Masteroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for matematiske fag

Marius Lindegaard

Feature Learning in the Neural Collapse regime of Deep Classifiers

An investigation into the activation patterns and feature learning in Deep Neural Networks exhibiting neural collapse

Masteroppgave i Industriell Matematikk
Veileder: Benjamin Adric Dunn
Medveileder: Akshay Rangamani, Sigurd Gaukstad
September 2023

NTNU
Kunnskap for en bedre verden

Marius Lindegaard

Feature Learning in the Neural Collapse regime of Deep Classifiers

An investigation into the activation patterns and feature learning in Deep Neural Networks exhibiting neural collapse

Masteroppgave i Industriell Matematikk
Veileder: Benjamin Adric Dunn
Medveileder: Akshay Rangamani, Sigurd Gaukstad
September 2023

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for matematiske fag

NTNU

Kunnskap for en bedre verden

ABSTRACT

We have identified the phenomenon of Neural Collapse in intermediate layers across various DNN architectures, including MLPs, Convnets, and Resnets, for the datasets MNIST, FashionMNIST, SVHN, and CIFAR10. This discovery, previously published in [1] (Rangamani & Lindegaard et. al., ICML 2023), offers insights into DNN behavior and performance. While the layer of collapse somewhat correlates with DNN generalization, it's not straightforward, suggesting an optimal middle-point for generalization. This phenomenon also influences the circuits that can form in networks, with shallower collapse layers necessitating shallower circuits.

Our analysis in revealed distinct activation patterns post-collapse. These patterns in deeper layers show sparsity and alignment with neuron-axes. However, features in pre-collapse layers cover the space without clear separations, exhibiting near symmetry in every principal direction. Despite apparent visual structure emerging through layers when using UMAP for dimensionality reduction, our clustering attempts yielded only partially accurate results, indicating that discernible patterns in pre-collapse layers remain elusive with our current methods.

Lastly, the patterns in intermediate layers could be at odds with prevalent DNN theories. Notably, networks exhibiting intermediate Neural Collapse might not align with the pruned networks of the Lottery Ticket Hypothesis, suggesting a divergence from pure SGD optimization and LTH theory. While our findings align with the superposition and circuits theory, the network with intermediate NC provides a more intricate example than previously analyzed networks in this domain, but significantly simpler than full networks for similar classification tasks. Analyzing these networks present an exciting avenue for future research, potentially allowing for new insights into the optimization and inference in deep neural networks.

SAMMENDRAG

Vi har identifisert Neural Collapse i de indre, skjulte lagene i flere typer dype nevralnett. Vi har vist at fenomenet oppstår i både fullstendig koblede nevralnett, konvolusjonsnett og residual-nett for flere datasett, spesifikt MNIST, FashionMNIST, SVHN, og CIFAR10. Denne oppdagelsen, tidligere publisert i [1] (Rangamani & Lindegaard et. al. 2023), gir innsikt i hvordan dype nevralnett fundamentalt fungerer. I denne oppgaven finner vi at hvor dypt kollapsen først opptrer har betydning for nevralnettets evne til å generalisere. Det er tydelig at det ikke er en klar monoton relasjon mellom de to, og at optimumet opptrer et sted mellom ekstremene. Dette har også implicasjoner for hvilke kretser som kan oppstå i nettverket, der grunnere kollaps krever grunnere kretser.

Vår analyse avdekker distinkte mønstre i aktiveringene post-kollaps (dypere) i netverket. Disse mønstrene er glisne og viser innretting mot nevron-aksene. På den andre siden finner vi at aktiveringene pre-kollaps (grunnere) i netverket er symmetriske og dekker hele rommet uten en tydelig separasjon eller innretting mot nevron-aksene. Vi finner kvalitativ visuall struktur som oppstår gjennom nettverket etter ikke-lineær projeksjon med UMAP. Til tross for dette kan vi kun delvis forklare aktiveringene med klyngeanalyse, og grundigere analyse kreves for å forklare mønstrene som oppstår pre-kollaps.

Til slutt diskuteres at mønstrene i de indre sjulte lagene kan stå som en motsetning til enkelte teorier for funksjonene til dype nevralnett. Vi bemerker særlig at nevrale nett som har indre nevral-kollaps kan fungere som moteksempler til loddhypotesen ("Lottery Ticket Hypotesis") for optimeringsprosedyren til nevrale nett. Resultatene stemmer bedre overens med superposisjon- og krets-hypotesene for nevrale nett. Våre nettverk er et mer komplekst eksempel enn netverk som tidligere er analysert, men er betydlig enklere å analysere med disse metodene enn tilsvarende nettverk. Dette er grunnet den forenklede strukturen som oppstår under indre nevral-kollaps. Observasjonene åpner for fremtidige analyser som kan drive feltet framover, og ved å analysere disse netverkene kan man få dypere innsikt i optimeringen og beregningene som foregår i dype nevralnett generelt.

PREFACE

I would like to thank everyone involved in the process culminating in this thesis, in particular my advisors on the thesis itself: Benjamin Adric Dunn, Akshay Rangamani and Sigurd Gaukstad.

This thesis was written as a continuation of work done through the summer and fall of 2022 at the Center for Brains, Minds and Machines (Poggio Lab) at the McGovern Institute for Brain Research at MIT. I would like to thank all the staff at CBMM, and in particular my collaborators Akshay Rangamani, Tomer Galanti and Tomaso Poggio for the deeply interesting discussions and continuing collaboration. A special thanks to Tomaso Poggio for taking me into the lab, and for the guidance provided by everyone involved. The original material is published at [1], and where this thesis and [1] overlaps I claim no originality in the work this thesis presents. The overlap is mainly section 4.1 of the results and in parts 5.1 of the discussion.

Thank you also to the Neural Data Science Group of the Department of Mathematical Sciences at NTNU. In particular, many thanks Benjamin Dunn for interesting discussions on topics both relevant to and outside the scope of this thesis, and to Sigurd Gaukstad for helping to separate out the good ideas from the muddle. It goes without saying that this thesis would not have looked the same without them.

Lastly, I want to thank my family and fiancée for their endless support and for keeping me grounded through the past year. Your encouragement and offers of support is always appreciated, no matter how far away.

CONTENTS

ABBREVIATIONS

List of all abbreviations in alphabetic order:

- **NTNU** Norwegian University of Science and Technology
- **MIT** Massachusetts Institute of Technology
- **PCA** Principal Component Analysis
- **AI** Artificial Intelligence
- **ANN** Artificial (often deep) Neural Network
- **DNN** Deep (often artificial) Neural Network
- **ML** Machine Learning
- **LLM** Large Language Model
- **MLP** Multi-Layer Perceptron
- **SGD** Stochastic Gradient Descent
- **SOTA** State-of-the-art
- **NC** Neural Collapse
- **NC1-4** Neural Collapse condition 1-4
- **INC** Intermediate Neural Collapse
- **LTH** Lottery Ticket Hypothesis
- **UMAP** Uniform Manifold Approximation and Projection

CHAPTER

ONE

INTRODUCTION

Modern statistics and Machine Learning (ML) methods are the driving forces of the recent advancements in Artificial Intelligence (AI). Recently, we have seen ever more advanced methods, an expanding use of ML-based tools, and widespread use of text-based AI systems. In short, intelligent systems are being used and developed on a scale never seen before. [2] While these models are constructed with precise mathematical formulations, the reasons for emergence of the intelligent behaviour remains a mystery:

> *We fundamentally do not understand how deep learning works.*[3] [4]

This thesis is an investigation into interesting phenomena emerging in Deep Neural Networks (DNNs) during training, contributing to our understanding of the underlying workhorse of AI developments.

In this chapter, I first present motivation for this line of work by connecting it to past methods and theories, recent developments, and future requirements of these systems. Secondly, a description of the project itself is presented, linking it to previous work. Finally, I give a brief overview of the novel contributions presented in this thesis.

1.1 Motivation

This section presents the background and core motivation for the work done in this thesis. Parts of this section, specifically 1.1.1 and 1.1.2, have been based on [5].

1.1.1 A Paradigm shift

A well established challenge in classical statistical learning theory is the bias-variance tradeoff [6]. In this regime, any learning algorithm that regression or classification on data has to strike a balance between the expressivity and the inherit priors of the algorithm in order to perform well. For the final learnt algorithm to perform well, you need to trade off variance for bias to achieve the goals specific to your problem. In most statistical learning courses, this perspective is still a foundational, often repeated, and largely correct picture.

In short, the learning algorithm must be expressive enough that it captures the relevant patterns in the data, while filtering out the irrelevant noise by inducing the right priors and biases. This can often require intimate knowledge of the data domain and of the different algorithms' signal processing behaviour in regards to the data features. If the model has a strong bias towards certain types of features, either by being unable to express others or incorporating priors over the parameter space, they might miss important relevant patterns in the dataset. If, on the other hand, the model is too expressive (high variance), it might pick up random fluctuations in the data and place importance on structures irrelevant to the underlying problem. This will lead the model to generalize less well, and to "false positives" when interpreting the significance of data characteristics.

With this in mind it is clear that it is imperative to choose the right models and priors to filter the noise while keeping the signal. Finding the balance that will allow you to ignore most non-generalizing patterns while keeping the true underlying structure of the data is crucial. It is crucial in designing your model to reflect the structure of the data and perform well on unseen samples.

Or so the classical story goes.

Modern developments in ML and statistical learning have been characterized by increasingly larger models [7]. Despite growing to parameter numbers orders of magnitude higher than the number of datapoints and features per datapoint, they have shown a remarkable capacity for generalization. Following classical theory, the extreme overparametrization and expressivity should make the models overfit the data, capturing all the noise, and thus give horrible generalization. Indeed, we can often see interpolation of the data, but the models generalize to new data despite having classically "overfit" the training data [8]. The contradiction of the empirical findings and the established theory prompts the development of new theory, theory that is applicable to this new paradigm of learning systems.

1.1.2 The Motivation for new Theory

Motivated by the recent developments and incredible achievements of modern ML, we are looking to understand the mechanisms underlying this success. In developing a theory of deep learning one would hope to contribute to both the engineering and science of intelligence.

In the the engineering of intelligence it is paramount to have an understanding of the learning methods in order to improve them. Thus far, a lot of the progress has been driven by an intuitive understanding and application of the inner workings of the algorithms used in ML. This has so far proven instrumental and successful in model scaling [9, 10], understanding and constructing failure modes of the models [11], introducing the relevant biases when needed [12], increasing expressivity in data-rich domains when biases are less relevant [13], and explaining model behavior [14].

Concerning the science of intelligence, the development of relevant understanding is a goal in and of itself. Fully explaining the inner workings of the top performing learning algorithms, be it biological or artificial, would be a major step forwards in this field. Until recently, most artificially constructed learning algorithms were unable to perform on the level of humans in nearly all domains. These algorithms now perform and outperform humans across several modes and metrics in an ever increasing range of problems that were once considered "deeply human" [15] Looking at these artificially constructed learning algorithms, we have an unprecedented opportunity to investigate the specific mechanics of a high-performing "intelligent" system. Furthermore, we may investigate the mechanics of the learning algorithm both during learning and inference down to the smallest deterministic parameter update. Comparing this to the biological systems, we currently have only some grasp on how a brain works on a neuronal level. In fact, we have way to measure or infer the activity of even a piece of a human brain down to the neuronal level. In artificial DNNs, however, we have complete control and measurability of the fundamental dynamics of the learning algorithm. The algorithms are constructed by humans, every state is stored in computer memory, and all data is stored and interacted with deterministically. This unprecedented access to the learning system presents a unique opportunity to make sense of the fundamentals of learning and information processing itself.

Despite this opportunity, no comprihensive theory exists on artificial DNNs. There is no holistic theory for even the simplest nonlinear DNNs, Multi-Layer Perceptrions (MLPs), on the simplest problems such as regression or classification.[3] This is, however, not for lack of effort [16, 17]. There is still a considerable gap between the state of the engineering performance and the science of DNNs.

The contrast becomes even more stark when comparing with the theory for kernel methods. While kernel machines historically were state-of-the-art (SOTA) in many complex problem domains, their development was accompanied by a thorough and deep understanding of their behavior on several layers of abstraction, theory and emergent behavior [18].

In order to develop a similarly strong theory of DNN dynamics, with countless layers of emergent complexity, it is crucial to understand the ability of the models to generalize. As a new paradigm of learning algorithms emerge and the old theory fails to explain the generalizing ability and dynamics of overparametrized models,

a new theory of learning is needed. This thesis aims to push the boundaries of our understanding of DNN dynamics from a bottom-up perspective. We believe significant insights can be gathered by observing the activations, weights, gradients, and learning updates of the DNNs. By building theory from the fundamental inner workings of the network, we wish to establish an *interpretable*, *mechanistic* picture of the model behavior.

1.1.3 Mechanistic Interpretability

In the burgeoning field of Mechanistic Interpretability, researchers are working to understand the fundamental inner workings of DNNs from a bottom-up perspective. The young subfield aims to fully characterise the models behavior by observing the model's activations, weights, and gradients.[4]

Fundamentally, mechanistic interpretability rests on the idea that there are legible, interpretable structures in DNNs that can be reverse-engineered. [19] The subfield differentiates itself from other interpretability research along several axes:

Internal Model Understanding is fundamental in the approach. Where other interpretability research focuses on high-level behavior and often treats the model as a black box, mechanistic interpretability revolves around individual activation patterns and how they interact to form a bottom-up theory. [19]

Specific, actionable insights that allow for detailed model interpretation and modification are a core part of the insights usually generated by this research.[4] As a consequence of the focus on quantitative insights this leads to a higher requirement for specificity and usually present interesting opportunities high-level ideas and modifications. While this kind of concrete, actionable theories also appear in conventional interpretability research (e.g. [11] and [20]), much research into interpretability has not been concerned with the underlying mechanics or presented such opportunities.

A central thesis of much work in this subfield is the concept of circuts introduced in [19] and further developed in [21] [22]. Briefly, the theory of circuts stipulate that one can subdivide the DNN into several abstract computational subgraphs [19]. These subgraphs encode interpretable features and often reflect properties of the underlying dataset. Some more detail is given in section 1.3.2.

This thesis will draw on the same motivation as much other mechanistic interpretability work. Still, the direction presented here is different than much of the work in this subfield of interpretability, and hopes to serve as another path to gaining a fundamental understanding of neural networks.

1.1.4 Interpretability and The Big Picture of Modern ML

The science of interpretability research in ML is in itself inherently interesting, but it can also be instrumental for aspects of modern ML and AI. Specifically, for AI safety, interpretability might play a crucial role in keeping AI systems aligned and robust.

Recently the ever increasing capabilities of machine learning models have translated into an explosion in use cases. Many such systems have become easily available to the masses, such as DALL-E, MidJourney, and recently chat-based

interfaces [1]. While not in the scope of this thesis, the possible economic impact of these systems cannot be understated[23]. This has demonstrated and made poignant the need to be able to steer these systems to behave in a safe manner. Several research labs are spending a lot of resources on safety and steering their systems towards safe behavior, and the topic of AI safety has been on the forefront of much public discourse [24] In fact, one of the major ways to improve the safety and alignment of the chat-based systems, reinforcement learning from human feedback, was crucial to the usability of these systems[25]. In line with this, developing new techniques for aligning these systems to our needs can have an outsized impact on both the system performance and usability.

It is clear from experience in other applied scientific disciplines that understanding the mechanics underlying systems can lead to both new insights, improved capabilities, and more reliable systems. Work in adversarial attacks and examples [11] has already demonstrated failure modes and given improved the training procedure in certain cases. Similarly, new paths contributing to the understanding and alignment of these systems can have a huge impact by providing improved performance, usability, alignment, and safety. Earlier work has explored the possible pathways through which interpretability research can have an impact on improving ML systems. Specifically, reasearch in mechanistic interpretability can help us significantly speed up alignment research [26], train systems to robustly behave well [27], audit and verify systems [26], drive capabilities [28], predict future system capabilities based on e.g. scaling laws and help avoid deceptive systems [26], among other things [29]. [2]

It should come as no surprise that gaining an understanding of how the underlying mechanics work could provide an outsized impact on nearly every aspect of these systems. Opening up new areas of research in this subfield and developing the already established theories could lead to significant developments in both the subfield and in ML as a whole.

1.2 Related work

This section contains the background for the scope of the project, as well as some related directions that do not directly feed into the approach taken in this thesis. Since much background for modern ML has been given in section 1.1, this section will keep the foundational ML references brief. Notably, this thesis builds on previous work conducted by the author while at the Center for Brains, Minds and Machines - Massachusetts Institute of Technology published at the International Conference of Machine Learning 2023 [1], in conjunction with Akshay Rangamani, Tomer Galanti and Tomaso Poggio.

[1] https://openai.com/dall-e-2, https://www.midjourney.com, and https://chat.openai.com/, respectively.

[2] Interestingly, much of the field is driven forwards by university unaffiliated researchers or smaller independent and recently started research labs. The attentive reader might notice that some of the sources given for the future of alignment research, and particularly mechanistic interpretability, are from unconventional organizations of researchers. These ideas have only recently been given more attention, with the popularity among the wider public and genuine worry about the capabilities of the systems.

1.2.1 Background

Major advances in ML, and in particular deep learning, have quickly outpaced what was possible with other learning methods. Initial breakthroughs in computer vision [9, 7, 10] with deep learning has led to an explosion in capabilities in many areas such as natural language processing [30, 25], image generation and multimodality [31], and decision making in diverse environments [32, 33].

With increasing capabilities, the problem of understanding how these networks operate becomes increasingly important (see sec. 1.1). As such, there are several lines of work attempting to understand these systems on a theoretical level. One such example is the neural tangent kernel that aims to explain DNN behavior with tools from kernel methods under the assumption of infinite width of the network, introduced in [34]. Another such line of work aims to explain shallower networks [35]. One key missing component of these approaches is their inability to appropriately model the multiple-level feature learning that is a central part of the learning algorithms of DNNs[36], and what makes them truly "deep".

With the lack of holistic theoretical frameworks for DNNs, much investigation has been done on the dynamics of the models themselves. Some major angles of theoretical research include the information bottleneck method [37], the lottery ticket hypothesis [20], and analysis of the loss landscape in terms of the parameters [38, 39]. Other approaches are more empirical, where conclusions are drawn from the observed patterns and structures in the systems, such as circuts [19, 40], double descent [8, 41], grokking [42], phase changes [43, 44], and particularly neural collapse [45].

1.2.2 Neural Collapse Background

The recent discovery of the phenomenon of Neural Collapse (NC) in [45] is an empirical observation of strong simplifying structures emerging in the final hidden layer of DNN classifiers. In short, the feature of every datapoint of the same class are mapped to the same point representation in this layer, and these points form a strict, maximally separated geometry that is also reflected in the weights (see sec. 2.3).

Several follow-up works have investigated the theoretical basis and implications of this phenomenon, while other works have demonstrated the prevalence of NC in different settings. Of particular interest for this thesis, [46] and [1] have demonstrated that NC extends from the last hidden layer and backwards into to the intermediate layers of some DNNs that exhibit NC. This phenomenon has been coined Intermediate Neural Collapse.

1.3 Core ideas in the field

Some core ideas in the interpretability of DNNs are given in order to give some color to the current state of the field. These ideas are developing, and not necessarily precisely defined. In fact, some of their strength might come from having moldable definitions that develop along with the theory.

1.3.1 Lottery Ticket Hypethesis

The Lottery Ticket Hypothesis first introduced in [20] briefly states that

> A randomly-initialized, dense neural network contains a subnetwork that is initialized such that—when trained in isolation—it can match the test accuracy of the original network after training for at most the same number of iterations.

Intuitively, it states that the first order effect of a DNN is that each randomly initialized network contains a "subnetwork" that already is a somewhat good classifier, and all other subnetworks are optimized away. The training of this subnetwork is a second order effect that, while important, is not the ultimate reason for the incredible performance of DNNs. This hypothesis has shown merit in several experiments and attracted substantial attention [47]. It is clear that in the space of mechanistic interpretability, this idea is idea is one of the key ways to approach explaining the internal mechanisms of DNNs.

1.3.2 Circuts, Features and Superposition

As mentioned, the work pioneered by Chris Olah now at Anthropic seeks to explain the inner workings of DNNs with a somewhat different approach.

The work revolves around three central speculative claims about DNNs (quote from [19]):

Claim 1: Features

> Features are the fundamental unit of neural networks. They correspond to directions.[3] These features can be rigorously studied and understood.

Claim 2: Circuits

> Features are connected by weights, forming circuits. [4] These circuits can also be rigorously studied and understood.

Claim 3: Universality

> Analogous features and circuits form across models and tasks.

At its core, it is an attempt to construct the fundamental building blocks to which one can reduce the behavior of DNNs.

In a similar vein of characterizing circuts, a hypothesis for how DNNs represent features is in "superposition". Introduced in [48], the core of neural networks representing data in superposition is based on assumptions that (a) features are

[3] By "direction" we mean a linear combination of neurons in a layer. You can think of this as a direction vector in the vector space of activations of neurons in a given layer. Often, we find it most helpful to talk about individual neurons, but we'll see that there are some cases where other combinations are a more useful way to analyze networks - especially when neurons are "polysemantic."

