277. PMLR.

Bartlett, P. L. and Maass, W. (2003). Vapnik-chervonenkis dimension of neural nets. *The handbook of brain theory and neural networks*, pages 1188–1192.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854.

Belkin, M., Hsu, D., and Xu, J. (2020). Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180.

Berner, J., Grohs, P., Kutyniok, G., and Petersen, P. (2021). The modern mathematics of deep learning. *arXiv preprint arXiv:2105.04026*.

Bietti, A. and Mairal, J. (2019). On the Inductive Bias of Neural Tangent Kernels. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Buchanan, S., Gilboa, D., and Wright, J. (2021). Deep networks and the multiple manifold problem. In *International Conference on Learning Representations*.

Cao, Y. and Gu, Q. (2019). Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in neural information processing systems*, 32.

Chen, S., He, H., and Su, W. (2020). Label-Aware Neural Tangent Kernel: Toward Better Generalization and Local Elasticity. In *Advances in Neural Information Processing Systems*, volume 33, pages 15847–15858. Curran Associates, Inc.

Chizat, L., Oyallon, E., and Bach, F. (2019). On Lazy Training in Differentiable Programming. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Chou, H.-H., Gieshoff, C., Maly, J., and Rauhut, H. (2024). Gradient descent for deep matrix factorization: Dynamics and implicit bias towards low rank. *Applied and Computational Harmonic Analysis*, 68:101595.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., and Usunier, N. (2017). Parseval Networks: Improving Robustness to Adversarial Examples. In *Proceedings of the 34th International Conference on Machine Learning*, pages 854–863. PMLR.

Cooper, Y. (2021). Global Minima of Overparameterized Neural Networks. *SIAM Journal on Mathematics of Data Science*, 3(2):676–691.

Cristianini, N., Shawe-Taylor, J., Elisseeff, A., and Kandola, J. (2001). On Kernel-Target Alignment. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314.

Demirkaya, A., Chen, J., and Oymak, S. (2020). Exploring the role of loss functions in multiclass classification. In *54th Annual Conference on Information Sciences and Systems, CISS 2020, Princeton, NJ, USA, March 18-20, 2020*, pages 1–5. IEEE.

DeVore, R., Hanin, B., and Petrova, G. (2021). Neural network approximation. *Acta Numerica*, 30:327–444.

Du, S. S., Hou, K., Salakhutdinov, R. R., Poczos, B., Wang, R., and Xu, K. (2019). Graph Neural Tangent Kernel: Fusing Graph Neural Networks with Graph Kernels. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Eldan, R. and Shamir, O. (2016). The Power of Depth for Feedforward Neural Networks. In *Conference on Learning Theory*, pages 907–940. PMLR.

Elkabetz, O. and Cohen, N. (2021). Continuous vs. Discrete Optimization of Deep Neural Networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 4947–4960. Curran Associates, Inc.

Fokina, D. and Oseledets, I. (2020). Growing axons: Greedy learning of neural networks with application to function approximation.

Fort, S., Dziugaite, G. K., Paul, M., Kharaghani, S., Roy, D. M., and Ganguli, S. (2020). Deep learning versus kernel learning: An empirical study of loss landscape geometry and the time evolution of the Neural Tangent Kernel. In *Advances in Neural Information Processing Systems*, volume 33, pages 5850–5861. Curran Associates, Inc.

Fout, A., Byrd, J., Shariat, B., and Ben-Hur, A. (2017). Protein interface prediction using graph convolutional networks. *Advances in neural information processing systems*, 30.

Funahashi, K.-I. (1989). On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192.

Geiger, M., Jacot, A., Spigler, S., Gabriel, F., Sagun, L., d'Ascoli, S., Biroli, G., Hongler, C., and Wyart, M. (2020). Scaling description of generalization with number of parameters in deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(2):023401.

Gönen, M. and Alpaydin, E. (2011). Multiple Kernel Learning Algorithms. *Journal of Machine Learning Research*, 12(64):2211–2268.

Graves, A., Mohamed, A.-r., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee.

Griewank, A. and Walther, A. (2008). *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM.

Gunasekar, S., Lee, J., Soudry, D., and Srebro, N. (2018). Characterizing Implicit Bias in Terms of Optimization Geometry. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1832–1841. PMLR.

Hanin, B. (2019). Universal Function Approximation by Deep Neural Nets with Bounded Width and ReLU Activations. *Mathematics*, 7(10):992.

Hanin, B. and Nica, M. (2019). Finite Depth and Width Corrections to the Neural Tangent Kernel. In *International Conference on Learning Representations*.

Hanin, B. and Rolnick, D. (2019). Deep ReLU Networks Have Surprisingly Few Activation Patterns. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Hanin, B. and Sellke, M. (2018). Approximating Continuous Functions by ReLU Nets of Minimal Width.

Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. (2022). Surprises in high-dimensional ridgeless least squares interpolation. *Annals of statistics*, 50(2):949–986.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Hu, Z. and Huang, H. (2021). On the Random Conjugate Kernel and Neural Tangent Kernel. In *Proceedings of the 38th International Conference on Machine Learning*, pages 4359–4368. PMLR.