[4] A "circuit" is a computational subgraph of a neural network. It consists of a set of features, and the weighted edges that go between them in the original network. Often, we study quite small circuits - say with less than a dozen features but they can also be much larger.

the building blocks of DNNs, (b) a feature is represented by a "direction" in the neuron space and (c) features are sparse. In that case, a DNN can represent more features than there are dimensions in the space, without expecting significant interaction between features.

Of relevance for this thesis is assumption "b", and that the DNN will be expected to utilize the full space to represent as many features as possible. Interested readers are advised to read [48].

1.3.3 The state of interpretability research

Finally, a note on the state of the field of "Mechanistic Interpretability" in its current form: It is not clear what the fundamental properties of "good science" in this field looks like, it is not clear what the fundamental definitions (e.g. of a "feature") are or should be in order to accurately capture the interesting effects of the system, and it is not clear what ideas and paradigms will eventually give rise to productive empirical and quantiatative modes of explaining the phenomena of DNNs we wish to explain [4] [19].

This subfield is in its very early stages, and faces challenges that any burgeoning field faces. One can compare the qualitative work in [49] and [48] to the now established benchmarks of capabilities research for DNNs, and observe the lack of objective measures not only of performance but of valid reasoning.

Some have likened this early stage of an obviously complex and so far qualitative field to the first time humans reached for the microscope and discovered the rich world of cell biology[19]: Our best theories of the world cannot describe the complexity without serious effort and hours spent piecing together an ever fuller picture. Most of our high-level ideas will be wrong, but through empirical evaluation and analysis we will slowly be able to figure out both the low-level mechanisms and the high-level motifs that will allow us to explain what we observe.

The dream is to one day find the DNA and proteins of deep neural networks, allowing for explainability, increased capability and safety of these models.

1.4 Scope and Approach

This thesis bases its work on the results established in [1] and [5], previous work by the main author and collaborators. The work that resulted in [1] was supported by the Poggio lab - McGovern Institute at MIT through the summer and fall of 2022.

The thesis seeks to extend the analysis of neural collapse (see the following chapter for terms and explanations) to the intermediate layers, as published in [1], and then further to investigate whether the layer of intermediate collapse is related to the generalization of the network. Further, it investigates the characteristics of the data transformations before the intermediate neural collapse. The ultimate goal is to characterize the mechanisms by which a single DNN performs classification. This thesis restricts itself to DNNs that has demonstrated intermediate neural collapse, and performs an exploratory analysis focused on the first order characteristics of the data in relation to the network.

1.5 Contributions

The work presented in this thesis is a continuation of results from the summer and fall of 2022 at the Center for Brains, Minds and Machines at McGovern Institute - MIT with Akshay Rangamani, Tomer Galanti and Tomaso Poggio later published in [1]. The results from this paper is included in section 4.1.

In addition to this, we investigate the correlation between the layer of intermediate collapse and generalization. Further, we investigate the activations in the pre- and post-collapse layers. We find that there is significant qualitative and quantitative patters both before and after the collapse. In particular, we investigate nonlinear projections through UMAP, results of clustering the data, as well as PCA and the sparsity of the neurons. We also relate this to relevant theories of DNNs internal functions.

The contributions in this thesis revolve around using these methods to analyze a new domain of neural networks, particularly DNNs with the simplifying structure producing Intermediate Neural Collapse.

CHAPTER

TWO

THEORY

Readers familiar with ML may skim section 2.1 and reference back to it for notation, while readers familiar with background of statistical learning and the new paradigm of overparametrization and generalization may skim section 2.2

2.1 Notation and Problem Specification

This section first outlines the problem specification, inference- and learning method before giving some relevant background on Deep Neural Networks. Note that this section is an adaptation of the background given in [5] by the same author.

2.1.1 Multi-class classification problem

We consider the problem of multi-class classification. Intuitively, the problem consists of mapping an unseen datapoint $\mathbf{x}$ from a probability distribution to its correct class c based on other examples of $(\mathbf{x}, c)$ pairs from the same distribution.

Mathematically, we define an input data space $\mathcal{X} \subseteq \mathbf{R}^d$ and an output class space $\mathcal{C} = \{1, 2, ..., C\}$,[1] with each index representing a different class of the in total C classes. Each datapoint $z = (\mathbf{x}, y)$ consists of features $\mathbf{x} \in \mathcal{X}$ and a corresponding label $y \in \mathcal{C}$. These datapoints z are drawn from a probability distribution $Z = (X \times Y) \in \mathcal{X} \times \mathcal{C}$, noting that X and Y commonly are dependent. For sake of notation, the feature vectors (e.g. $\mathbf{x}$) are assumed to be column vectors. We assume the classes are balanced, so $\Pr_{z=(\mathbf{x},y)\sim Z}[y = c] = \frac{1}{C}$ for all $c \in \mathcal{C}$.

The goal in the problem setup is to minimize the true risk by finding a hypothesis function $h \in \mathcal{H} \subset (\mathcal{X} \to \mathcal{C})$ from a chosen, sufficiently expressive, hypothesis space (i.e. function class) $\mathcal{H}$ such that the true risk is minimized, that is we want to find h^* such that

$$h^* = \arg\min_{h \in \mathcal{H}} \Pr_{z=(\mathbf{x},y)\sim Z}[h(\mathbf{x}) \neq y]. \tag{2.1}$$

Since we usually do not have access to the true distribution Z but instead a set of samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^M$. We use the common technique of assuming the samples are sampled independently from the underlying distribution Z and partition the data into two (differently sized) splits, the training split and the testing split of the data. We then use only the training data to find and estimate of h^* and using the testing data to give an estimate of the true risk. [2]

Mathematically, we denote the training set

$$S = \{(\mathbf{x}_{i,c}, c)\}_{i=1,c=1}^{N,C} \tag{2.2}$$

ensuring that S is a balanced training set of N samples of each class $c \in \mathcal{C}$.

2.1.2 Empirical risk minimization and regularization

First, in order to ease optimization and to give the hypothesis functions the ability to express a level of "uncertainty" in their result, we make our output space one-hot encoded. This is done by redefining the output class space from $\mathcal{C} = \{1, \ldots, C\}$ to $\mathcal{Y} = \mathbb{R}^C$. In this space, a sample $(\mathbf{x}, \mathbf{y}) \in (\mathcal{X}, \mathcal{Y})$ belongs to class c iff $\mathbf{y} = e_c$,

[1] Note the change of notation $\mathcal{C} \to \mathcal{Y}$ in sec. 2.1.2 for later sections.

[2] As will be clear later in the text, we will additionally tune our hyperparameters in model selection on this true risk estimate, which gives an inflated estimate of the accuracy of our final chosen h. While this is highly relevant for accurately describing the true performance of the model, our analysis evaluates the mechanics of the function and only place importance on the performance in relation to the internal mechanics. For this reason, the use of the test-dataset is mainly for choosing models based on hyperparameters, and not to report an unbiased empirical risk estimate.

the unit vector along dimension c in $\mathbb{R}^C$. Note then than the hypothesis space $\mathcal{H} \subset (\mathcal{X} \to \mathcal{Y}) = (\mathbb{R}^d \to \mathbb{R}^C)$. A common way to specify the estimated class from the now vector-valuated $h(\mathbf{x}) = \mathbf{y}$ is to use $c = \arg\max_{c' \in \mathcal{C}} \langle e_{c'}, h(\mathbf{x}) \rangle$, simply the highest valuated class entry of the output vector.

Assuming a choice of an appropriate hypothesis space $\mathcal{H}$, how does one choose an appropriate metric for evaluating estimates $\tilde{h}$ of the optimal h^*? A common way is to employ Empirical Risk Minimization (ERM), minimizing the risk over the training set S, by letting

$$\tilde{h} = \arg\min_{h \in \mathcal{H}} \frac{1}{NC} \sum_{(\mathbf{x},y) \in S} \ell(h(\mathbf{x}), y). \tag{2.3}$$

where ℓ is some convex loss function giving a notion of how "wrong" an output is.

Common choices of ℓ in the multi-classification problem include the cross-entropy loss [6] and the square loss [50]. In this project we mainly use the square loss as this interfaces nicely with theory on the subject. In $\mathbb{R}^C$, the square loss is defined as

$$\ell(\tilde{\mathbf{y}}, \mathbf{y}) = \frac{1}{2} \|\tilde{\mathbf{y}} - \mathbf{y}\|_2^2. \tag{2.4}$$

In addition to finding the ERM model, it is common to have some prior distribution over $\mathcal{H}$ which biases the choice of $\tilde{h}$ towards simpler functions.[6] For example, in linear regression it is common to employ a prior that is a symmetric multivariate normal distribution around the origin on all the parameters of the model.[51] This intuitively amounts to biasing towards simpler functions, and mathematically amouts to adding a penalizing regularization term $R(h_\theta) = \frac{1}{2}\lambda\|\theta\|_2^2$ to the ERM minimizer when finding $\tilde{h}$. Here, λ is a parameter defining the variance of the symmetric multivariate normal distribution, and θ are the parameters of h.

With the regularizer and loss function, we arrive at the common optimization goal of minimizing the regularized empirical risk:

$$\begin{aligned} \tilde{h} &= \arg\min_{h \in \mathcal{H}} \frac{1}{NC} \sum_{(\mathbf{x},y) \in S} \ell(h(\mathbf{x}), y) + R(h) \\ &= \arg\min_{h \in \mathcal{H}} \left(J(h; S) + R(h) \right) \\ &= \arg\min_{h \in \mathcal{H}} \ \tilde{J}(h; S) \end{aligned} \tag{2.5}$$

naming $\tilde{J}$ the total loss.

2.1.3 Deep Neural Networks

In this project, we use various versions of DNNs as our hypothesis space $\mathcal{H}$. This function class consists of multiple compositions of parametrized linear transformations with nonlinear transformations in between. This generally gives rise to a very rich hypothesis space, depending on the specific composed transformations, typically called the *arcitechture*.

In its simpest form, a deep neural network is a composition of linear transformations $\mathbf{W} = W^0, \ldots, W^L$ with some nonlinear function σ between them:

$$h(\mathbf{x}) = W^L \sigma(\ \ldots\ \sigma(W^1 \sigma(W^0 \mathbf{x}))\ \ldots) \tag{2.6}$$

Note that the nonlinearity is necessary to ensure that the function h is not linear in its entirety.

In our experiments, the following specifications of the DNN is used:

Bias term: A bias term is sometimes added to the linear transformations W^l so that $W^l\mathbf{x}^l \to W^l\mathbf{x}^l + \mathbf{b}^l$.

ReLU Activation: The nonlinearity σ is an element-wise Rectified Linear Unit (ReLU)[52], so $\sigma(\mathbf{x})_i = \max\{0, x_i\}$. Using this simple piecewise linear activation permits theoretical analysis such as [53] and [54].

Convolutional layer: Several of the architechtures employ convolutional layers, Convolutional Neural Networks (CNNs). These layers are a subset of of the "fully connected" linear layers, with most elements in the matrix set to 0. Intuitively, this layer convolves the C_{in}-channel $H \times W$-size image with a smaller *filter* matrix of size $(C_{\text{out}}, C_{\text{in}}, k, k)$ where $k << H, W$. This gives rise to local interactions in "image space". Restricting models to these local interactions is a way of implementing the intuitive bias that pixel values in an image *relative to its neighbors* is more important than relative to all other pixels in the image.[55] [3]

MaxPool: Some convolutional networks also employ *max pooling*. This layer is simalar to the convulutional layer but is nonlinear in nature, has no trainable parameters, and is used for *downsampling* the image. In essence, a max pooling layer takes as input an image, looks at non-overlapping subregions of it, and ouputs the maximum value of each subregion. In our implementations, these subregions are 2×2 pixels, and the MaxPool mapping is thus $(C_{\text{in}}, H, W) \to (C_{\text{in}}, H/2, W/2)$. [4]

Batch Normalization: In some cases, a *Batch Normalization* (BN) layer [56] might be appended to the linear transformation, $W^l\mathbf{x}^l \to \text{BN}(W^l\mathbf{x}^l)$. This is done to reduce the internal covariate shift significantly speeding up and easing model optimization [56]. This layer acts to return the distribution of activations to a β-mean, γ-covariance distribution in expectation over the dataset where β and γ are trainable parameters. The transformation can be written

$$\text{BN}(\mathbf{x}) = \frac{\mathbf{x} - \mathbb{E}\left[\mathbf{x}\right]}{\sqrt{\text{Var}[\mathbf{x}] + \epsilon}} * \gamma + \beta \tag{2.7}$$

noting that all operations are element-wise (expectation, variance and multiplication included). [5] The small $\epsilon = 10^{-5}$ is added for stability.

Residual connections: In some architechture classes (see sec. 3), there are *residual connections* or *skip connections*. In short, this residual connection adds the values of the nodes in a previous layer to the current layer in the DNN function composition. Mathematically, adding a skip connection from layer l to layer $l + 2$

[3] See https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html and code implementation for further details.

[4] See https://pytorch.org/docs/stable/generated/torch.nn.MaxPool2d.html and code implementation for further details.

[5] See https://pytorch.org/docs/stable/generated/torch.nn.BatchNorm1d.html and code implementation for further details.

transforms

$$\begin{aligned} \mathbf{x}^{l+2} &= \sigma(W^{l+1}\mathbf{x}^{l+1}) = \sigma(W^{l+1}\sigma(W^{l}\mathbf{x}^{l})) \\ &\downarrow \\ \mathbf{x}^{l+2} &= \mathbf{x}^{l} + \sigma(W^{l+1}\sigma(W^{l}\mathbf{x}^{l})) \end{aligned} \tag{2.8}$$

where x^l is the activation values in layer l. The use of residual connections have been shown to ease optimization, especially in very deep DNNs.[10]

Using these layers to construct different function spaces $\mathcal{H}$ in which to optimize (regularized) ERM. While convolutional layers and max pooling layers are used to imprint an explicit bias towards pixel-wise local interactions in image space, the other amendments are used to ease optimization. The focus on successful optimization is necessary because, as detailed in 2.2 and 3.2, these models can become very large and contain many orders of magnitude more parameters than to classical approaches.

In total, these DNNs will implement a function parametrized by the set of variables θ, $h_\theta \in \mathcal{H}$, as detailed in the previous section. The selection of the architecture determines the possible function space $\mathcal{H}$, but θ has to be optimized. In general, using sufficiently large models, the function space $\mathcal{H}$ is very rich and can fit nearly any function to arbitrary precision [35].

2.1.4 Model training and evaluation

Due to the extremely complex function space $\mathcal{H}$ reperesentable by DNNs and the highly nonlinear behavior of the function h_θ with respect to its parameters θ, it is in general not possible to find an explicit formula for a global optimum for any interesting loss function. [6]

In order to minimize the regularized ERM (eq. 2.5), we employ variations of Gradient Descent in the parameter space θ, *training* the model with gradual parameter updates.

As detailed in sec. 2.1.2 and eq. 2.4 we use an MSE loss function ℓ. Note results showing that the choice of MSE loss over cross-entropy loss empirically does not significantly impact the emergence of neural collapse[57], but lends itself easier to theoretical investigations.[50] We also use a symmetric multivariate normal distribution around the origin as a prior on all of the parameters θ, giving the regularization term $R(h_\theta) = \frac{1}{2}\lambda\|\theta\|_2^2$ for some hyperparameter scalar $\lambda > 0$ inversely proportional to the multivariate normal prior.

We use a time-descrete optimization procedure for the regularized ERM minimization, as opposed to attempting continuous time GD. For each step this gives an update rule [6]

$$\theta^{(t+1)} = \theta^{(t)} + \gamma\nabla_\theta \tilde{J}(h_{\theta^{(t)}}; S) \tag{2.9}$$

$$= (1-\gamma\lambda)\theta^{(t)} + \gamma\nabla_\theta J(h_\theta^{(t)}; S) \tag{2.10}$$

where $\tilde{J}(h, S) = J(h, S) + R(h)$ as detailed in sec. 2.1.2. In this form it is visible how λ works to "decay" the weights in each step, giving it the colloquial name *weight decay*.

For our experiments we use Stochastic Gradient Descent (SGD), using only a *batch* of the data S at any given time to perform the optimization. In addition to

making the optimization procedure tractable on modern hardware, this introduces noise to the optimization procedure helping avoid local minima.[58] Further details are given in sec. 3.3.

2.1.4.1 Model Evaluation

While we are optimizing the regularized ERM (eq. 2.5), the goal of the optimization is to minimize the true risk (eq. 2.1). As an estimate of the true risk, we use the classification accuracy on the held-out part of the sampled data, the test data, and evaluate the model using this accuracy.

2.2 The Two Regimes

2.2.1 The classical regime and bias/variance tradeoff

Classical theory of statistical learning dictates that there is a tradeoff between model expressivity and generalization. [6]

Model bias are assumptions in the learning algorithm, e.g. a restricted function class $\mathcal{H}$ (linear models, feature selection, convolutional DNNs) or a regularizer explicit (see sec. 2.1.2 and 2.1.4) or implicit ([53] [46]) in the optimizer. This gives worse performance on the ERM, but is an important way to ensure generalization by hopefully picking up the true signal in the data while regularizing away the noise. With higher model complexity comes higher sensitivity to small fluctuations in the training set, resulting in the model fitting random noise in the dataset (*overfitting*). This, in theory, leads to worse generalization since the noise does not generalize. Figure 2.2.1 illustrates the tradeoff and contribution to error in this classical domain.

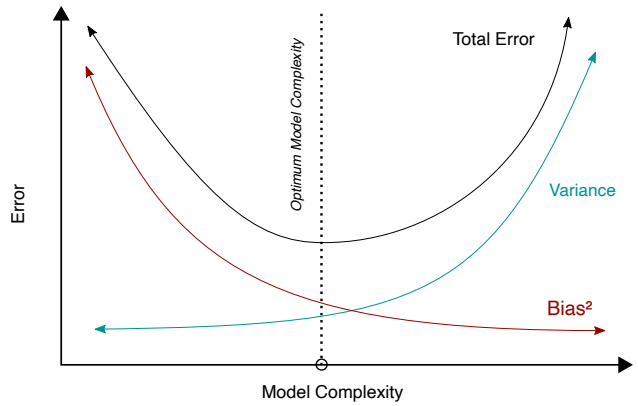


Figure 2.2.1: Classical picture of testing error due to bias and variance as a function of model expressivity. Note the optimal model complexity point where the balance between model expressivity (variance) and generalization (bias, model priors) produces the lowest generalization error.

The underlying idea is that we want to fit the signal in the data and not the noise. Restricting model complexity (low variance) and penalizing fitting what is suspected to be "noise" (through regularizers/explicit bias) - while allowing

sufficient expressivity to fit the major patterns - has been the traditional way to extract the signal in the data.

In the past years, with the advent of models with parameter numbers nearing the trillions [25], is has become clear that this idea is insufficient in explaining the performance of modern techniques.

2.2.2 The modern regime and overparametrization

2.2.2.1 Overparametrization of DNNs

Recent work in the closely related field of Machine Learning (ML), with a stronger focus on prediction accuracy than mathematical explainability, have demonstrated that the over-parametrized and highly expressive DNNs (when compared to the data and traditional methods) are able to generalize well. [7] Investigation into this phenomenon has been done in data-centric [59], optimization focused [16] [60] [61], and simplified [62] [17] settings. Despite this theoretical focus on the phenomenon, no cohesive theory exists for the empirical phenomenon.

Demonstrating the degree of their overparametrization, these models giving state-of-the-art prediction accuracy can also fit the data with 100% label noise.[63]

Further developments in Vision Transformers (ViTs) [13] shows an interesting similar trend. In short, the ViT shapes input format of the visual data allowing usage of the attention mechanism intruduced in [30] for language models. The ViT architecture disposes of the strong CNN bias towards local pixel-wise interactions by allowing arbitrary, learnt, interactions between different parts of the image. This property makes these models even more expressive than CNNs, and have demonstrated higher accuracy than CNNs [64] when enough data is available to learn these interactions [65].

2.2.2.2 Double descent and the terminal phase of training

Recent findings of the double descent in DNNs rigurously establish the break with the classical regime.[8] In essence, there are two regimes of test error behavior as a function of model complexity. In the previously described classical regime, there is a bias-cariance tradeoff, but if model complexity (i.e. variance) is increased further we eventually enter the *modern regime*. In the modern regime, bigger is better. Higher model complexity will result in better generalization and test accuracy. (See figure 2.2.2)

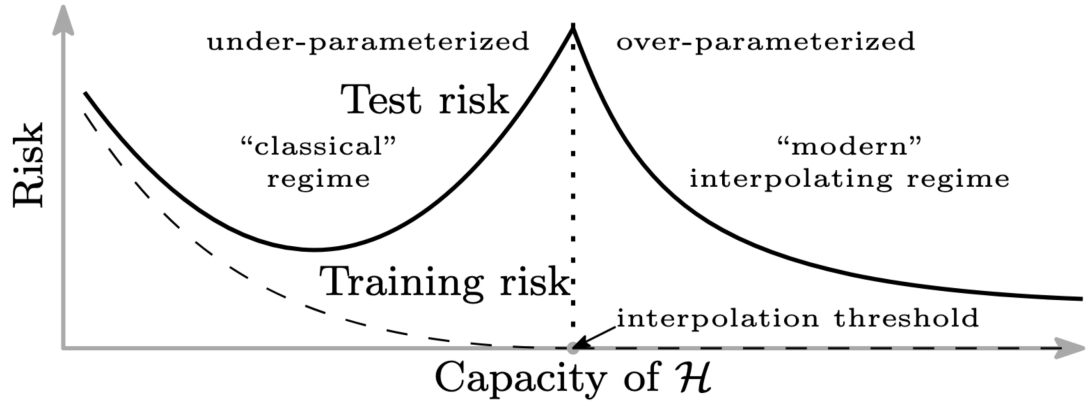


Figure 2.2.2: Qualitative description of the break with the classical regime for overparametrized function classes $\mathcal{H}$ as shown in [41]. The misclassification risk attains a minimum in the classical regime, corresponding to figure 2.2.1, but then decreases with increasing function class capacity in the modern regime after interpolation. DNNs of modern scale typically occupy the modern regime in their capacity.