Huang, J. and Yau, H.-T. (2020). Dynamics of Deep Neural Networks and Neural Tangent Hierarchy. In *Proceedings of the 37th International Conference on Machine Learning*, pages 4542–4551. PMLR.

Huang, K., Wang, Y., Tao, M., and Zhao, T. (2020). Why Do Deep Residual Networks Generalize Better than Deep Feedforward Networks? — A Neural Tangent Kernel Perspective. In *Advances in Neural Information Processing Systems*, volume 33, pages 2698–2709. Curran Associates, Inc.

Hui, L. and Belkin, M. (2021). Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Ji, Z. and Telgarsky, M. (2018). Gradient descent aligns the layers of deep linear networks. In *International Conference on Learning Representations*.

Ji, Z. and Telgarsky, M. (2020). Directional convergence and alignment in deep learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 17176–17186. Curran Associates, Inc.

Jiang, Y., Krishnan, D., Mobahi, H., and Bengio, S. (2018). Predicting the Generalization Gap in Deep Networks with Margin Distributions. In *International Conference on Learning Representations*.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589.

Karakida, R., Akaho, S., and Amari, S.-i. (2019). Universal Statistics of Fisher Information in Deep Neural Networks: Mean Field Approach. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pages 1032–1041. PMLR.

Kidger, P. and Lyons, T. (2020). Universal Approximation with Deep Narrow Networks. In *Proceedings of Thirty Third Conference on Learning Theory*, pages 2306–2327. PMLR.

Kothapalli, V. (2023). Neural Collapse: A Review on Modelling Principles and Generalization. *Transactions on Machine Learning Research*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Laurent, T. and Brecht, J. (2018). Deep Linear Networks with Arbitrary Loss: All Local Minima Are Global. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2902–2907. PMLR.

Lee, C., Hasegawa, H., and Gao, S. (2022). Complex-Valued Neural Networks: A Comprehensive Survey. *IEEE/CAA Journal of Automatica Sinica*, 9(8):1406–1426.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. (2018). Deep Neural Networks as Gaussian Processes. In *International Conference on Learning Representations*.

Lee, J., Schoenholz, S., Pennington, J., Adlam, B., Xiao, L., Novak, R., and Sohl-Dickstein, J. (2020). Finite Versus Infinite Neural Networks: An Empirical Study. In *Advances in Neural Information Processing Systems*, volume 33, pages 15156–15172. Curran Associates, Inc.

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. (2019). Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Li, D., Ding, T., and Sun, R. (2018). Over-parameterized deep neural networks have no strict local minima for any continuous activations. *arXiv preprint arXiv:1812.11039*.

Li, M., Nica, M., and Roy, D. (2021). The future is log-Gaussian: ResNets and their infinite-depth-and-width limit at initialization. In *Advances in Neural Information Processing Systems*, volume 34, pages 7852–7864. Curran Associates, Inc.

Li, Z., Luo, Y., and Lyu, K. (2020). Towards Resolving the Implicit Bias of Gradient Descent for Matrix Factorization: Greedy Low-Rank Learning. In *International Conference on Learning Representations*.

Liu, C., Zhu, L., and Belkin, M. (2022). Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *Applied and Computational Harmonic Analysis*, 59:85–116.

Liu, F., Liao, Z., and Suykens, J. (2021). Kernel regression in high dimensions: Refined analysis beyond double descent. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 649–657. PMLR.

Lyu, K. and Li, J. (2019). Gradient Descent Maximizes the Margin of Homogeneous Neural Networks. In *International Conference on Learning Representations*.

Matthews, A. G. d. G., Hron, J., Rowland, M., Turner, R. E., and Ghahramani, Z. (2018). Gaussian Process Behaviour in Wide Deep Neural Networks. In *International Conference on Learning Representations*.

Mei, S. and Montanari, A. (2022). The Generalization Error of Random Features Regression: Precise Asymptotics and the Double Descent Curve. *Communications on Pure and Applied Mathematics*, 75(4):667–766.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning.

Nakkiran, P., Kaplun, G., Bansal, Y., Yang, T., Barak, B., and Sutskever, I. (2021). Deep double descent: Where bigger models and more data hurt*. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003.

Nguyen, Q., Mukkamala, M. C., and Hein, M. (2018). On the loss landscape of a class of deep neural networks with no bad local valleys. In *International Conference on Learning Representations*.

Papyan, V., Han, X., and Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663.

Parcollet, T., Morchid, M., and Linarès, G. (2020). A survey of quaternion neural networks. *Artificial Intelligence Review*, 53(4):2957–2982.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library.

Pinkus, A. (1999). Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195.

Poggio, T. and Liao, Q. (2019). Generalization in deep network classifiers trained with the square loss. *Center for Brains, Minds and Machines (CBMM) Memo No*, 112.