The phenomenon is prevalent in both classical techniques (e.g. decision trees) and DNNs [41] and was observed already in [66] and [67]. Despite this, double descent is still not well understood in DNNs.

The double descent phenomenon appears not only in the expressiveness of the function space $\mathcal{H}$, but also for other regularization techniques such as explicit regularizers and early stopping. [8] With this in mind, [45] coins the *terminal phase of training* (TPT) as the training steps past the (badly generalizing) interpolation threshold.

Describing the properties emergent during the double descent in DNNs (in the TPT or outside) could prove greatly beneficial to our understanding of modern machine learning, and the scaling of future machine learning models.

2.3 Neural Collapse

As a step in understanding the overparametrized regime, investigations have been made into the emergent structures in models during the TPT.

The phenomenon of Neural Collapse (NC) originally described in [45] is an empirical observation of the activations and weight matrix in the final fully connected hidden layer of DNN classifiers. The NC structure arises when the model interpolates the data in the TPT continuing training with weight regularization. We give the four core definitions of NC, provide some intuition for the phenomenon, and describe some of the preliminairy theoretical explanation of the phenomenon.

2.3.1 Intuition of Neural Collapse

In order to provide intuition for the NC phenomenon, we first define the Simplex ETF with notation as in [45], following the definitions from [68]:

Definition 1 ***Simplex ETF***

A simplex Equiangular Tight Frame (ETF) is a set of C vectors forming the vertices of a $C-1$ dimensional simplex around the origin. Interpreted at the columns of a matrix $M^* \in \mathrm{R}^{C\times C}$, we have a standard *simplex ETF*

$$M^* = \sqrt{\frac{C}{C-1}}\left(\mathbf{I_C} - \frac{1}{C}\mathbf{1_C}\mathbf{1_C}^\top\right) \tag{2.11}$$

where I_C is the C-dimensional identity matrix and $\mathbf{1}_C$ the C-element one-vector. For our purposes we assume a general simplex ETF

$$M = \alpha U M^* \in \mathbb{R}^{d\times C}, \qquad C \leq d \in \mathbb{N} \tag{2.12}$$

where $\alpha \in \mathbb{R}_+$ is a scaling factor and $U \in \mathbb{R}^{d\times C}$ is a partial orthogonal matrix such that $U^\top U = I_C$.

Assuming the general simplex ETF allows for arbitrary scalings and rotations of the simplex.

We now consider the properties of NC in the final hidden layer in order:

NC1 (Variability Collapse) states that the variance in the activations in the final hidden layer go to zero for all datapoints in the same class. Essentially, the activations become entirely determined by the class to which the datapoint belong to, converging to the means of each class.

NC2 (Convergence to Simplex ETF) states that the class means (to which the datapoints converge) for a particular structure, a (general) simplex ETF, when centering by the global mean. Intuitively, the C global-mean centered class means converge to form a structure with maximal angular separation and equal vector magnitude. The points with maximal pairwise angular separation is the vertices of the simplex, a generalization of the tetrahedron for higher dimensions. [68]

NC3 (Convergence to self-duality) states that the linear classifiers comprising the rows of the weight matrix converge to the class means, up to rescaling. This also ensures the weights form a simplex ETF, with each classifying row vector corresponding to the relevant class mean, and symmetry between each of the other class means for a single class mean.

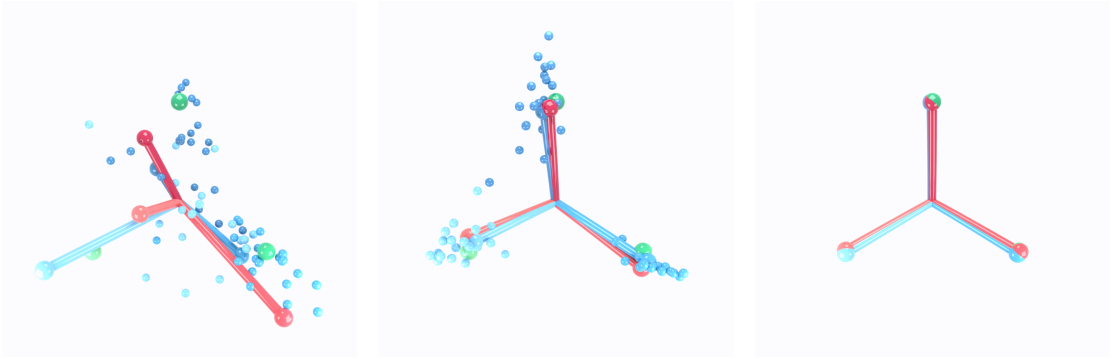
NC4 (Simplification to NCC classifier) states that the classification rule of the network corresponds perfectly to choosing the closest class mean in the last hidden layer, simplifying classification.

The convergence to these NC1-4 properties imply a simple structure in the final hidden layer activations. Through rigorous mathematical definitions we can observe the convergence to these properties occur in experiments.

2.3.2 Definitions of Neural Collapse

Formally, we define the four NC conditions as follows:

Closely following the definitions in [45], for sample i of class c we define the activations in the last hidden layer as $h_{i,c} \in \mathbb{R}^P$ after the activation. The linear classifier from the last hidden layer to the output layer consists of vectors $w_{c'} \in \mathbb{R}^P$ for each class $c' \in 1, ..., C$ so the output activation for class c' for sample i of class c is $\langle h_{i,c}, w_{c'}\rangle$. Then, using the global mean $\mu_G = \frac{1}{NC}\sum_{i,c} h_{i,c}$, we have relative feature class-means $\mu_c = \mu'_c - \mu_G = \frac{1}{N}\sum_i h_{i,c} - \mu_G$.



(a) Early epochs **(b)** Middle epochs **(c)** Late epochs

Figure 2.3.1: Progression of embedded representations showing NC in last hidden layer through training (right to left), adapted from [45]. The plot shows a projection into 2D of the representations in the last hidden layer of a VGG13 [7] on CIFAR10. The blue balls are sample representations, with class mean represented by the ball-on-stick. The red ball-on-sticks are linear classifier vectors (normalized). Different classes are represented with different color shadings. Green balls represent the Simplex ETF structure, see eq. 1. Note the noisy representations in early epochs, linear separability in middle epochs, and the convergence towards NC in later epochs. Detailed description is omitted as this plot serves to form intuition for the phenomenon, and is not part of the core results. Full animation can be found at https://purl.stanford.edu/br193mh4244.

Neural collapse is then characterized by the following properties of convergence over increasing number of training epochs. We have

(NC1) Within-class variabillity collapse:

$$\Sigma_W = \frac{1}{NC}\sum_{i,c}(h_{i,c} - \mu_c)(h_{i,c} - \mu_c)^T \to \mathbf{0} \tag{2.13}$$

(NC2) Convergence to simplex ETF:

$$\frac{\langle \mu_c, \mu_{c'} \rangle}{\|\mu_c\|_2 \|\mu_{c'}\|_2} \to \begin{cases} 1 & c = c' \\ -\frac{1}{C-1} & c \neq c' \end{cases}, \tag{2.14}$$

and

$$\|\mu_c\|_2 - \|\mu_{c'}\|_2 \to 0 \tag{2.15}$$

(NC3) Convergence to self-duality:

$$\frac{w_c}{\|w_c\|_2} - \frac{\mu_c}{\|\mu_c\|_2} \to \mathbf{0} \tag{2.16}$$

(NC4) Simplification to nearest class center classification:

$$\arg\max_c \langle w_c, h \rangle \to \arg\min_c \|h - \mu_c\|_2 \tag{2.17}$$

These four properties characterize the phenonenon of NC in the last hidden layer before a linear classification layer. As most neural networks contain some stochasticity in both initialization and training, the properties described are limits towards which the measures of NC converge.

2.3.3 Metrics of Neural Collapse

In order to measure the convergence to the afformentioned NC1-4 properties, it is often easier to look at related measures that better fit the dynamics of the system. For example, it is easier to evaluate a single scalar aggregate than visualizing a whole vector or matrix converging to 0.

The following measures are used in practice to identify NC:

NC1 is measured with

$$\texttt{Tr}[\Sigma_B^\dagger \Sigma_W]/C \to 0 \tag{2.18}$$

where $\Sigma_B^\dagger$ is the Moore-Penrose inverse of the between-class-means covariance matrix defined as

$$\Sigma_B = \frac{1}{C}\sum_c \mu_c \mu_c^T.$$

noting that $\mu_c = \mu_c' - \mu_G$ is the globally centered class mean for class c. This measure ensures normalization of the within-class covariance by the between-class covariance, essentially scaling the original NC1-definition to the correct scale of the total data variance. Using the trace of the resulting matrix also ensures a scalar value, and the positive semi-definite nature of $\Sigma_B^\dagger \Sigma_W$ ensures that $\Sigma_B^\dagger \Sigma_W \to \mathbf{0} \Leftrightarrow \texttt{Tr}[\Sigma_B^\dagger \Sigma_W]/C \to 0$.

NC2 is measured with three measures for the simplex ETF property. Let $\cos_\mu(c, c') = \langle \mu_c, \mu_{c'} \rangle / (\|\mu_c\|_2 \|\mu_{c'}\|_2)$ be the cosine angle between two (globally centered) class means. We then measure the average cosine angle between the different relative class means, as well as the standard deviation:

$$\operatorname*{Avg}_{c,c' \neq c} \cos_\mu(c, c') + \frac{1}{C-1} \to 0 \tag{2.19}$$

$$\operatorname*{Std}_{c,c' \neq c} \cos_\mu(c, c') \to 0. \tag{2.20}$$

To determine equal length of the relative class means we measure

$$\frac{\operatorname{Std}_c \|\mu_c\|}{\operatorname{Avg}_c \|\mu_c\|} \to 0. \tag{2.21}$$

Together these conditions form necessary and sufficient conditions for convergence of the class means to a simplex ETF. Note that equivalent measures could be used for the rows for the weights as well, but are for brevity left out since NC3 ensures equivalence between the two.

NC3 is measured by the convergence of the convergence of the weight matrix to the dual of the class means, up to rescaling. As presented in [45], a way to measure this is by observing the Frobenius norm of the normalized difference of the matrices,

$$\|\widetilde{\mathbf{W}}^\top - \widetilde{\mathbf{M}}\|_F \to 0 \tag{2.22}$$

where $\widetilde{\mathbf{W}} = \mathbf{W}/\|\mathbf{W}\|_F$ and $\widetilde{\mathbf{M}} = \mathbf{M}/\|\mathbf{M}\|_F$ letting $\mathbf{M} = [\mu_1\ \mu_2\ ...\ \mu_C] \in \mathcal{R}^{d \times C}$ be the matrix of centered class means in its columns.

NC4 is the condition that the classifying rule goes to nearest-class-center classification, which can be observed by checking the mismatch between the NCC classification and the network classification. We essentially observe the rate of disagreement between the networks output $\arg\max_c \langle w_c, h \rangle$ and the NCC classification $\arg\min_c \|h - \mu_c\|$ for a given activation h in the last hidden layer.

By observing these measurements for the properties NC1-4, we can ascertain the NC phenomenon in the last hidden layer.

2.3.4 Intermediate Layer Collapse

Establishing NC in the final hidden layer naturally leads to the question of whether this phenomenon exists in other parts of the network as well. In recent work, [46] describes NC4 and a version of NC1 in intermediate layers of simpler CNNs. In addition to establishing the existence of NC1 and NC4 structures in intermediate layers in the networks, it establishes a connection between the depth at which the collapse happens and the difficulty of the dataset. In short, for more difficult datasets the collapse happens later in the network, corresponding to a more complex function being implemented. Additionally, there is very little correlation between the number of layers in the DNN and when the collapse happens. The collapse is dataset-dependent, but network depth-independent. This phenomenon is coined *depth-regularization*.

2.3.4.1 Extending measures to intermediate layers

Since NC2 and NC4 is defined exclusively in the activation space and easily computable, these can be measured in the intermediate layers without significant modifications to the measurements.

NC3, which relates the activations to the weights, has to be modified to accommodate the weights' no longer being a rank-C matrix, but of rank possibly as high as the output channels.

In order to conserve the intuition of NC3 in intermediate layers, we want to establish the "convergence of the weight matrix to the dual of the class means". Since the classification weights had an exact correspondence to each class, mapping them to their respective outputs, this is simple in the final hidden layer. For intermediate layers, however, we can establish this connection by:

1. The weight matrix is of rank C.
2. The C most significant (input) singular vectors of the weight matrix occupy the same subspace as that spanned by the class means.
3. The to C singular vectors are balanced between the class means.

In order to measure this phenomenon we should observe that

1. The first C singular values are significantly separated from the rest.
2. The Principal Angle Between Subspaces (PABS) [69] between (a) the space spanned by top C input singular vectors of the weight matrix and (b) the class means is small.

3. The first C singular vectors have approximately the same associated singular value.

This definition generalizes the definition of NC3 in the final hidden layer to intermediate layers.

Mathematically, for a fully connected linear layer multiplication $W^l\mathbf{x}^l$ we compute the SVD $W^l = U\Sigma V^T$ where $\Sigma = \text{diag}[\sigma_1, \sigma_2, ...,]$ is a diagonal matrix of all the singular values of W^l. We then calculate the relative singular values

$$\tilde{\sigma_i} = \frac{\sigma_i}{\sum_i \sigma_i} \tag{2.23}$$

and should observe that

$$\tilde{\sigma_i} = \begin{cases} \frac{1}{C} & \text{for } i \leq C \\ 0 & \text{for } C < i \end{cases} \tag{2.24}$$

to uphold property 1 and 3.

For property 2, measuring the PABS, we use the first C rows of V^T, V_C^T, and wish to find the overlap of span$\{V_C\}$ and span$\{M\}$ where $M = [\mu_1\ \mu_2\ ...\ \mu_C] \in \mathcal{R}^{d\times C}$ again is the matrix of class mean column vectors stacked column-wise. The cosine angle between these subspaces is given by the singular values of $V_C^T V_M = U'\Sigma' V'^T$, $\Sigma' = \text{diag}[\sigma'_0, \ldots, \sigma'_C]$, so

$$\text{AvCosPABS}(V_C, M) = \frac{1}{C}\sum_{i=1}^{C} \cos(\theta_i) \tag{2.25}$$

$$= \frac{1}{C}\sum_{i=1}^{C} \sigma'_i. \tag{2.26}$$

and for NC3 to hold we expect $\text{AvCosPABS}(V_C, M) \to 1$.

For the convolutional layers we treat each channel as an individual feature, similar to a single neuron in the fully connected network. With this interpretation we first flatten the features and filter weights along the dimensions which are not the input channels, and then perform the same procedure on the resulting weight and feature matrices.

With these tools in hand we may evaluate the NC conditions throughout the layers of different DNNs.

CHAPTER

THREE

METHODS

3.1 Datasets

We use several commonly used image classification datasets, all with 10 classes each:

- MNIST [70]
- FashionMNIST [71]
- SVHN [72]
- CIFAR10 [73]

See Appendix A.2 for further details.

3.2 Neural Netork Models

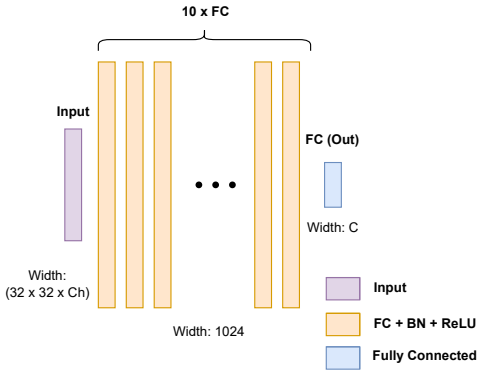
The following DNN architechtures are used[1]:

MLP-L: A Multi-Layer Perceptrion (MLP) with L fully connected 1024-width layers with Batch-Normalization and a ReLU activation function. See figure 3.2.1a

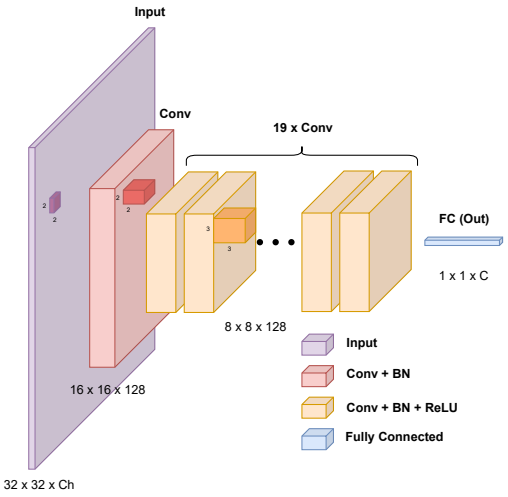
ConvNet-L: A convolutional network of L layers of 128 channels each and a final fully connected layer for classification. The first two convolutions use a (2×2) kernel and stepsize 2 to downsample the image. All convolutional layers use Batch Normalization, and all but the first downsampling use a ReLU activation function. See figure 3.2.1b and footnote [3] in sec. 2.1.3 for further details.

ResNet$L$: A L-layer ResNet [10] architecture employing convolutional layers, residual connections, Max-Pool layers and Batch Normalization. Also contains a final fully connected layer for classification.

[1]Note that while these types of architechtures are commonly used, only **ResNet-L** is standard notation for the corresponding network. **MLP-L** and **ConvNet-L** is adapted from [46].



(a) MLP-10 arcitechture with a $(32 \times 32 \times$ Ch) input as a class C classifier. Each vertical bar represents a layer. The bottom right labels describe the contents of each layer. the majority of the network consists of fully connected (FC) layers with Batch Normalization and ReLU activation.



(b) ConvNet-20 arcitechture with a $(32 \times 32 \times$ Ch) input as a class C classifier. Each vertical bar represents a layer. The bottom right labels describe the contents of each layer, and the height, width and channel depth is displayed below each layer. The majority of the network consists of convolutional layers layers with Batch Normalization and ReLU activation. Note the small darker prisms within the larger layers signifying the kernel size of the following layer.

Figure 3.2.1: Custom DNN Architectures

3.3 Training procedure

The models are trained following standard practice with Stochastic Gradient Descent (SGD), mini-batching, learning rate decay and weight decay.

The training is performed by running through the whole training dataset in one *epoch* by splitting it into random mini-batches shuffled randomly for each epoch, performing SGD. We use mini-batches of 128 at each optimization step. Each experiment runs for 350 epochs, doing learning rate decay by a factor 0.1 at epoch 100 and 200. We use a weight decay parameter $\lambda = 5 \cdot 10^{-5}$ for the ResNets and $\lambda = 2 \cdot 10^{-3}$ for the MLPs and ConvNets.

As choosing the correct learning rate is crucial to achieving good performance, we perform a hyperparameter search over the learning rate for each of the models and datasets. We run training with 11 learning rates from 0.0001 to 0.2 (inclusive) with approximately logarithmic spacing. We then choose the model with the best accuracy on the test dataset, evaluating the model as specified in sec. 2.1.4.1, and perform our analysis on that model.

3.4 Clustering and dimensionality reduction methods

To visualize and cluster the datapoints we respectively utilize UMAP and, after experimenting with several methods[2], agglomerative clustering with complete linkage.

This section will not give a complete treatment of the two methods, but rather give the core ideas of the methods that is relevant for this thesis. For a more complete treatment, see the publications referenced in the two sections.

We additionally note that the cosine distance is given as

$$\textit{cosine-distance}(u,v) = 1 - S_C(u,v) = 1 - \frac{u \cdot v}{\|u\|_2 \|v\|_2} = 1 - \cos(\theta) \in [0,2] \quad (3.1)$$

where u and v are two vectors and θ is the angle between them.

3.4.1 Uniform Manifold Approximation and Projection - UMAP

UMAP is a dimensionality reduction algorithm first introduced in [74], where it demonstrated it's effectiveness in visualizing high-dimensional datasets.

At its core, UMAP operates based preserving local neighbors. Specifically, it constructs a high-dimensional graph representation of the data and then optimizes a low-dimensional counterpart to be as structurally similar as possible. This is achieved by building a "fuzzy simplicial complex", a weighted graph where edge weights signify the probability that two points are connected. The connection is determined by extending a radius from each data point and connecting points when these radii overlap. The choice of this radius is determined locally based on the distance to each point's n'th nearest neighbor. The choice of how many neighbors, n, to consider is a hyperparameter that when increased increases the global accuracy of the transform, at the cost of higher runtime.