Poggio, T. A. and Liao, Q. (2021). Explicit regularization and implicit bias in deep network classifiers trained with the square loss. *arXiv preprint arXiv:2101.00072*.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. (2016). Exponential expressivity in deep neural networks through transient chaos. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Schoenholz, S. S., Gilmer, J., Ganguli, S., and Sohl-Dickstein, J. (2016). Deep Information Propagation. In *International Conference on Learning Representations*.

Seleznova, M. and Kutyniok, G. (2022a). Analyzing Finite Neural Networks: Can We Trust Neural Tangent Kernel Theory? In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference*, pages 868–895. PMLR.

Seleznova, M. and Kutyniok, G. (2022b). Neural Tangent Kernel Beyond the Infinite-Width Limit: Effects of Depth and Initialization. In *Proceedings of the 39th International Conference on Machine Learning*, pages 19522–19560. PMLR.

Seleznova, M., Weitzner, D., Giryes, R., Kutyniok, G., and Chou, H.-H. (2023). Neural (Tangent Kernel) Collapse. In *Advances in Neural Information Processing Systems*, volume 36. Curran Associates, Inc.

Shan, H. and Bordelon, B. (2022). A Theory of Neural Tangent Kernel Alignment and Its Influence on Training.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Soudry, D., Hoffer, E., Nacson, M. S., and Srebro, N. (2018). The Implicit Bias of Gradient Descent on Separable Data. In *International Conference on Learning Representations*.

Sutskever, I., Martens, J., and Hinton, G. E. (2011). Generating text with recurrent neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 1017–1024.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9.

Tirer, T., Bruna, J., and Giryes, R. (2022). Kernel-Based Smoothness Analysis of Residual Networks. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference*, pages 921–954. PMLR.

Valle-Pérez, G. and Louis, A. A. (2020). Generalization bounds for deep learning.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.

Vardi, G. and Shamir, O. (2021). Implicit Regularization in ReLU Networks with the Square Loss. In *Proceedings of Thirty Fourth Conference on Learning Theory*, pages 4224–4258. PMLR.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wang, S., Yu, X., and Perdikaris, P. (2022). When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Willers, O., Sudholt, S., Raafatnia, S., and Abrecht, S. (2020). Safety Concerns and Mitigation Approaches Regarding the Use of Deep Learning in Safety-Critical Perception Tasks. In Casimiro, A., Ortmeier, F., Schoitsch, E., Bitsch, F., and Ferreira, P., editors, *Computer Safety, Reliability, and Security. SAFECOMP 2020 Workshops*, Lecture Notes in Computer Science, pages 336–350, Cham. Springer International Publishing.

Woodworth, B., Gunasekar, S., Lee, J. D., Moroshko, E., Savarese, P., Golan, I., Soudry, D., and Srebro, N. (2020). Kernel and Rich Regimes in Overparametrized Models. In *Proceedings of Thirty Third Conference on Learning Theory*, pages 3635–3673. PMLR.

Wu, Y., Lian, D., Xu, Y., Wu, L., and Chen, E. (2020). Graph convolutional networks with markov random field reasoning for social spammer detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 1054–1061.

Xiao, L., Pennington, J., and Schoenholz, S. (2020). Disentangling Trainability and Generalization in Deep Neural Networks. In *Proceedings of the 37th International Conference on Machine Learning*, pages 10462–10472. PMLR.

Yang, G. (2020a). Scaling Limits of Wide Neural Networks with Weight Sharing: Gaussian Process Behavior, Gradient Independence, and Neural Tangent Kernel Derivation.

Yang, G. (2020b). Tensor programs II: neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*.

Yang, G. and Hu, E. J. (2022). Feature Learning in Infinite-Width Neural Networks.

Yarotsky, D. (2017). Error bounds for approximations with deep ReLU networks. *Neural networks*, 94:103–114.

Yun, C., Krishnan, S., and Mobahi, H. (2020). A unifying view on implicit bias in training linear neural networks. In *International Conference on Learning Representations*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115.

Zhavoronkov, A., Ivanenkov, Y. A., Aliper, A., Veselov, M. S., Aladinskiy, V. A., Aladinskaya, A. V., Terentiev, V. A., Polykovskiy, D. A., Kuznetsov, M. D., Asadulaev, A., Volkov, Y., Zholus, A., Shayakhmetov, R. R., Zhebrak, A., Minaeva, L. I., Zagribelnyy, B. A., Lee, L. H., Soll, R., Madge, D., Xing, L., Guo, T., and Aspuru-Guzik, A. (2019). Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nature Biotechnology*, 37(9):1038–1040.

Versicherung an Eides statt

(gemäß § 8 Abs. 2 Nr. 5 der Promotionsordnung vom 12. Juli 2011)

Hiermit erkläre ich, Mariia Seleznova, an Eides statt, dass die Dissertation mit dem Titel „Analyzing Training Dynamics of Deep Neural Networks: Insights and Limitations of the Neural Tangent Kernel Regime“ von mir selbstständig und ohne unerlaubte Beihilfe angefertigt wurde.

München	27.05.2024	
Ort	Datum	Seleznova, Mariia