UMAP shares similarities with t-SNE, but is preferred in this thesis due to the size of the datasets used. UMAP scales significantly better with the number of datapoints used, and has arguably qualitatively better preserving projections than t-SNE [74].

Readers are encouraged to explore the original publication [74] or interactive library of visualizations and comparisons at `https://pair-code.github.io/understanding-umap/`. The library for running UMAP has documentation available at `https://umap-learn.readthedocs.io/en/latest/`.

In this thesis, all UMAP-clusterings are using the cosine-similarity metric.

[2]Other attempted clustering methods include: K-means, affinity propagation, birch-clustering, DBSCAN, OPTICS, spectral clustering, mean-shift clustering and gaussian mixture models. A limited hyperparameter search was conducted, with a mixture of data-augmentations such as using cosine similarity/euclidean distance metric, mean-centering (where applicable), normalization to the 1-sphere and dimensionality reduction by PCA. Agglomerative clustering was chosen based on ability to cluster the data in clusters of appropriate sizes, stability over sets of hyperparameters, runtime, and most importantly the "accuracy" (based on class labels) of the clusters.

3.4.2 Agglomerative clustering

Agglomerative clustering is a bottom-up hierarchical clustering method. Through experimentation, it was found that the "complete" linkage method gave the qualitatively best results, but the specific choice of agglomerative clustering method gave only minor variations.

Intuitively, the method builds a tree-structure (a dendrogram) from the bottom-up. It first considers all points as it's own cluster. Then, at each step, the algorithm combines the two clusters such that the distance between the two datapoints *furthest* from each other in the two clusters is minimized (thus "complete" linkage). This process repeats until either a set number of clusters is reached, or a distance threshold is reached.

The specific algorithm used is the scikit-learn implementaion `sklearn.cluster.AgglomerativeClustering`, described at `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html`.

Move lots of stuff from 4.X here!!!

3.5 PCA and Sparsity Methods

3.5.1 PCA

In order to evaluate the principal components of the activations in each layer of the network, we utilize the covariance matrix of the data in each layer l: Σ^l. This 1024×1024 matrix (for the **MLP-10** we are analysing), now has singular values squared of the singular values of the data. Dropping the l superscirpt for brevity, we perform singular value decomposition on Σ^l and find

$$\Sigma = V\Sigma' V^T \tag{3.2}$$

where V is orthonormal containing the eigenvectors of Σ in its columns, and $\Sigma' = \text{diag}[\sigma_0^2, \sigma_1^2, \ldots, \sigma_{1023}^2]$ contains the singular values of Σ ordered such that $\sigma_i^2 \geq \sigma_{i+1}^2$. Since the columns of V are the ordered eigenvectors of Σ, they are the principal components of the data and σ_i their corresponding covariances along that axis[3] [75].

3.5.2 Sparsity by ℓ_1 norm

In order to evaluate the sparsity of a normed vector in a high-dimensional space, we do not count the number of zero-entries in the vector but rather the alignment with the natural basis formed by the neurons.

Observe that if an ℓ_2 normed vector x is perfectly aligned with the neuron-basis, e.g. the first neuron so $x = [1, 0, 0, \ldots, 0]$, then it's ℓ_1 norm is $\|x\|_1 = 1$. If, on the other hand, x is perfectly misaligned with the axes, so $x = \frac{1}{\sqrt{d}}[1, 1, \ldots, 1]$, we find that the ℓ_1 norm is $\sqrt{d}$ where d is the dimensionality of the space. With this intuition, it is natural to see how the ℓ_1 norm can be used as a measure for (lack of) sparsity.

[3] We actually only need to take the SVD of the data itself, since the right singular vectors will correspond to the eigenvectors of Σ [75].

Further analysis shows that in fact in high dimensions d we find that the expected ℓ_1 norm of a vector x sampled from a uniform distribution on the ℓ_2 ball is

$$\sqrt{2d/\pi} \quad \text{with standard deviation} \quad \sqrt{1-3/\pi} \quad [76]. \tag{3.3}$$

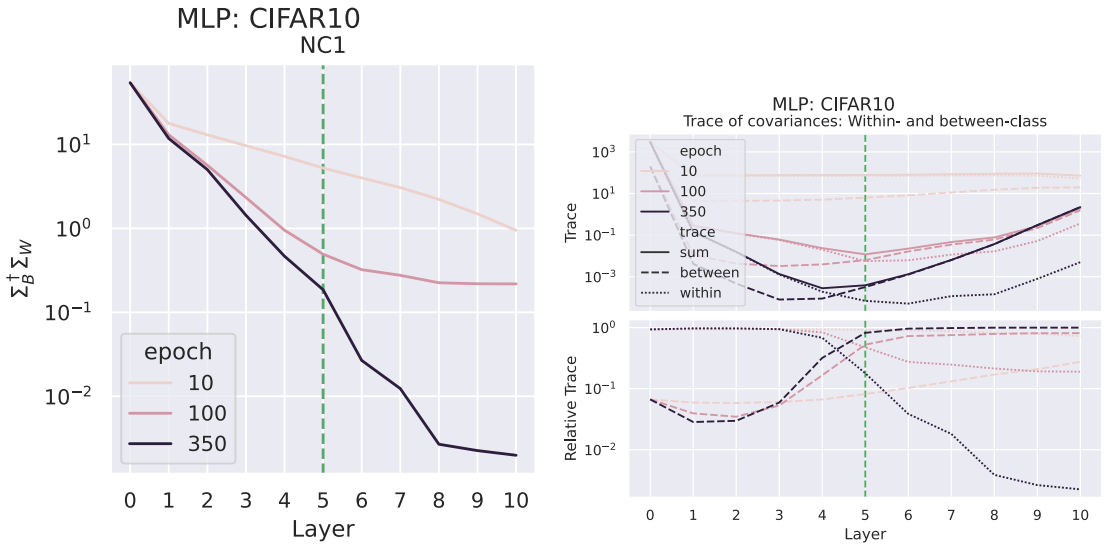
With this we can evaluate the "corresponding dimensionality" of a vector by assuming it was sampled randomly from the unit ℓ_2 sphere and taking the inverse of eq 3.3. This tells us, for a large enough sample, what dimensionality of a space aligned with the standard basis the vectors are sampled from.

In the relevant plots (figure 4.4.3) we also give the ℓ_1 norm of a random orthonormal matrix for comparison with the analytical calculation, and see that it empirically holds true.

CHAPTER

FOUR

RESULTS



(a) NC1 from [45] given in eq. 2.13 **(b)** NC1 - Trace measure following eq. 2.18

Figure 4.1.1: NC1 on MLP-10 trained with CIFAR10. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the input data.

4.1 Intermediate Neural Collapse

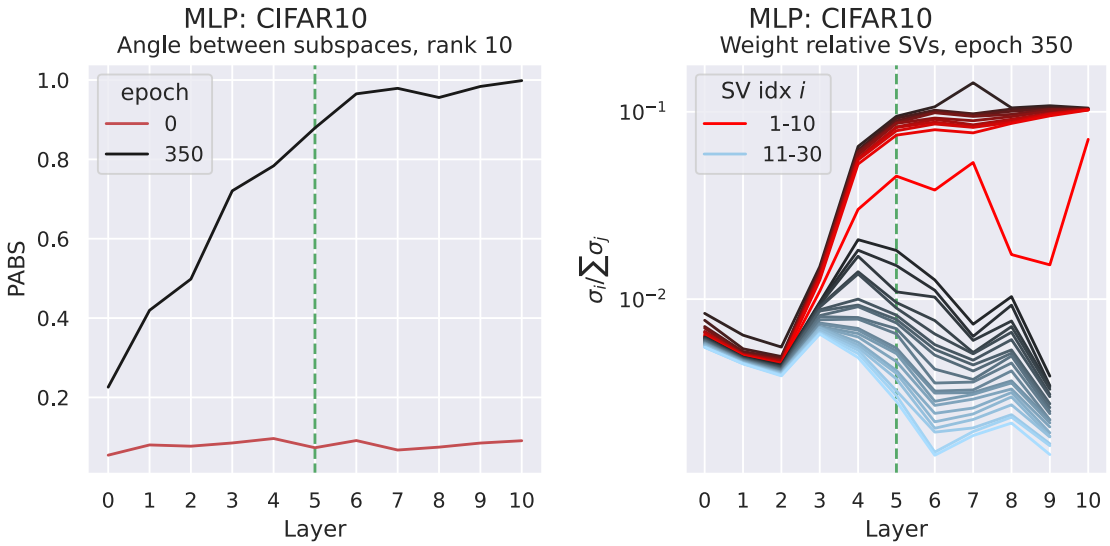
We first demonstrate the existence of all four properties of NC in an MLP-10 trained on CIFAR10. We then extend our comparison to MNIST to compare with the simpler dataset, before referencing our CIFAR10-results on a ConvNet-20 and Resnet50. Additional results for all arcitechtures (MLP-10, Convnet-20, ResNets) and datasets (MNIST, FashionMNIST, SVHN, CIFAR10) are shown in appendix B.

All code is available at the link given in appendix A.1. The hyperparameters resulting from the hyperparameter search are included. Accuracies of all reported results are given graphically in B.

4.1.1 NC1-4 on MLP-10

Trained with CIFAR10 we find that the MLP-10 achieves the NC1-condition of neural collapse around layer 5. This is shown in figure 4.1.1, and defined as the earliest layer when the relative within-class trace (detailed in eq. 2.18) is below a threshold, $\mathrm{Tr}[\Sigma_B/\Sigma_T] < 0.2$. Note that while the total trace varies in a parabolic-like curve, the representations steadily have an increasing proportion of between-class variance compared to within class variance.

The second condition, NC2, is also met. As seen in figure 4.1.2, the mean and standard deviation measured by eqs. 2.19, 2.20, and 2.21 converges to a lower value through training and becomes more pronounced particularly through the deeper layers after the point of collapse at layer 5.



(a) AvCosPABS defined by eq. 2.25 **(b)** Weight singular-values, as in eq. 2.23

Figure 4.1.3: NC3 on MLP-10 trained with CIFAR10. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the weight matrix acting on the input data. Convergence to 1 in AcCosPABS combined with convergence to $\frac{1}{C} = \frac{1}{10}$ for the first 10 singular values establish convergence to self-duality, by eqs. 2.25 and 2.24.

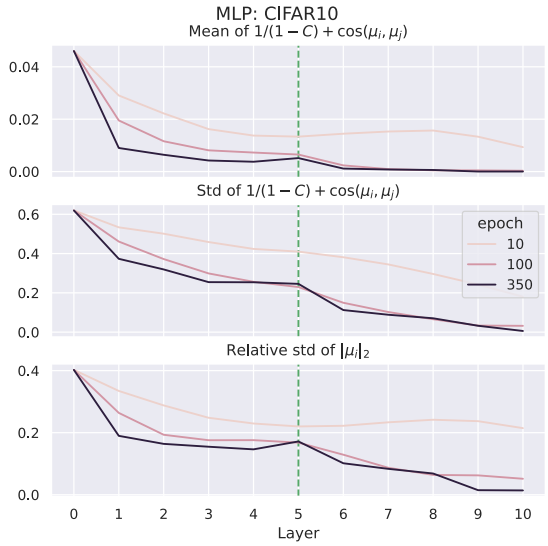


Figure 4.1.2: NC2 on MLP-10 trained with CIFAR10. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the input data. Convergence to 0 of all these measures demonstrates convergence to a Simplex ETF 1 by eqs. 2.19, 2.20, and 2.21.

Further, we also find NC3, the convergence to self-duality in the weights. As outlined in sec. 2.3.4, the convergence to overlapping subspaces of weights and class means combined with a rank C weight matrix is sufficient and necessary to establish NC3.

Lastly we also find that the NCC classifier achieves perfect accuracy on the train-set and performance on the test comparable to the full network at layer 5.

By figure 4.1.4, it is clear that after training the network the classification rule simplifies significantly. This is in correspondence with similar results shown in [46]. Note also that the comparatively poor performance on the test-set is due to the poor performance of MLPs on CIFAR10, and the test-accuracy of the NCC classifier is the same as the test-accuracy of the full network.

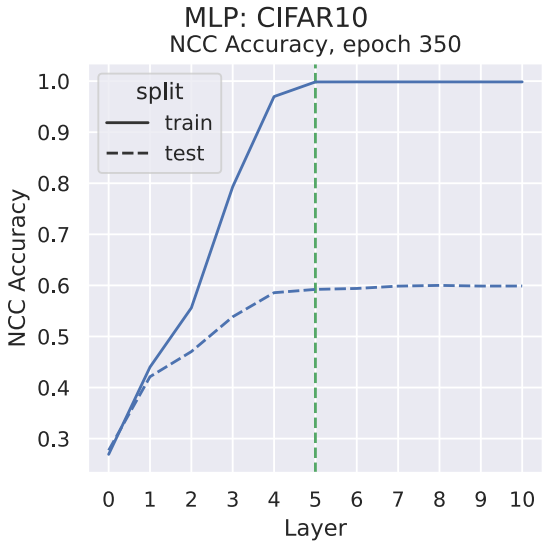


Figure 4.1.4: NC4 on MLP-10 trained with CIFAR10. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the input data. Convergence to a train-accuracy of 1 demonstrates NC4, that the classification rule is equivalent to the NCC classifier as per eq. 2.17.

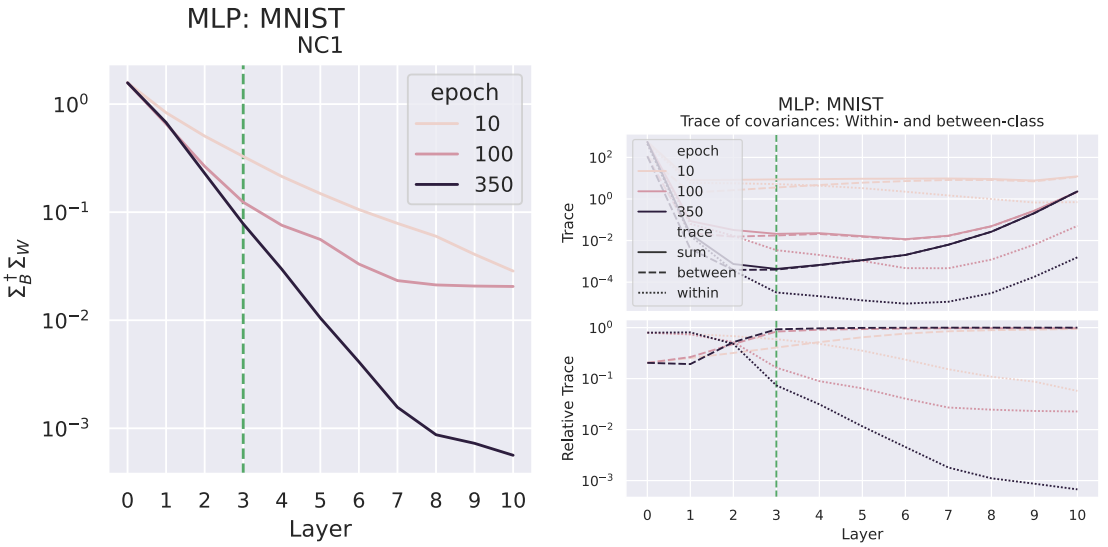
With this we have established the convergence towards and occurence of NC1-4 in the intermediate layers of MLP-10 trained on CIFAR10. See the appendix (sec. B) for results on other datasets.

4.1.2 Comparison across datasets and minimal depth

As established in [46], there is a bias towards minimal depth in DNNs and simpler training-data give rise to an earlier collapse in NC1 and NC4. We are now able to demonstrate that this effect also occurs in NC3, but apparently less perfectly in NC2.

Comparing the collapse and convergence as defined by the NC1-metric, we observe the collapse from around layer 3 when an MLP-10 is trained on MNIST (a considerably simpler dataset than CIFAR10 [71]). The depth regularization is seen when comparing figure 4.1.5 and 4.1.1, with collapse at layer 5 in MLP-10 on CIFAR10. This is as observed and predicted by [46].

In figure 4.1.6, measuring NC2, we do not immediately observe a convergence towards the Simplex ETF. Looking at the magnitude of the axes, however, one can observe that the magnitudes of the measures are as small as in figure 4.1.2 for CIFAR10. The convergence is not stronger than in CIFAR10, but happens strongly in later layers (layer 8).



(a) NC1 from [45] given in eq. 2.13 **(b)** NC1 - Trace measure following eq. 2.18

Figure 4.1.5: NC1 on MLP-10 trained with MNIST. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the input data.

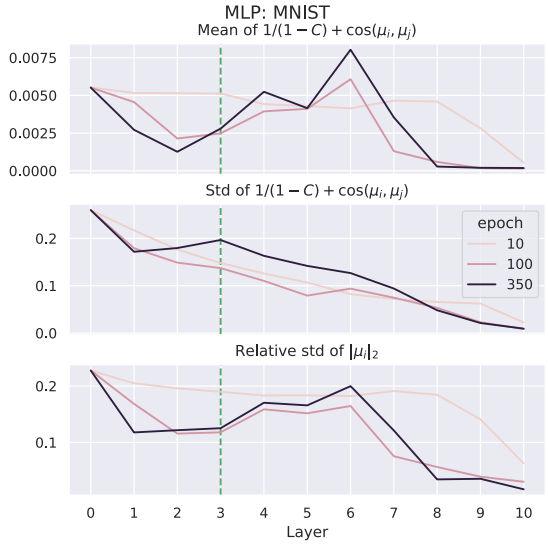
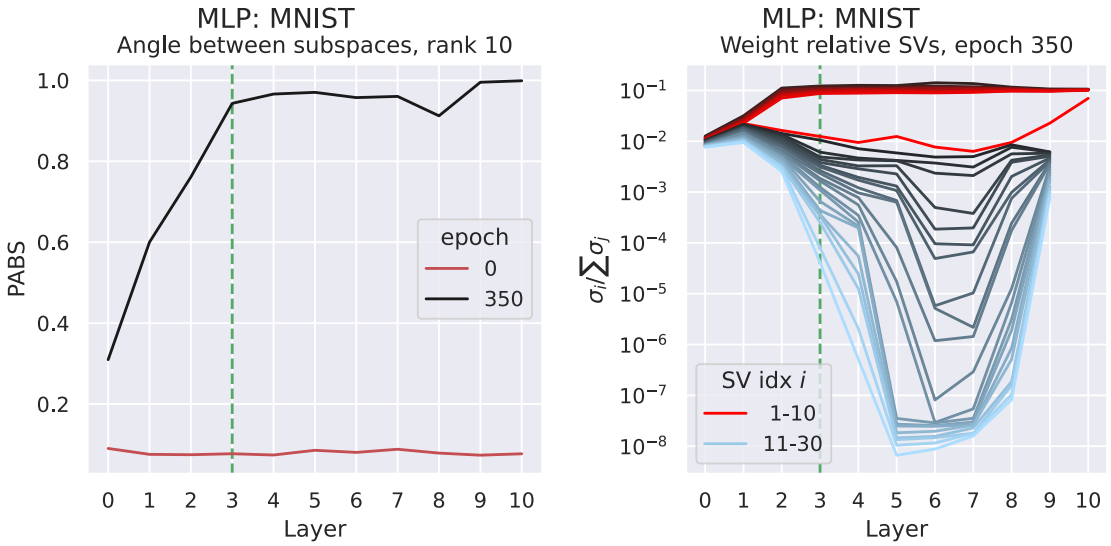


Figure 4.1.6: NC2 on MLP-10 trained with MNIST. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the input data. Convergence to 0 of all these measures demonstrates convergence to a Simplex ETF 1 by eqs. 2.19, 2.20, and 2.21.

NC3, on the other hand, happens both earlier (around layer 2) and stronger, demonstrated by the extremely low relative singular values in layers 4-8 after the first 10 singular values. This is shown in figure 4.1.7. Note also the convergence to a rank $9 = C - 1$ matrix, instead of a rank C matrix.

NC4 is left out for brevity, and shown in appendix B.



(a) AvCosPABS defined by eq. 2.25 **(b)** Weight singular-values, as in eq. 2.23

Figure 4.1.7: NC3 on MLP-10 trained with MNIST. Green line defines when the NC1-condition is met (to a sufficient degree). Layer 0 corresponds to the weight matrix acting on the input data. Convergence to 1 in AcCosPABS combined with convergence to $\frac{1}{C} = \frac{1}{10}$ for the first 10 singular values establish convergence to self-duality, by eqs. 2.25 and 2.24.

4.1.3 Extension to ConvNet and Resnet

We further demonstrate the extension of these properties to ConvNet-20 and Resnet50 with the CIFAR10 dataset.

For this we reference the plots given in the appendix, section B, specifically figure B.0.1 and B.0.2.[1]

These demonstrate the NC1-4 phenomenon, albeit not as perfectly as in the case for MLP-10. Note particularly the somewhat weaker NC3-condition. The observed "zig-zag" in the Resnet50 measurements is due to the block-wise nature of the Resnet architecture [10], and should not be considered an artifact of the NC convergence.

4.2 Intermediate collapse and generalization

By observing the correspondence between the layer of collapse and the generalization of the trained network we aim to determine whether INC is correlated with the DNNs generalizing ability.

To do this we train **MLP-10**, **ConvNet-20** and **ResNet50** on CIFAR10 using the same weight-decay and batch-size as previously. We vary the other hyperparameters, specifically the learning-rate and weight-decay, and observe the correspondence between the generalization (test-accuracy) and the layer of collapse of the networks.

The resulting layers of collapse and test-errors are presented in figure 4.2.1

[1] We do this as to easily provide a way of comparison between the models, and for brevity in the main text.

(with larger versions in appendix C.1). These plots demonstrate patterns between the layer of collapse and the generalization-ability of the network.

Firstly, all the networks with the hyperparameters optimized for generalization show a similar layer of collapse, which is somewhere in the middle of the network.

Secondly, for several runs with the same hyperparameters there is a general trend to end up with not just the same generalization-ability (similar along the y-axis), but also a similar layer of collapse (similar along the x-axis). This hints at the networks converging to the same set of optima, under some set of transformations, that have a common layer of collapse.

Thirdly, there are networks with similar but slightly worse generalization performance that collapse both at a deeper *and a shallower* layer than the optimum.

Finally, there are several networks that show collapse at a single layer, but they can have highly varying abilities to generalize.

Note that all of these points are especially clear on the relatively simpler MLP, and somewhat more noisy on the ConvNet.

4.3 Clustering of activations in intermediate layers

For this section the analysis is performed on a single network that has converged to Intermediate Neural Collapse. The DNN is a **MLP-10** (see fig. 3.2.1a) trained on the CIFAR10 dataset [73]. With learning rate 0.025, lr-decay factor of 0.2 four times in the total 500 epochs it was trained. The batch size used is 128, weight-decay is 0.002 and no warmup period is used. This is the same learning-rate and lr-decay as the top best performing network of the **MLP-10**s, but with 200 epochs of additional training with a further decayed learning rate in order to reduce noise.

The network shows collapse in layer 4, and has a training-accuracy of 99.82% and a test-accuracy of 59.7%. This is in line with the top performing networks in the previous analysis.

4.3.1 UMAP for 2D-visualization of neighborhoods

As described in section 3.4.1, we perform a UMAP clustering on the activations both in the input and on the hidden layers of the DNN. The clustering is performed with a cosine-distance metric, considering 20 neighbors. The following figure 4.3.1 is a sample of the results from this clustering, with each of the datapoints colored according to their true class.

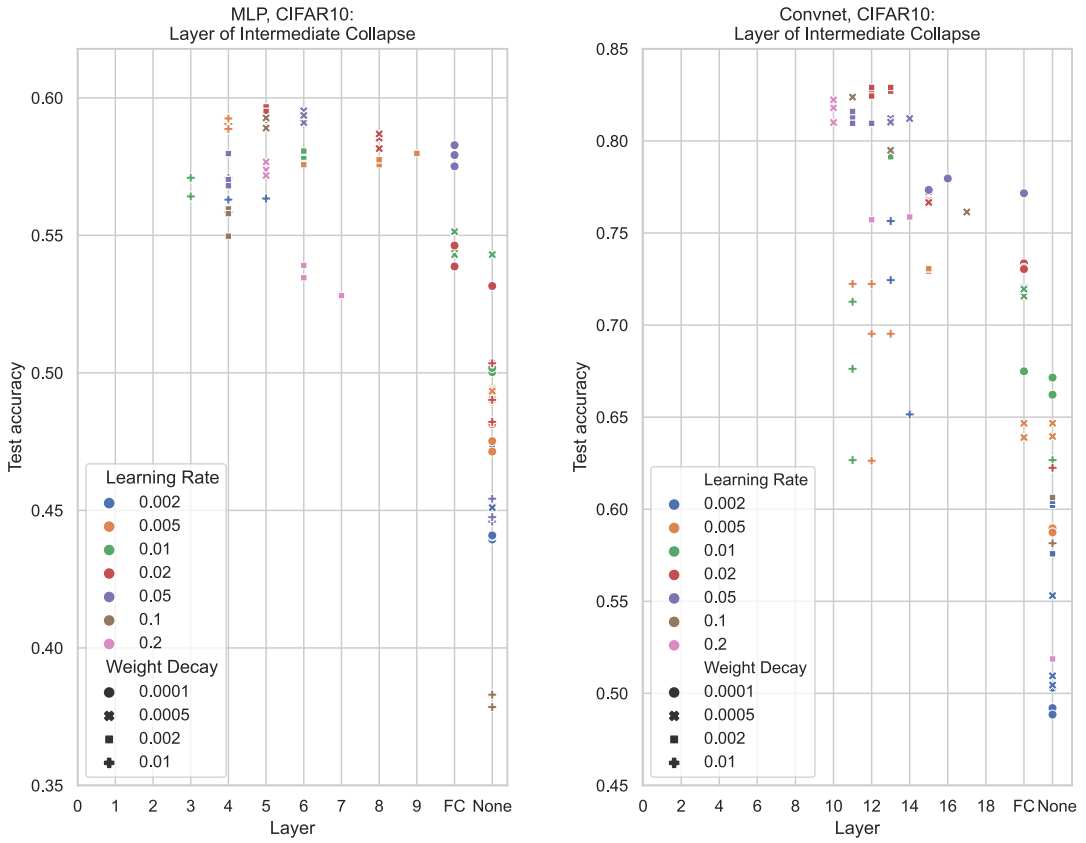
From the figure we observe that (a) visually the classes separate more strongly from each layer to the next, and (b) in the clustering before layer 4 (figure 4.3.1e) the classes are fully separated.

Larger figures are given in appendix C.2.1 for easier visual inspection.

4.3.2 Agglomerative clustering in cosine-space

Clustering the datapoints was done with agglomerative clustering using the cosine-angle between points as distace metric. [2]

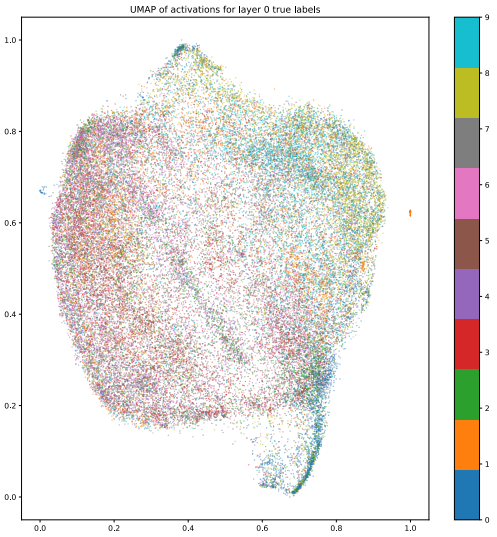
[2]The hyperparameters of agglomerative clustering were found with qualitative empirical analysis, but was stable in terms of the number of clusters for a range of hyperparameters.



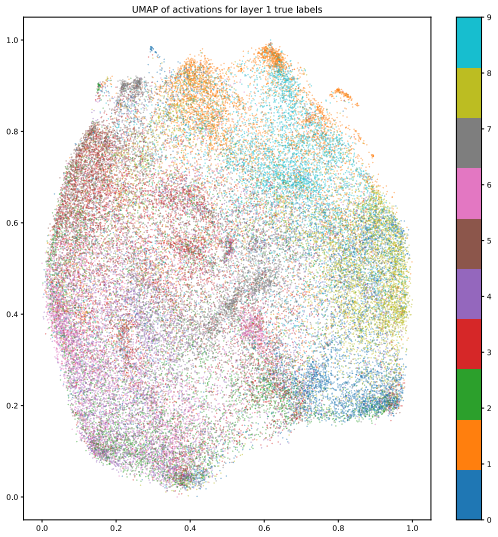
(a) Intermediate NC for MLP on CIFAR10 as a function of learning rate and weight decay.

(b) Intermediate NC for Convnet on CIFAR10 as a function of learning rate and weight decay.

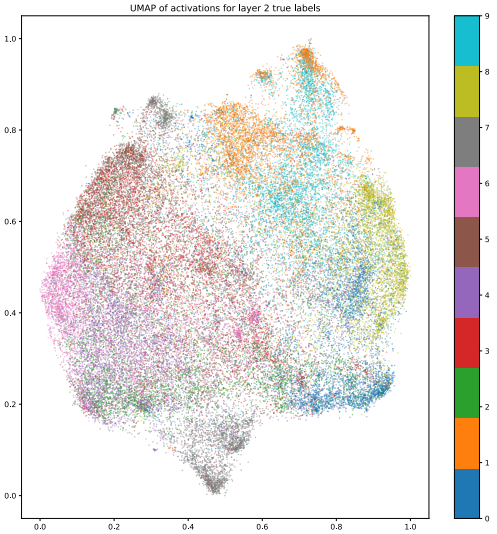
Figure 4.2.1: Layer of Intermediate Neural Collapse on CIFAR10 in two networks. This demonstrates the qualitatively different behaviors of the networks based on the layer of collapse for different hyperparameters. We ran 3 or 6 samples for each set of hyperparameters. Note that this has been cropped to exclude certain networks that all failed to converge to interpolating the dataset. For scaled up versons of these plots and further data on Resnet50, see C.1.



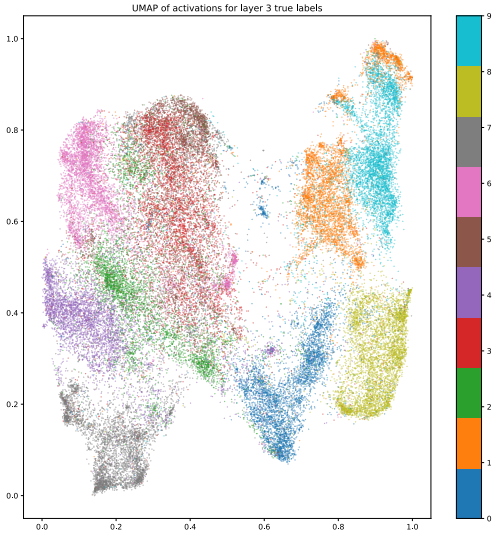
(a) UMAP clustering before layer 0



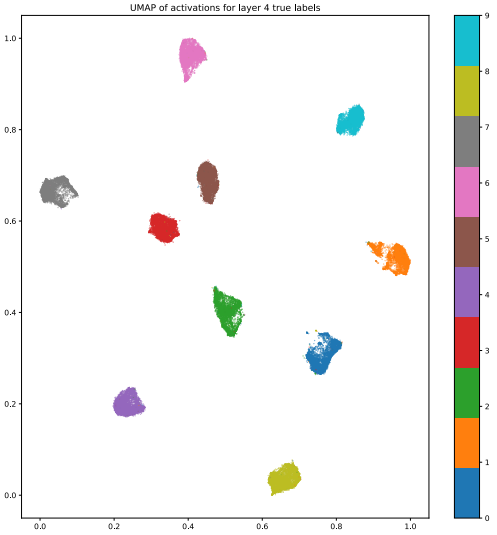
(b) UMAP clustering before layer 1



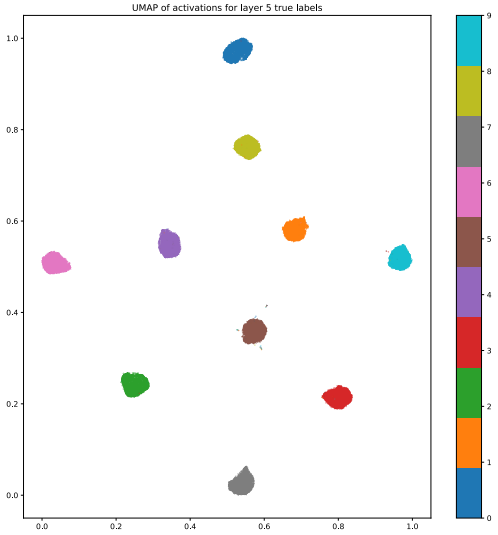
(c) UMAP clustering before layer 2



(d) UMAP clustering before layer 3



(e) UMAP clustering before layer 4



(f) UMAP clustering before layer 5

Figure 4.3.1: Plots of acticationsafter nonlinear projection with UMAP

Layer	Avg. #datapoints per cluster	Total Accuracy	Excluded datapoints
0	11	45%	1997
1	4.4	69%	12791
2	4.2	77%	14861
3	5.0	88%	8062
4	18	99.2%	130
5	543	99.9%	6

Table 4.3.1: Statistics on the clusterings before each layer. The total number of datapoints is 50,000.

The figures in this section are based on a clustering of the datapoints in activation-space with agglomerative clustering. The distance-metric used is the cosine-similarity, requiring complete linkage and a cosine angle threshold of 0.7. After this clustering, each cluster of 3 or more datapoints is given a label equal to the most prevalent class in that cluster. The clusters are then arranged based on their given label, and sorted within each label such that the cluster with the highest number of datapoints is listed first. The resulting confusion matrix from clusters to the element's true labels is plotted. See figure 4.3.2.

In interpreting this figure, it is useful to imagine how an "ideal" clustering would look: If all clusters perfectly only contain a slingle class, one would expect a perfectly diagonal set of dark areas, as each row (cluster) would contain only 0s in each other column (class) than the majority class.

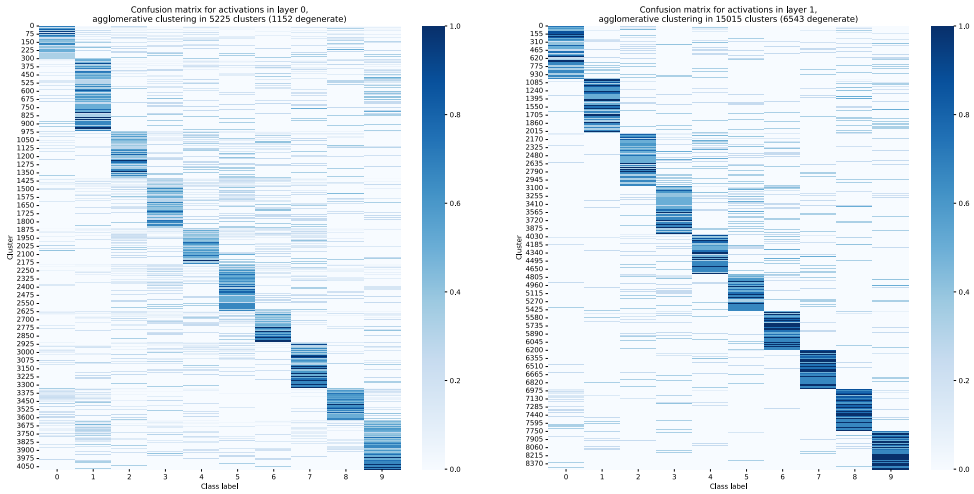
Notably, a perfectly random distribution of the data would still show some diagonal structure as the clusters are sorted after the most prevalent class in that cluster.

It is clear from the figure that there is some diagonal structure beyond the purely random, and that using the same hyperparameters gives a stronger class-based clustering for each applied layer in the network, that is: The accuracy increases with the depth of the network. Still, even before the collapse right before layer 4 (figure 4.3.2e), there is significant noise in the clusters.

In order to determine how well the data is clustered, we report the average number of datapoints per cluster and the accuracy of the clustering given labelling as specified. This can be read in table 4.3.1. We observe that deeper in the network, even before the collapse, there is a clear increase in the accuracy of the clusterings. Notably, we do not observe significant growth in average cluster-size, but (at least from layer 2 to 3) see a significant drop in "degenerate clusters" with only 1 or 2 datapoints in them: A drop from 14861 datapoints (29.7%) to 8062 datapoints (16.1%).

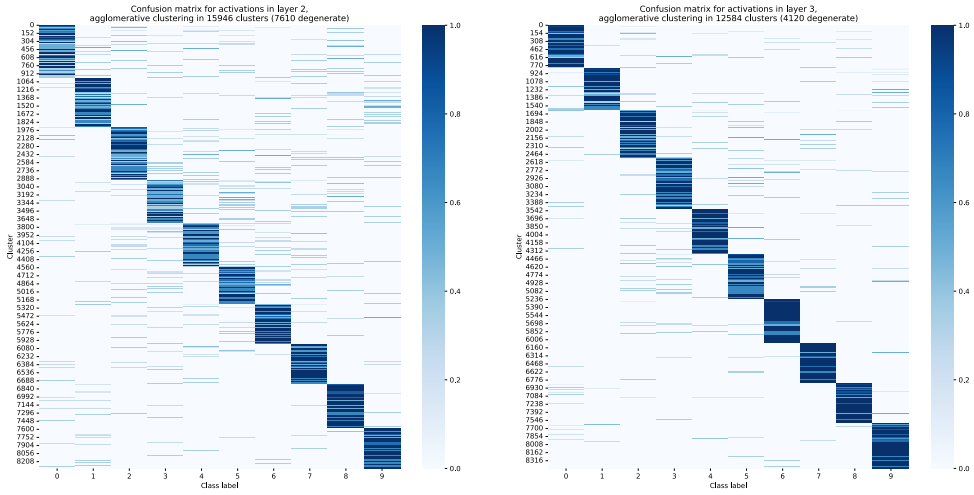
Appendix C.2 includes plots of the induced cluster labels on the UMAP clustering, as well as plots highlighting the disagreement between the induced labels and the true labels.

We also include the same plots using an euclidean distance metric in appendix C.2.2.



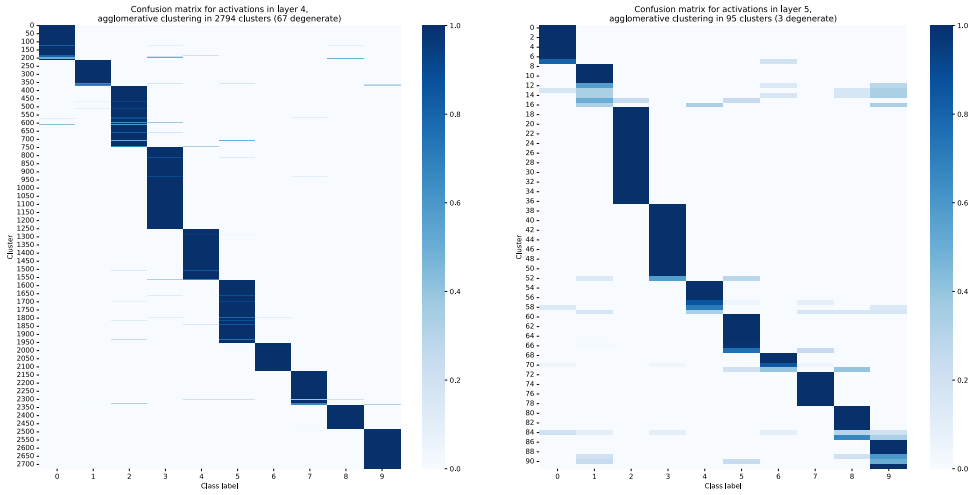
(a) Layer 0

(b) Layer 1



(c) Layer 2

(d) Layer 3



(e) Layer 4

(f) Layer 5

Figure 4.3.2: Plots of clusters from agglomerative clustering. Each row represents a cluster, and each column the class of the datapoints contained in that cluster. Each cluster (row) is normalized so it sums to 1, and as such the darker fields are the most prevalent in that cluster (row).

4.3.3 Lack of Hierarchichal Clustering

In order to evaluate whether the network performs hierarchical clustering, we can check whether the clusters in a shallow layer "stay together". That is, to what degree is a cluster in layer i a perfect subset of a cluster in layer $i+1$?

Before running the analysis, it is worth noting that we from table 4.3.1 see that the number of clusters from layer 1 to 2 is non-decreasing, but the classification-accuracy is slightly increasing. This gives us a strong prior that the clustering should not be hierarchical.

We report the % of datapoints in cluster a in layer l that is contained in the cluster in layer $l+1$ with the most datapoints from cluster a, and take the weighted average by number of datapoints in each cluster in layer l. That is, for each layer l we report

$$\frac{1}{NC}\sum_{a\in A^l}\max_{a'\in A^{l+1}}|a\cap a'| \tag{4.1}$$

where A^l is the set of clusters in layer l, NC is the total number of datapoints. The results are given in table 4.3.2.

We find that the rate at which a cluster stays connected from one layer is significanty lower than 100%.

Layer	Hierarchical accuracy
$0\rightarrow 1$	19%
$1\rightarrow 2$	48%
$2\rightarrow 3$	50%
$3\rightarrow 4$	81%
$4\rightarrow 5$	99.8%

Table 4.3.2: Accuracy of assuming hierarchical clusterings through each layer.

4.4 PCA and sparsity measurements

In order to evaluate the sparsity of the activations, we evaluate (a) whether the main principal components of the data span the full space (b) whether there are significantly sparse neurons, (c) whether the main singular vectors are aligned with the neuron-axes, and as such give a set of "most important neurons".

We evaluate this as described in section 3.5

4.4.1 PCA of activations

We plot the principal components of the activations in each layer in figure 4.4.1. From this figure, it is clear that pre-collapse, layer 1-3, there are no significantly different regimes. Specifically, the ordered singular values are largely continuous within each layer, and they are qualitatively similar between the layers. We do, however, observe a clear shift in layer 4 and deeper in the network. It is clear that after the collapse, the data occupies a significantly smaller space than the full 1024-dimensional space. This fits well with the observed neural collapse from

layer 4 and deeper, with the already described ever decreasing significance within-class variance (the tail of the singular values) compared to the major componens of between-class-mean-variance. (See section 4.1.)

4.4.2 Singular Vector Sparsity

Previous work [77] has shown that the singular vectors of the data contains important information on the characteristics of the network. In the theory on superposition [48] this is also highlighted that in DNNs that utilize neuron-wise operations have a bias towards aligning features along the principal axes of the activation space. Specifically for our networks, we could expect certain features of the activations to align with the neurons due to the neuron-wise ReLU activation function.

To this end, we evaluate the sparsity of the principal components of the data. We do this as described in section 3.5.2. We find the ℓ_1 norm of the singular vectors and compare to the expected norm for a random orthonormal matrix and the theoretical value of a random vector, and give the ℓ_1 norm of the vector in addition to the expected dimensionality that the vector is sampled from as described in section 3.5.2.

From figure 4.4.2 it is clear that pre-collapse the principal components of the data do not align to the neuron-basis. We make a few other observations as well. From the plots in the first three layers, the most significant principal component has a higher ℓ_1 norm than what is expected from a random 1024-dimensional vector. This corresponds to the global mean of the data being biased away from the neuron-axes and towards $\frac{1}{\sqrt{1024}}[1, \dots, 1]$.

Secondly, the the principal components except the (last and least significant ones) all have a corresponding dimensionality ~ 1024, showing that there is no bias to align the most significant directions with the neuron-basis.

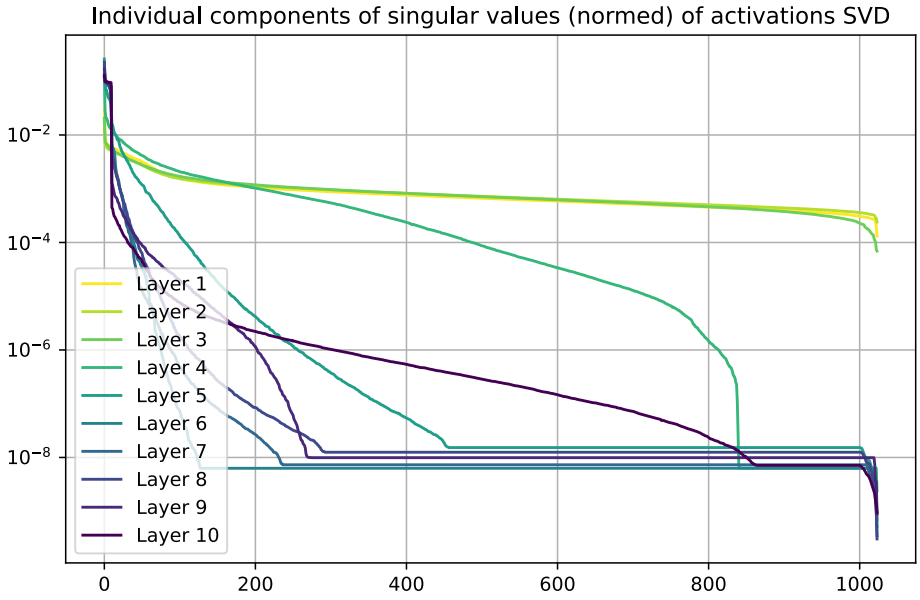
Thirdly, as a contrast to this, we see from figure 4.4.2d, 4.4.2e and 4.4.2f that there are several subspaces that are preferred by different magnitudes of singular vectors. As an example, observe that in layer 6 (figure 4.4.2e) there is a plateau from principal component 150 to 850. This plateau contains singular vectors with ℓ_1 norms that correspond to a randomly picked vector in a up to 700-dimensional space (read from the right y-axis). In the same plot, the region from 850 to 1000 contains vectors with norms picked from a up to 120-dimensional space. This shows that in the post-collapse regime, there is some bias towards aligning with the neuron-axes.

Plots for all layers and aligned with plots of the magnitudes of the principal components are given in appendix C.2 figure C.2.7 and C.2.7.

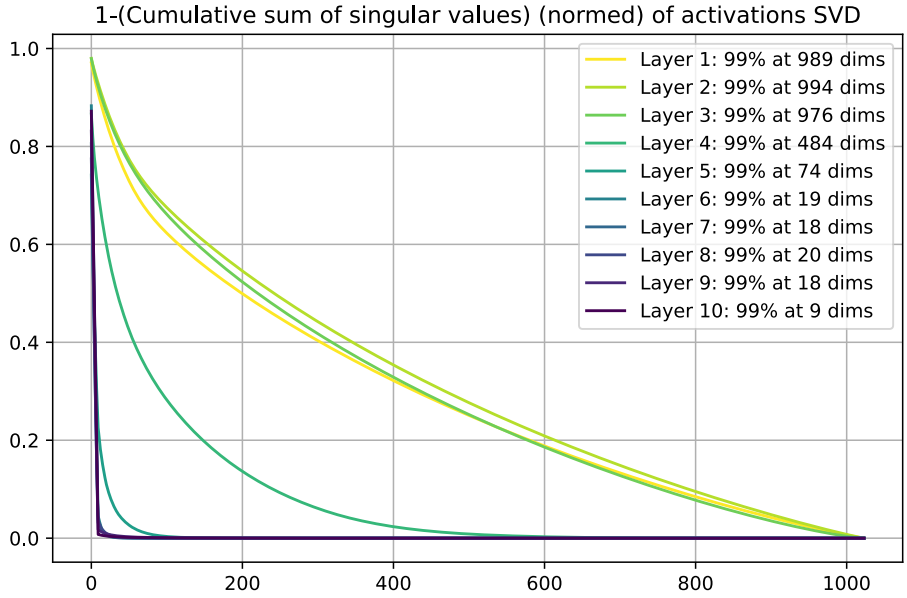
4.4.3 Sparsity of neuron activations

We also find the sparsity of each of the neurons. That is, how often does the neuron have a 0-activation, having been clamped to 0 by the preceding ReLU (see sec. 2.1.3)? To this end, we feed that data through the network, and for each neuron j in layer l observe the sparsity ratio s_i^l

$$s_j^l = \frac{1}{NC} \sum_{i,c} [[(h_{i,c}^l)_j == 0]] \tag{4.2}$$

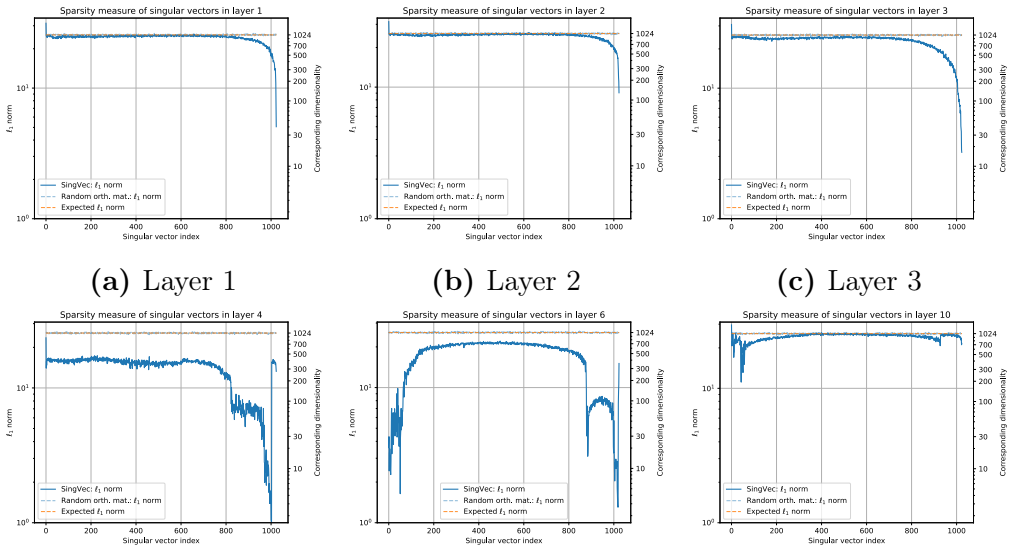


(a) Singular values of covariance matrix.



(b) 1 - (Cumulative sum of singular values), labelling the neuron at which it passes 0.01.

Figure 4.4.1: Plots of ordered singular values of the covariance matrix of the data at different layers.



(a) Layer 1 **(b)** Layer 2 **(c)** Layer 3

(d) Layer 4 **(e)** Layer 6 **(f)** Layer 10

Figure 4.4.2: ℓ_1 norm of ordered singular vectors of the

where C, N, c, i as before are the number of classes, the number of datapoints per class, class index, datapoint index, and $(h^l_{i,c})_j$ is the j-th element of the activation-vector of datapoint i from class c in layer l. We order the neurons by sparsity.

From figure 4.4.3 we observe that pre-collapse there is a simple structure in the sparsity of the neurons: They are all distributed continuously and tightly around approximately 0.6 with no neurons either extremely sparse or without sparsity.

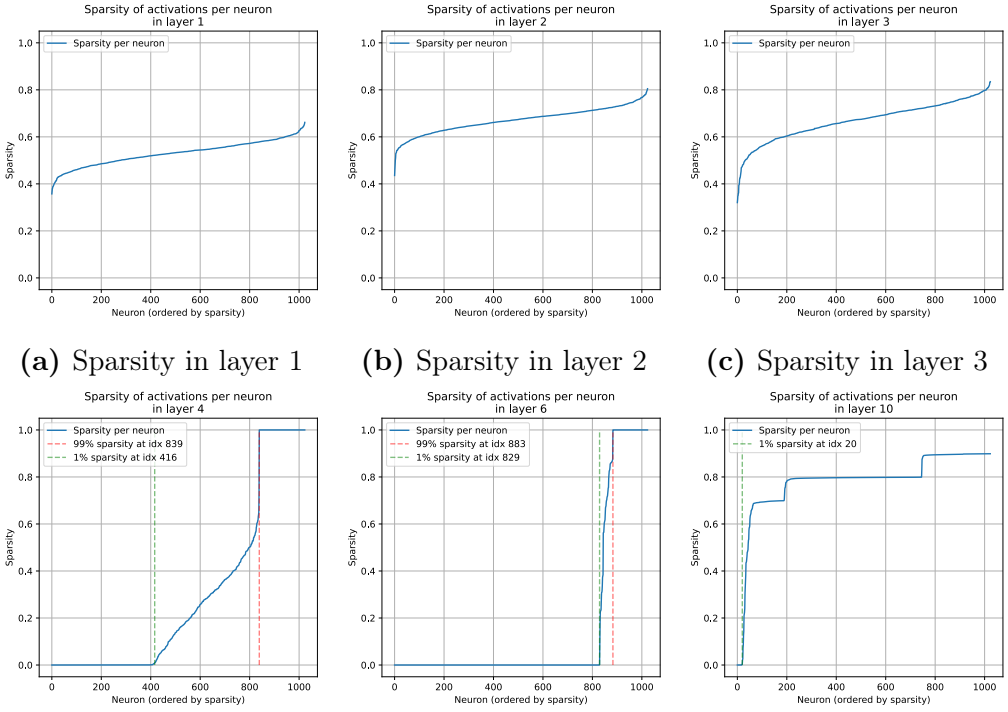
For comparison, the sparsity of select later layers is also provided, demonstrating that in the collapsed region of the DNN we find patters of extreme sparsity and extreme non-sparsity.

We also investigate the mean and standard deviation of all the non-zero activations of each neuron, to determine their importance.

From figure 4.4.4 we observe that the layers pre-collapse once again have a approximately continuous distribution. There is no clear structure of qualitatively different regimes, but interestingly it seems that the activations shift downwards by about half an order of magnitude per layer.

In this figure there is again a clear contrast between the pre-collapse layers and the post-collapse layers, where the post-collapse layers show clear regions of high-activations when sorted by sparsity. The regions of different regimes in the mean and std correspond to the regimes in the sparsity of the neurons (figure 4.4.3). Further study of these post-collapse phenomenons is left as future work.

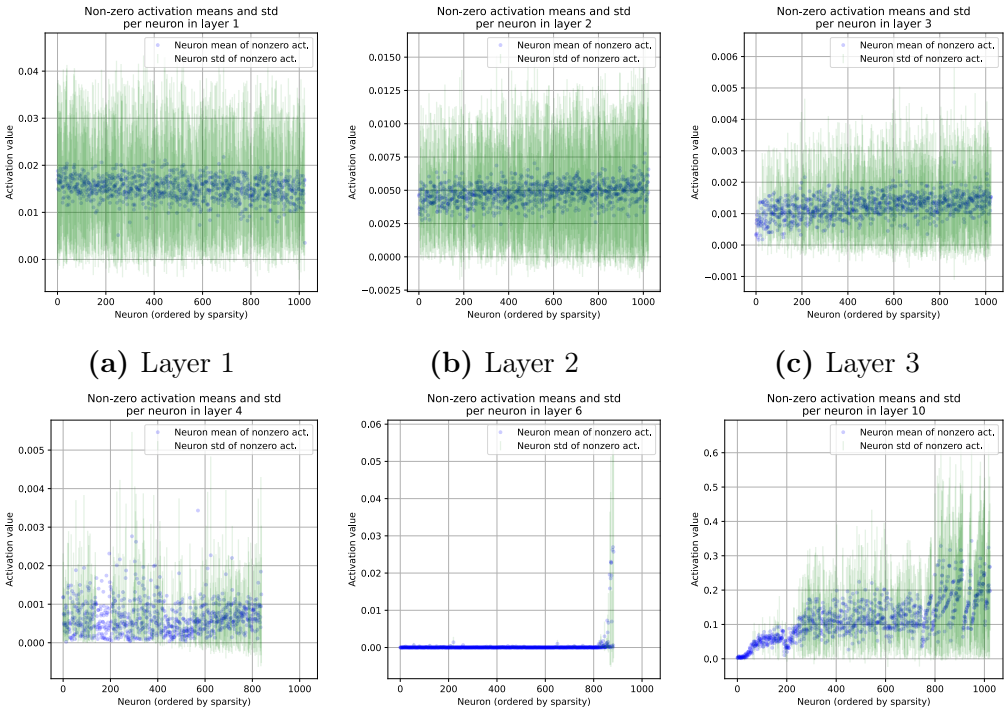
Further plots for all layers are provided in the appendix in figures C.2.9 and C.2.10.



(a) Sparsity in layer 1 **(b)** Sparsity in layer 2 **(c)** Sparsity in layer 3

(d) Sparsity in layer 4 **(e)** Sparsity in layer 6 **(f)** Sparsity in layer 10

Figure 4.4.3: Sparsity of Principal Components of the data, measured by the ℓ_1 norm of the vectors and their corresponding dimensionality.



(a) Layer 1 **(b)** Layer 2 **(c)** Layer 3

(d) Layer 4 **(e)** Layer 6 **(f)** Layer 10

Figure 4.4.4: Means of all non-zero activations in select layers. Blue dots represent the mean, and the green bars for each dot represents a two-sided standard deviation of the non-zero activation distribution for that neuron. Note that the green line crossing the y-axis does not imply that there are negative datapoints, but rather that the value of the standard deviation is larger than the mean. This implies a heavy (positive) tail distribution.

CHAPTER

FIVE

DISCUSSION

5.1 Implications of Intermediate Neural Collapse

We have demonstrated the convergence to NC1-4 through MLPs, Resnets and Convnets. Still, the convergence is not perfect and deserves additional analysis.

5.1.1 Imperfect convergence

Firstly, as observed in figure 4.1.7, the convergence towards the rank C matrix is not perfect in the simplest example on the simplest model. In fact, it seems like the convergence is happening towards a rank $C - 1 = 9$ matrix, especially in the intermediate layers.

Additionally, this is correlated with an imperfect convergence towards NC2, as described in section 4.1.2 and shown in figure 4.1.7.

A similar phenomenon might be present in the ConvNet on MNIST from figure B.0.3 and B.0.4, with higher NC2 discrepancies a matrix rank even lower than $C - 1$.

This raises several questions: Is a rank-C subspace the correct subspace for NC3? Does the intermediate representations in fact converge towards a Simplex ETF for the deepest networks on the easiest datasets, or just something quite similar? Do the effects of regularization in intermediate layers and the final layer result in different emergent structures?

We investigate the last question in the following sections, but leave the rest as future work.

5.1.2 Correlations of NC with generalization abilities

There seems to be a correlation between generalization abilities (as measured by test-accuracy) and NC properties. This is based on qualitative observations during hyperparameter-tuning. Specifically, it was observed, that there was a correlation between the final fractions of a percentage point of test-accuracy and the strength of the intermediate layer neural collapse. Intermediate layer NC seemed to occur by far the strongest in the most generalizing final models.

This observation motivates the work in section 5.2.

5.1.3 Extensions to more complex models

As demonstrated in appendix B, the convergence often stronger in the simpler models such as the MLP compared with the ConvNet and especially the Resnet. Still, the phenomenon exists.

For larger architectures with qualitatively different methods, e.g. the Transformer [30], this structure might be even less apparent or even not emerge at all.

The regime of state-of-the-art text analyzing models [78] might be qualitatively different from the regime of our experiments. Given the immense size of the training data of these models, it is not expected for them to be able to interpolate the training data. As our models interpolate the data with relative ease, the fundamental regime might be different from the largest, most useful, models in use today. It might still be the case that similar structures emerge in a somewhat

more limited form, and that the lens of NC analysis could give a way to analyze these models' behavior.

5.2 Intermediate Neural Collapse and Test Accuracy

From section 4.2 we observed that there is a correspondence between the layer of collapse and the test-accuracy of the DNNs. For clarity and simplicity, this analysis will focus on the results from the **MLP-10**. It is noted that the results seem to qualitatively extend to the **ConvNet-20** and **ResNet50**, though with increasingly noisy results.

From the four observations listed in the results, there are interesting conclusions to be drawn.

5.2.1 Layer of collapse and generalization

Previous work has established generalization bounds from the activations and layer of collapse to the generalizing abilities of the network.[46]

The results presented in this thesis show that collapsing at an earlier layer than the final layer is a clear indicator of improved perfomance, but there comes a saturation point after which earlier collapse is correlated with *worsening* generalization performance of the classifier.

This demonstrates, as the author notes in [46], that the bound is not tight. It also demonstrated that there is "no free lunch" in intermediate layer collapse; enforcing collapse at a shallow point in the network will correlate with improved performance up to a point, after which performance reaches an optimum and will worsen if shallow collapse enforced further.

5.2.2 Implications for circuts

The pattern observed in figure 4.2.1 displays quantitative qualities of optimizing with different hyperparameters and the different generalizing abilities of the resulting networks.

It is clear that with different hyperparameters the DNNs converge to different qualitative funciton implementations. Specifically, the process by which the DNN performs classification employs a different amount of functional pre-collapse layers for different hyperparameters. Consequently, this requires there to be a different set of circuts [19] implemented that performs the classification.

With this view, these similar networks with slight differences might provide a baseline for understanding how weight-decay influences the formation and optimization of circuts. For example, what is the ultimate difference between the circuts of (a) the networks with lr= 0.05, wd= 0.0005 that collapse in layer 6, and (b) the networks with lr= 0.02, wd= 0.002 that collapse in layer 5? Why does the latter generalize slightly better? Are there circuts in the former that are better pruned in the latter, and why specifically should they not be pruned further as demonstrated by the even worse generalization of (c) the networks with lr= 0.005, wd= 0.01 that collapse in layer 6?

Further attention is given to the circuts-analysis in section 1.3.2. We also explore another approach to how neural networks approach C-class clustering in the next section.

5.3 Hierarchical clusterings in DNNs showing intermediate NC

As an analouge to the C-class clustering in intermediate NC, one hypothesis for how neural networks perform classification is by iterative clustering. The main message of the following section that this is empirically *not* how a specific sample of a network showing intermediate NC implements the final C-class clustering.

We consider the network described in section 4.3, a single **MLP-10** showing neural collapse in the activations before layer 4. From the UMAP-plots (figure 4.3.1) it is visually clear that the classes become increasingly separated as they pass through the layers, even before the collapse in figure 4.3.1e. Still, as shown in section 4.3.2 (with additional treatment in appendix C.2), traditional clustering does not capture the structure that the DNN is performing computations on. Furthermore, as described in table 4.3.2 the clusters are clearly non-hierarchical which further suggest that the analyzed MLP does indeed not work in clusterings.

5.4 PCA and Sparsity analysis

In addition to the traditional method of PCA, sparsity has been a recurring and important topic in the analysis of DNNs (see e.g. [79] and [80]). We have first analyzed the space of the activations by their magnitude and direction of the principal components. Note that the activations before layer 0 is excluded in this analysis, as it is the raw input-data itself and does not accurately reflect the behavior of the DNN since it has not yet been transformed by any of the DNN layers.

Pre-collapse in activations before layers 1-3, as described in 4.4.1, we observe that the principal components of the space are continuous and without different regimes. In short, there is no preferred subspace and (approximately) the full 1024-dimensional space is utilized symmetrically. This is further established in section 4.4. It is clear from the sparsity of the singular vectors that they are generally not aligned with the neuron-basis but rather span most of the 1024-dimensional space with no preferred direction.

Post-collapse, however, we do observe interesting patterns in these metrics. As described in section 4.4.3, the post-collapse layers exhibit subsets of the principal components of the data with similar magnitude that spans a subspace of the 1024-dimensions. These subspaces do generally line up with the neuron-axes. This demonstrates, as alluded to in [48], that networks with a privileged basis can show a bias towards subspaces aligned with this basis. In our case, the privileged basis is established by the nonlinearity, the neuron-wise ReLU, and as such the subspaces align with this basis.

We also investigate the sparsity of the neurons themselves, and find the same pattern as in the PCA of the activations:

Pre-collapse, there is a continuum of sparsity in each neuron. From figure 4.4.3 and 4.4.4 there is no clear pattern established when ordering the neurons by sparsity in the pre-collapse layers. The sparsity of the neurons form a continuous distribution with no neurons on the extremes (figure 4.4.3a-c), and there is no apparent pattern in the mean or standard deviation of the activations (figure 4.4.4).

Post-collapse is again a contrast to these observations, where there are very clear regimes in the sparsity and mean activation of the neurons. We find that certain neurons are completely "regularized away" and exhibit 100% sparsity, for example the last few hundred neurons in layer 4 and 6 in figure 4.4.3. Additionally, the activations of the neurons with 0% sparsity (the first few hundred in layer 4 and 6) generally have a much lower activation than the sparse neurons, as shown in figure 4.4.4. We already know from section 5.1 that the layers after the intermediate layer collapse essentially do not perform any additional computation. As such, the observations in the post-collapse layers are clearly an artifact of the optimization procedure rather than being determined by the classification task and dataset. This naturally leads to the idea of *pruning* the full network down to something that does not utilize the final layers, or at leas only utilized the effective identity transformations that are needed to feed through the layers before classification.

5.5 Pruning and relation to the Lottery Ticket Hypothesis

5.5.1 Layer-wise Pruning

A common technique to decrease the resources needed to evaluate DNNs at inference time is pruning. [80] The pruning technique commonly includes setting weights to 0 and sometimes (as a consequence or explicitly) dropping out entire neurons. It is less common do drop entire layers of computation, effectively reducing the depth of the network. Our analysis, as can also be concluded from the published paper [1], suggests that pruning by dropping entire layers can be an efficient way of optimizing DNNs for resource use at inference time.

It is important to note that with the intruduction of residual (skip-) connections [10], pruning a network could theoretically lead to entire layers being pruned out. The residual connection provides an identity transformation as a backbone for the whole network, and by setting all weights or neurons to 0 in a block you effectively get an identity transform from one block to the next.

Still, while the history of DNN pruning is long and extensive, the approach of pruning entire layers seems to have attracted only a minority of the attention of other methods [81] [82]. Generally, "layer-collapse" (pruning away all neurons in a layer) has been a pitfall of many methods pruning too aggressively. One reason for layer-collapse generally not being preferred can be that the layer-pruning phenomenon is specific to networks that have been trained to Intermediate Neural Collapse, however previous work has observed that pruning usually is more aggressive in the later layers of the network [83]. This suggests that already for networks that are not trained entirely to INC, it is possible to discard large parts

and possibly all of later layers.

5.5.2 Hints against The Lottery Ticket Hypothesis

The Lottery Ticket Hypothesis as posed by [20] and described in 1.3.1 is essentially a statement about how an untrained network can be pruned to its winning "lottery ticket" already before training. Pruning procedures might be more aggressive in the later layers, but generally do not include pruning away entire layers to avoid "layer-collapse". As such, the observations in sections 5.3 and 5.4 where it is clear the network utilizes the full width and expressivity of the network, might possibly not be well described by the pruned "lottery ticket network". Using a full-depth neuron-subset of the network, as is the result in most pruning-algorithms, might not be compatible with full utilization of the first e.g. 3 layers of the network before complete Neural Collapse in e.g. layer 4 onwards.

Further study is required to determine whether the pruned version of the network analyzed in section 4.3 is similar to other pruned networks. Recent work has suggested that most if not all networks trained on CIFAR-10 converge to the same global basin and can be brought in alignment with a reference model through permutations [84]. In our case, it is not obvious whether this alignment can be achieved due to the qualitative layer-wise properties of the INC-network.

Specifically, relating to the LTH-pruned network, *is the network produced by evaluating the LTH on the pre-trained MLP equivalent to the pruned network after training to INC?* The author weakly conjectures that these might be different due to the likely benefit of utilizing the full depth of the network when finding the optimal "lottery-ticket". It is well known that several computational aspects of DNNs increase exponentially with depth but not with width [85]. Consequently, there is likely a similar relationship from the depth to the different "lottery-tickets" implemented by a network, and as such the method presented in LTH would likely find one of the tickets spanning the full network depth.

In the case where these pruned networks have a different architecture, it is reasonable to ask whether the implemented function is qualitatively as well as functionally different. If they are different, this would suggest that either (a) the DNN found by optimizing until INC, as in this thesis, does not correspond to the networks typically implemented by more standard training procedures, or (b) the network implemented by pruning according to LTH does not correspond to the DNNs trained by standard procedures. Still, these are empirical questions and left as future work.

5.6 Implications for Superposition and Representation Learning

Finally, considering only the pre-collapse layers, an interesting observation discussed in section 5.4 is that there is an apparent symmetry in the linear space of observations. There is no preferred directions, neither visible by PCA, the sparsity of the principal components, nor the activations or sparsity of the neurons. Except for the first singular vector acting as a bias, the network is seemingly fully utilized

in implementing the desired function. The sparsity of the neurons individually is around 60%, with a continuous and thight specter around the mean.

Intuitively, this matches well with the theory of superposition briefly mentioned in section 1.3.2, which assumes the full width of the network is utilized to represent as many sparse features as possible. If superpositions can be established in this regime, this will be a major step forwards towards decoding the qualitative implementation of the classifier in terms of features. We note that the ETF appearing post-collapse is equivalent to representing 10 features in 9 dimensions, exactly the type of superpositions covered in [48]. Why should we not see similar patterns before the collapse? Further investigations into the representations in these intermediate layers through the lens of superpositions of features is left as future work.

This analysed network also puts a very strict limit on the circuts the DNN can implement by virtue of its shallow nature. Since the network has been optimized in the overparametrized regime, having access to the full network, we can assume that while these circuts are relatively small, they share characteristics with the circuts in larger networks. In fact, as they are a consequence of optimizing to extremely small step-sizes and losses, we can expect these circuts to be more "pure" than circuts in other networks. Again, investigating these circuts is left as future work.

CHAPTER

SIX

CONCLUSION

6.1 Establishing Intermediate Neural Collapse

We have demonstrated the phenomenon of Neural Collapse in intermediate layers of a variety of DNN arcitechtures and for several datasets, namely the MLPs, Convnets and Resnets architechtures and MNIST, FashionMNIST, SVHN and CIFAR10 datasets. This was also included in [5] and parts in [1]. This is a step on the path towards understanding the behavior, performance and learning of DNNs, showing promising results for future analysis.

In particular, this line of research could provide a way to understand the full compositional function implemented by deep neural networks by extending analysis of the phenomenon to the pre-collapse layers in the network.

We have also established that while the layer of collapse is in some regard correlated with the generalizing abilities of the DNN, it is not a clear cut relation and some middle-point of optimal collapse for generalization seems to be the norm. This also impacts which circuts can apprear in the networks, as networks with shallower layers of collapse naturally will require that the circuts formed are shallower.

6.2 Feature analysis

It is also clear that we in sections 5.3 and 5.4 have described clear distinct patterns in the structure of the activations post-collapse (in the deeper layers). We have found sparsity, regimes of different principal component magnitudes, and that subspaces spanned by principal components with the same magnitude align themselves with the neuron-axes.

The feature patterns that we have have uncovered in the pre-collapse layers, however, are ones of fully covering the space with no clear separation between different parts of the features. We have seen that the features are near symmetric in every principal direction and magnitude, near symmetric in sparsity and magnitude in the neurons.

Investigating clustering algorithms in the pre-collapse layers have shown, visually with UMAP, that there is increasing structure as the data is transformed through the layers. We found, however, that attempts at clustering the datapoints gave only partially accurate results. We did not find any hierarchical set of clusters through the layers, nor were we able to cluster the datapoints solely based on their activations and find that the classes neatly correspond. This is importantly not at odds with theories of feature representations as concentrated along directions (such as in [49], [48] and [1]). It can very well be the case that interpretable features are aligned with linear directions in the space, even if the classes are not directly clusterable by these linear features.

In all, it is clear that there are patterns emerging in both pre- and post-collapse, but the interesting patters in pre-collapse have not lended themselves to separate into smaller more interpretetable chuncks with the methods we have investigated in this thesis.

6.3 Connections and future work

The patterns emerging in the intermediate layers relate to and can importantly be at odds with important theories of the inner workings of DNNs. Specifically we found that it is possible that the patterns exhibited by DNNs that show intermediate NC will not be equivalent to the pruned networks of the Lottery Ticket Hypothesis. This warrants future investigation, and could lead to insights into how optimization by pure SGD differs from the theory of the LTH.

For the theory of superposition and circuts we find that the network analyzed does not seem to be at odds with the theory, but could serve as a relatively simple example of how the circuts and superpositions are functioning. The network showing intermediate NC is a more complex example than most previous networks analyzed in this fashion, but likely a far simpler example than other networks due to its limited depth and optimization to convergence.

Other interesting phenomena that could be connected to the theory of intermediate neural collapse is DNN "grokking" [42]. As the grokking phenomenon occurs in networks trained to interpolation and the further with weight decay, we should expect them to exhibit some form of representation learning similar to NC. In fact, analysing the low-dimensional representations found in NC could be key to understanding the grokking phenomenon.

Additionally, the discovery and analysis of simplified topologies in a certain regime of DNNs could also be deeply related to emergent topologies in biological neural networks such as [86]. This offers the opportunity to use the statistical tools of neuroscience in DNNs, and is a promising direction of future research.

REFERENCES

[1] Akshay Rangamani*, Marius Lindegaard*, and Tomer Galanti Tomaso Poggio. *Feature learning in deep classifiers through Intermediate Neural Collapse*. 2023. URL: https://icml.cc/virtual/2023/poster/23548.

[2] Michael Chui et al. "The state of AI in 2023: Generative AI's breakout year". In: (2023). https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-in-2023-generative-AIs-breakout-year.

[3] Lei Gao and Ling Guan. *Interpretability of Machine Learning: Recent Advances and Future Prospects*. 2023. arXiv: 2305.00537 [cs.MM].

[4] Cris Olah. "Interpretability Dreams". In: (2023). https://transformer-circuits.pub/2023/interpretability-dreams/index.html.

[5] Marius Lindegaard, Akshay Rangamani, and Benjamin Dunn. "Project thesis: Intermediate Layer Collapse in Deep Classifiers". In: *NTNU Internal* (2023).

[6] Mette Langaas. *Advanced methods in Statistical Inference*. 2021. URL: https://github.com/mettelang/MA8701V2021/.

[7] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. DOI: 10.48550/ARXIV.1409.1556. URL: https://arxiv.org/abs/1409.1556.

[8] Preetum Nakkiran et al. *Deep Double Descent: Where Bigger Models and More Data Hurt*. 2019. DOI: 10.48550/ARXIV.1912.02292. URL: https://arxiv.org/abs/1912.02292.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[10] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. DOI: 10.48550/ARXIV.1512.03385. URL: https://arxiv.org/abs/1512.03385.

[11] Christian Szegedy et al. *Intriguing properties of neural networks*. 2013. DOI: 10.48550/ARXIV.1312.6199. URL: https://arxiv.org/abs/1312.6199.

[12] Md Zahangir Alom et al. *The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches*. 2018. DOI: 10.48550/ARXIV.1803.01164. URL: https://arxiv.org/abs/1803.01164.

[13] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. DOI: 10.48550/ARXIV.2010.11929. URL: https://arxiv.org/abs/2010.11929.

[14] Gabriel Goh et al. "Multimodal Neurons in Artificial Neural Networks". In: *Distill* (2021). https://distill.pub/2021/multimodal-neurons. DOI: 10.23915/distill.00030.

[15] Stuart Armstrong, Kaj Sotala, and Sean HEigeartaigh. "The errors, insights and lessons of famous AI predictions – and what they mean for the future". In: *Journal of Experimental & Theoretical Artificial Intelligence* 26 (June 2014). DOI: 10.1080/0952813X.2014.895105.

[16] Tomaso Poggio et al. *Theory of Deep Learning III: explaining the non-overfitting puzzle*. 2018. DOI: 10.48550/ARXIV.1801.00173. URL: https://arxiv.org/abs/1801.00173.

[17] Behnam Neyshabur et al. *Towards Understanding the Role of Over-Parametrization in Generalization of Neural Networks*. 2018. DOI: 10.48550/ARXIV.1805.12076. URL: https://arxiv.org/abs/1805.12076.

[18] B. Scholkopf and A.J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive Computation and Machine Learning series. MIT Press, 2018. ISBN: 9780262536578. URL: https://books.google.no/books?id=7r34DwAAQBAJ.

[19] Chris Olah et al. "Zoom In: An Introduction to Circuits". In: *Distill* (2020). https://distill.pub/2020/circuits/zoom-in. DOI: 10.23915/distill.00024.001.

[20] Jonathan Frankle and Michael Carbin. *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks*. 2019. arXiv: 1803.03635 [cs.LG].

[21] Nelson Elhage et al. "A Mathematical Framework for Transformer Circuits". In: *Transformer Circuits Thread* (2021). https://transformer-circuits.pub/2021/framework/index.html.

[22] Arthur Conmy et al. *Towards Automated Circuit Discovery for Mechanistic Interpretability*. 2023. arXiv: 2304.14997 [cs.LG].

[23] Michael Chui et al. "The economic potential of generative AI: The next productivity frontier". In: (2023). https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier.

[24] Several Authors. "Mitigating the risk of extinction from AI should be a global priority alongside other societal-scale risks such as pandemics and nuclear war." In: (2023). https://www.safe.ai/statement-on-ai-risk.

[25] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. DOI: 10.48550/ARXIV.2005.14165. URL: https://arxiv.org/abs/2005.14165.

[26] Neel Nanda. "A Longlist of Theories of Impact for Interpretability". In: *AI Alignment Forum* (2022). https://www.alignmentforum.org/posts/uK6sQCNMw8WKzJeCQ/a-longlist-of-theories-of-impact-for-interpretability.

[27] Several Authors. "Core Views on AI Safety: When, Why, What, and How". In: *Anthropic* (2023). https://www.anthropic.com/index/core-views-on-ai-safety.

[28] Lee Sharkey, Sid Black, and Beren Millidge. "Current Themes in Mechanistic Interpretability Research". In: *AI Alignment Forum* (2022). https://www.alignmentforum.o themes-in-mechanistic-interpretability-research.

[29] Neel Nanda. "Another list of theories of imact for interpretability". In: *AI Alignment Forum* (2023). https://www.alignmentforum.org/posts/YQALrtMkeqemAF5GX/another-list-of-theories-of-impact-for-interpretability.

[30] Ashish Vaswani et al. *Attention Is All You Need*. 2017. DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.

[31] Aditya Ramesh et al. *Zero-Shot Text-to-Image Generation*. 2021. arXiv: 2102.12092 [cs.CV].

[32] Scott Reed et al. *A Generalist Agent*. 2022. arXiv: 2205.06175 [cs.AI].

[33] Danijar Hafner et al. *Mastering Diverse Domains through World Models*. 2023. arXiv: 2301.04104 [cs.AI].

[34] Arthur Jacot, Franck Gabriel, and Clément Hongler. *Neural Tangent Kernel: Convergence and Generalization in Neural Networks*. 2020. arXiv: 1806.07572 [cs.LG].

[35] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[36] Minshuo Chen et al. *Towards Understanding Hierarchical Learning: Benefits of Neural Representations*. 2021. arXiv: 2006.13436 [cs.LG].

[37] Ravid Shwartz-Ziv and Naftali Tishby. *Opening the Black Box of Deep Neural Networks via Information*. 2017. arXiv: 1703.00810 [cs.LG].

[38] Hao Li et al. *Visualizing the Loss Landscape of Neural Nets*. 2018. arXiv: 1712.09913 [cs.LG].

[39] Mengjia Xu et al. "Dynamics in Deep Classifiers Trained with the Square Loss: Normalization, Low Rank, Neural Collapse, and Generalization Bounds". In: *Research* 6 (Jan. 2023). DOI: 10.34133/research.0024.

[40] Catherine Olsson et al. "In-context Learning and Induction Heads". In: *Transformer Circuits Thread* (2022). https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[41] Mikhail Belkin et al. "Reconciling modern machine-learning practice and the classical bias–variance trade-off". In: *Proceedings of the National Academy of Sciences* 116.32 (July 2019), pp. 15849–15854. DOI: 10.1073/pnas.1903070116. URL: https://doi.org/10.1073%2Fpnas.1903070116.

[42] Alethea Power et al. *Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets*. 2022. arXiv: 2201.02177 [cs.LG].

[43] Jason Wei et al. *Emergent Abilities of Large Language Models*. 2022. arXiv: 2206.07682 [cs.CL].

[44] Deep Ganguli et al. "Predictability and Surprise in Large Generative Models". In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. ACM, June 2022. DOI: 10.1145/3531146.3533229. URL: https://doi.org/10.1145%2F3531146.3533229.

[45] Vardan Papyan, X. Y. Han, and David L. Donoho. "Prevalence of neural collapse during the terminal phase of deep learning training". In: *Proceedings of the National Academy of Sciences* 117.40 (Sept. 2020), pp. 24652–24663. DOI: 10.1073/pnas.2015509117. URL: https://doi.org/10.1073%5C%2Fpnas.2015509117.

[46] Tomer Galanti, Liane Galanti, and Ido Ben-Shaul. *On the Implicit Bias Towards Minimal Depth of Deep Neural Networks*. 2022. DOI: 10.48550/ARXIV.2202.09028. URL: https://arxiv.org/abs/2202.09028.

[47] Minghui Chen. "Awesome Deep Phenomena". In: (2023). https://github.com/MinghuiChen43/awesome-deep-phenomena.

[48] Nelson Elhage et al. "Toy Models of Superposition". In: *Transformer Circuits Thread* (2022). https://transformer-circuits.pub/.

[49] Preetum Nakkiran et al. *Deep Double Descent: Where Bigger Models and More Data Hurt*. 2019. arXiv: 1912.02292 [cs.LG].

[50] X. Y. Han, Vardan Papyan, and David L. Donoho. *Neural Collapse Under MSE Loss: Proximity to and Dynamics on the Central Path*. 2021. DOI: 10.48550/ARXIV.2106.02073. URL: https://arxiv.org/abs/2106.02073.

[51] Stefanie Muff and Thiago Martins. *Statistical Learning (NTNU - TMA4268)*. 2022. URL: https://wiki.math.ntnu.no/tma4268/2022v/start.

[52] Geoffrey E. Hinton Vinod Nair. "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *Proceedings of the International Conference on Machine Learning* (2010). URL: https://www.cs.toronto.edu/%5C%7Efritz/absps/reluICML.pdf.

[53] Tomer Galanti et al. *SGD and Weight Decay Provably Induce a Low-Rank Bias in Neural Networks*. 2022. DOI: 10.48550/ARXIV.2206.05794. URL: https://arxiv.org/abs/2206.05794.

[54] Tianxiang Gao et al. *A global convergence theory for deep ReLU implicit networks via over-parameterization*. 2021. DOI: 10.48550/ARXIV.2110.05645. URL: https://arxiv.org/abs/2110.05645.

[55] Yann LeCun et al. "Handwritten Digit Recognition with a Back-Propagation Network". In: *NIPS*. 1989.

[56] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. DOI: 10.48550/ARXIV.1502.03167. URL: https://arxiv.org/abs/1502.03167.

[57] Jinxin Zhou et al. *On the Optimization Landscape of Neural Collapse under MSE Loss: Global Optimality with Unconstrained Features*. 2022. DOI: 10.48550/ARXIV.2203.01238. URL: https://arxiv.org/abs/2203.01238.

[58] Robert Kleinberg, Yuanzhi Li, and Yang Yuan. *An Alternative View: When Does SGD Escape Local Minima?* 2018. DOI: 10.48550/ARXIV.1802.06175. URL: https://arxiv.org/abs/1802.06175.

[59] Sébastien Bubeck and Mark Sellke. *A Universal Law of Robustness via Isoperimetry*. 2021. DOI: 10.48550/ARXIV.2105.12806. URL: https://arxiv.org/abs/2105.12806.

[60] Qianli Liao and Tomaso Poggio. *Theory II: Landscape of the Empirical Risk in Deep Learning*. 2017. DOI: 10.48550/ARXIV.1703.09833. URL: https://arxiv.org/abs/1703.09833.

[61] Karthik A. Sankararaman et al. *The Impact of Neural Network Overparameterization on Gradient Confusion and Stochastic Gradient Descent*. 2019. DOI: 10.48550/ARXIV.1904.06963. URL: https://arxiv.org/abs/1904.06963.

[62] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. "Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks". In: *Nature Communications* 12.1 (May 2021). DOI: 10.1038/s41467-021-23103-1. URL: https://doi.org/10.1038%2Fs41467-021-23103-1.

[63] Chiyuan Zhang et al. *Understanding deep learning requires rethinking generalization*. 2016. DOI: 10.48550/ARXIV.1611.03530. URL: https://arxiv.org/abs/1611.03530.

[64] Li Yuan et al. *Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet*. 2021. DOI: 10.48550/ARXIV.2101.11986. URL: https://arxiv.org/abs/2101.11986.

[65] Maithra Raghu et al. *Do Vision Transformers See Like Convolutional Neural Networks?* 2021. DOI: 10.48550/ARXIV.2108.08810. URL: https://arxiv.org/abs/2108.08810.

[66] Yann Le Cun, Ido Kanter, and Sara A. Solla. "Eigenvalues of covariance matrices: Application to neural-network learning". In: *Phys. Rev. Lett.* 66 (18 May 1991), pp. 2396–2399. DOI: 10.1103/PhysRevLett.66.2396. URL: https://link.aps.org/doi/10.1103/PhysRevLett.66.2396.

[67] A Krogh and J A Hertz. "Generalization in a linear perceptron in the presence of noise". In: 25.5 (Mar. 1992), p. 1135. DOI: 10.1088/0305-4470/25/5/020. URL: https://dx.doi.org/10.1088/0305-4470/25/5/020.

[68] Thomas Strohmer and Robert Heath. *Grassmannian Frames with Applications to Coding and Communication*. 2003. DOI: 10.48550/ARXIV.MATH/0301135. URL: https://arxiv.org/abs/math/0301135.

[69] Ake Bjorck and Gene Golub. "Numerical Methods for Computing Angles Between Linear Subspaces". In: *Mathematics of Computation* 27 (July 1973), p. 123. DOI: 10.2307/2005662.

[70] Li Deng. "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.

[71] Han Xiao, Kashif Rasul, and Roland Vollgraf. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. 2017. DOI: 10.48550/ARXIV.1708.07747. URL: https://arxiv.org/abs/1708.07747.

[72] Yuval Netzer et al. "Reading Digits in Natural Images with Unsupervised Feature Learning". In: *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*. 2011. URL: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.

[73] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images". In: (2009), pp. 32–33. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[74] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. 2020. arXiv: 1802.03426 [stat.ML].

[75] *Principal Component Analysis — Wikipedia, The Free Encyclopedia*. [Online; accessed 2023-09-18]. 2004. URL: %5Curl%7Bhttps://en.wikipedia.org/wiki/Principal_component_analysis%7D.

[76] John Mount. "Expected L1 Norm of Unit L2 vectors". In: *Win-Vector* (2023). https://win-vector.com/2023/06/24/the-expected-l1-norm-of-unit-l2-vectors/.

[77] Vardan Papyan. *Traces of Class/Cross-Class Structure Pervade Deep Learning Spectra*. 2020. arXiv: 2008.11865 [cs.LG].

[78] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. DOI: 10.48550/ARXIV.2005.14165. URL: https://arxiv.org/abs/2005.14165.

[79] Wei Wen et al. *Learning Structured Sparsity in Deep Neural Networks*. 2016. arXiv: 1608.03665 [cs.NE].

[80] Trevor Gale, Erich Elsen, and Sara Hooker. *The State of Sparsity in Deep Neural Networks*. 2019. arXiv: 1902.09574 [cs.LG].

[81] Tailin Liang et al. *Pruning and Quantization for Deep Neural Network Acceleration: A Survey*. 2021. arXiv: 2101.09671 [cs.CV].

[82] Davis Blalock et al. *What is the State of Neural Network Pruning?* 2020. arXiv: 2003.03033 [cs.LG].

[83] Pavlo Molchanov et al. *Pruning Convolutional Neural Networks for Resource Efficient Inference*. 2017. arXiv: 1611.06440 [cs.LG].

[84] Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. *Git Re-Basin: Merging Models modulo Permutation Symmetries*. 2023. arXiv: 2209.04836 [cs.LG].

[85] Guido Montúfar et al. *On the Number of Linear Regions of Deep Neural Networks*. 2014. arXiv: 1402.1869 [stat.ML].

[86] Richard Gardner et al. "Toroidal topology of population activity in grid cells". In: *Nature* 602 (Feb. 2022), pp. 1–6. DOI: 10.1038/s41586-021-04268-7.

[87] Tomer Galanti, András György, and Marcus Hutter. "On the Role of Neural Collapse in Transfer Learning". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=SwIp410B6aQ.

APPENDICES

APPENDIX

A

CODEBASE AND DATASETS

A.1 Github Repository

The code for running the experiments and generating the figures in this document is given at

- https://github.com/mariuslindegaard/DNN_Analysis/tree/dev/thesis

A.2 Dataset specifications

Dataset	#Train	#Test	Input dim.	Classes
MNIST [70]	60k	10k	$(1 \times 28 \times 28)$	10
FashionMNIST [71]	60k	10k	$(1 \times 28 \times 28)$	10
SVHN [72]	$\sim$73k	$\sim$26k	$(3 \times 32 \times 32)$	10
CIFAR10 [73]	50k	10k	$(3 \times 32 \times 32)$	10

APPENDIX

B

INTERMEDIATE NEURAL COLLAPSE: EXPERIMENTAL RESULTS

We show the results from doing the analysis of intermediate NC in the full set of models and datasets. These results are also presented in [1].

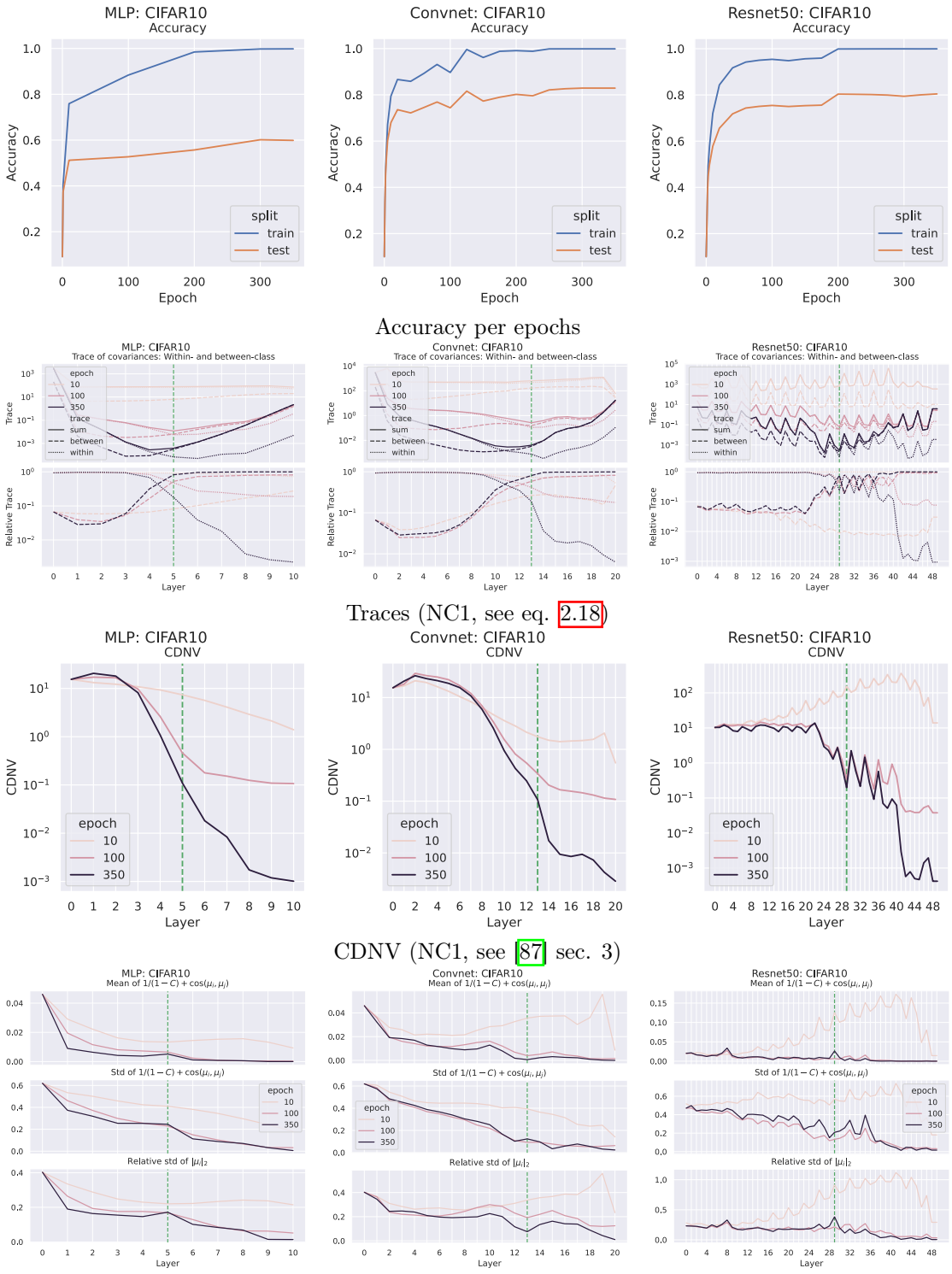


Figure B.0.1: Accuracy and NC1-2 on MLP-10, ConvNet-20 and Resnet50 trained with CIFAR10.

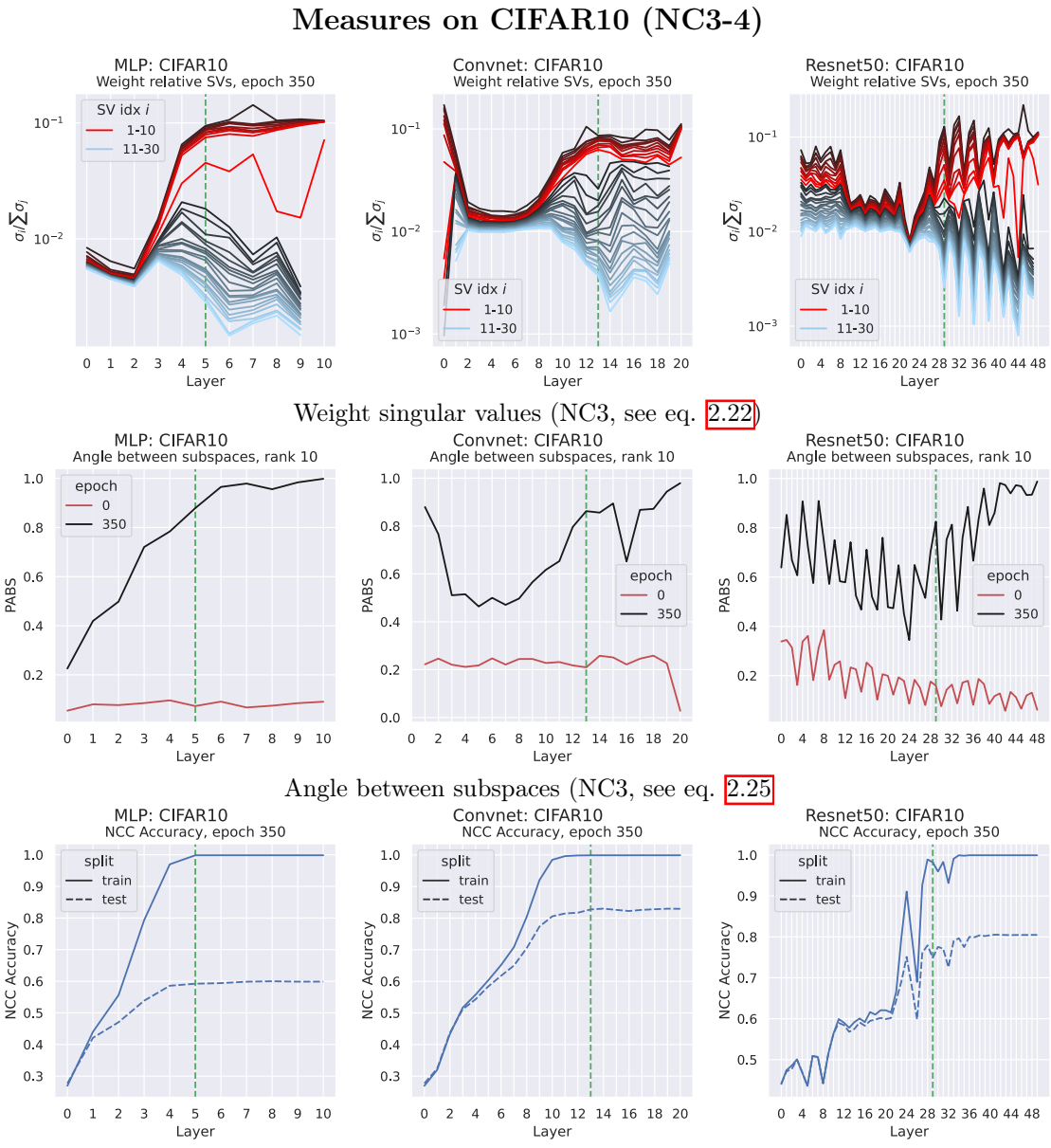


Figure B.0.2: NC3-4 on MLP-10, ConvNet-20 and Resnet50 trained with CIFAR10.

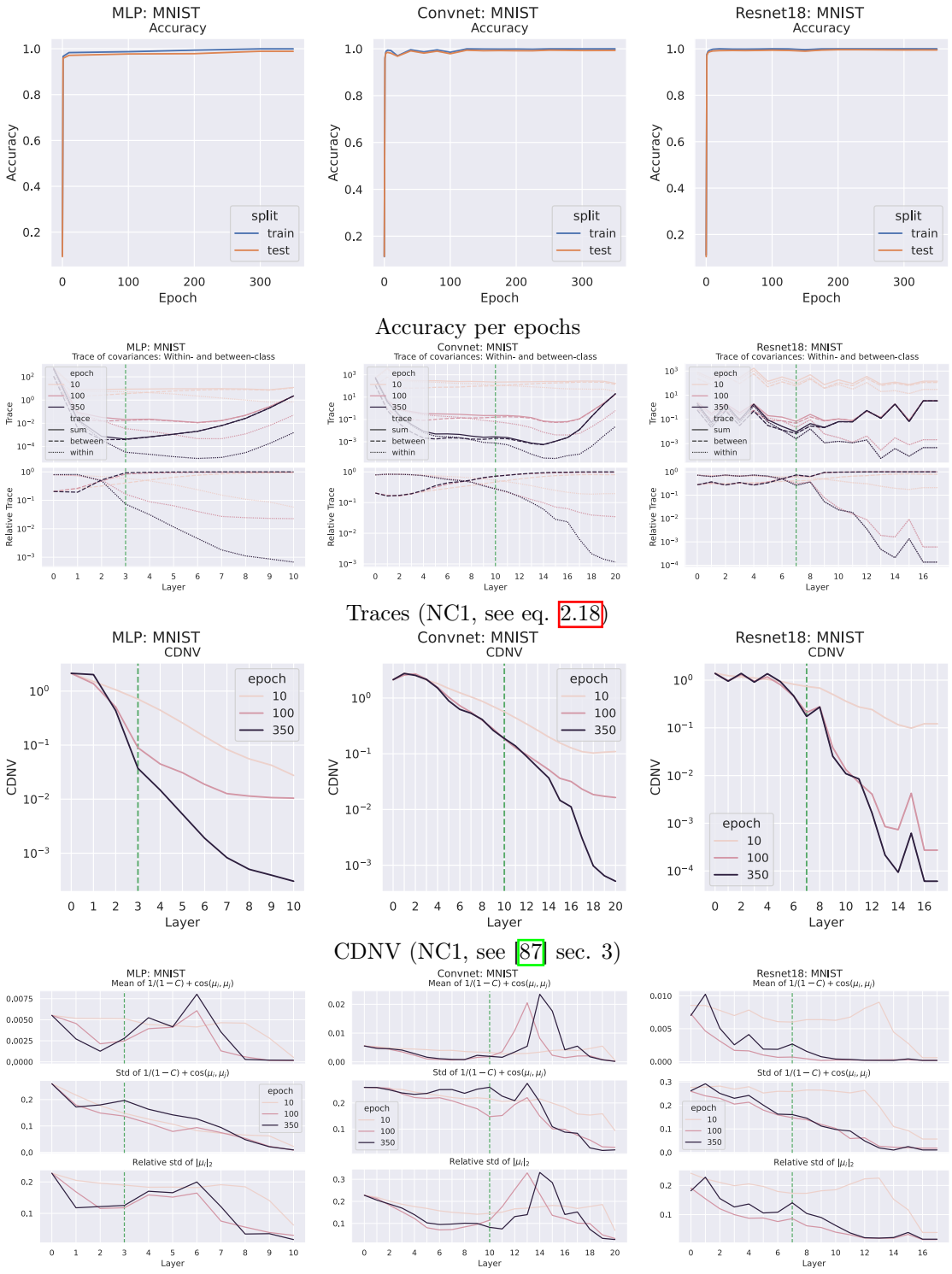


Figure B.0.3: Accuracy and NC1-2 on MLP-10, ConvNet-20 and Resnet50 trained with MNIST.

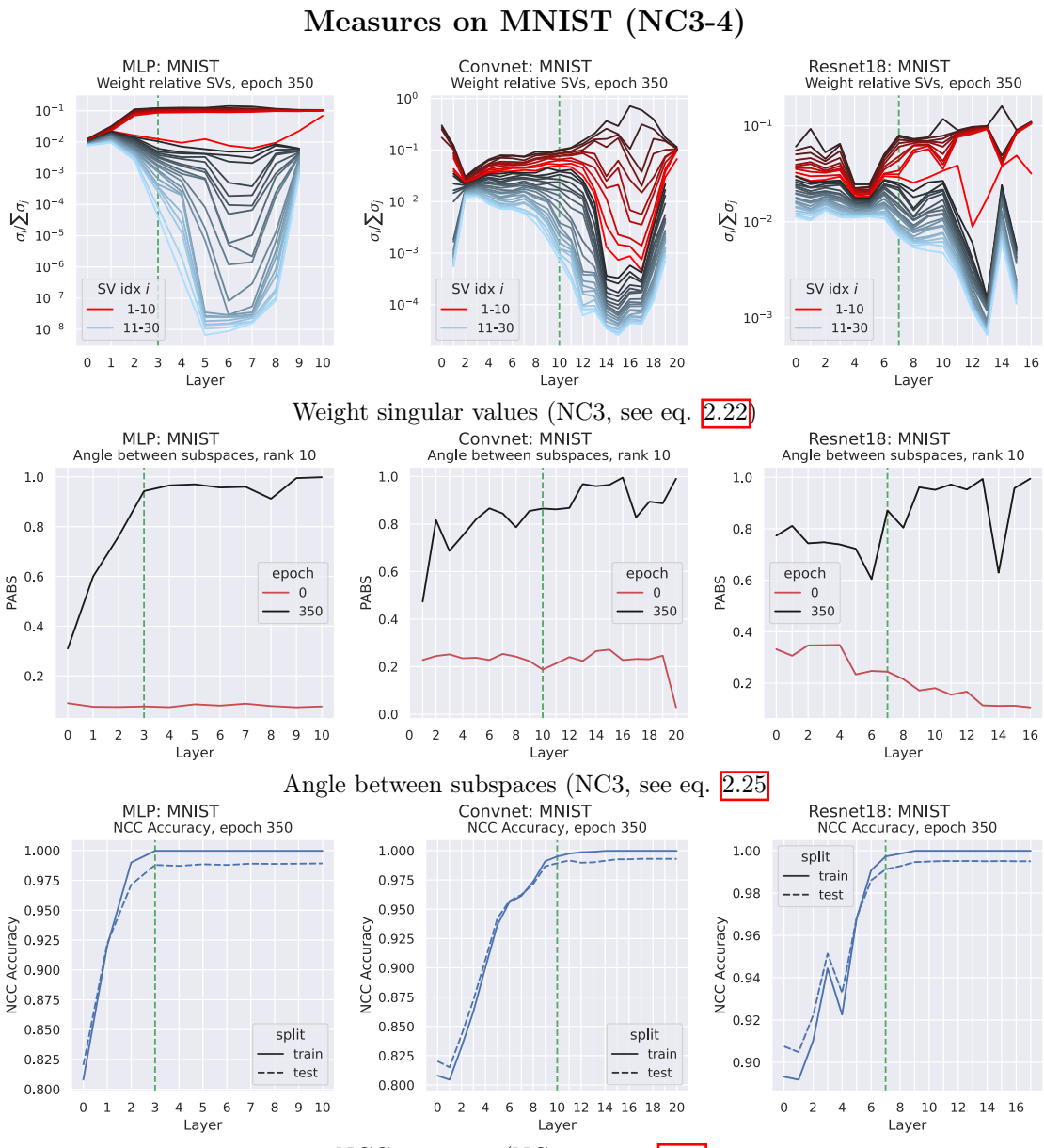


Figure B.0.4: NC3-4 on MLP-10, ConvNet-20 and Resnet50 trained with MNIST.

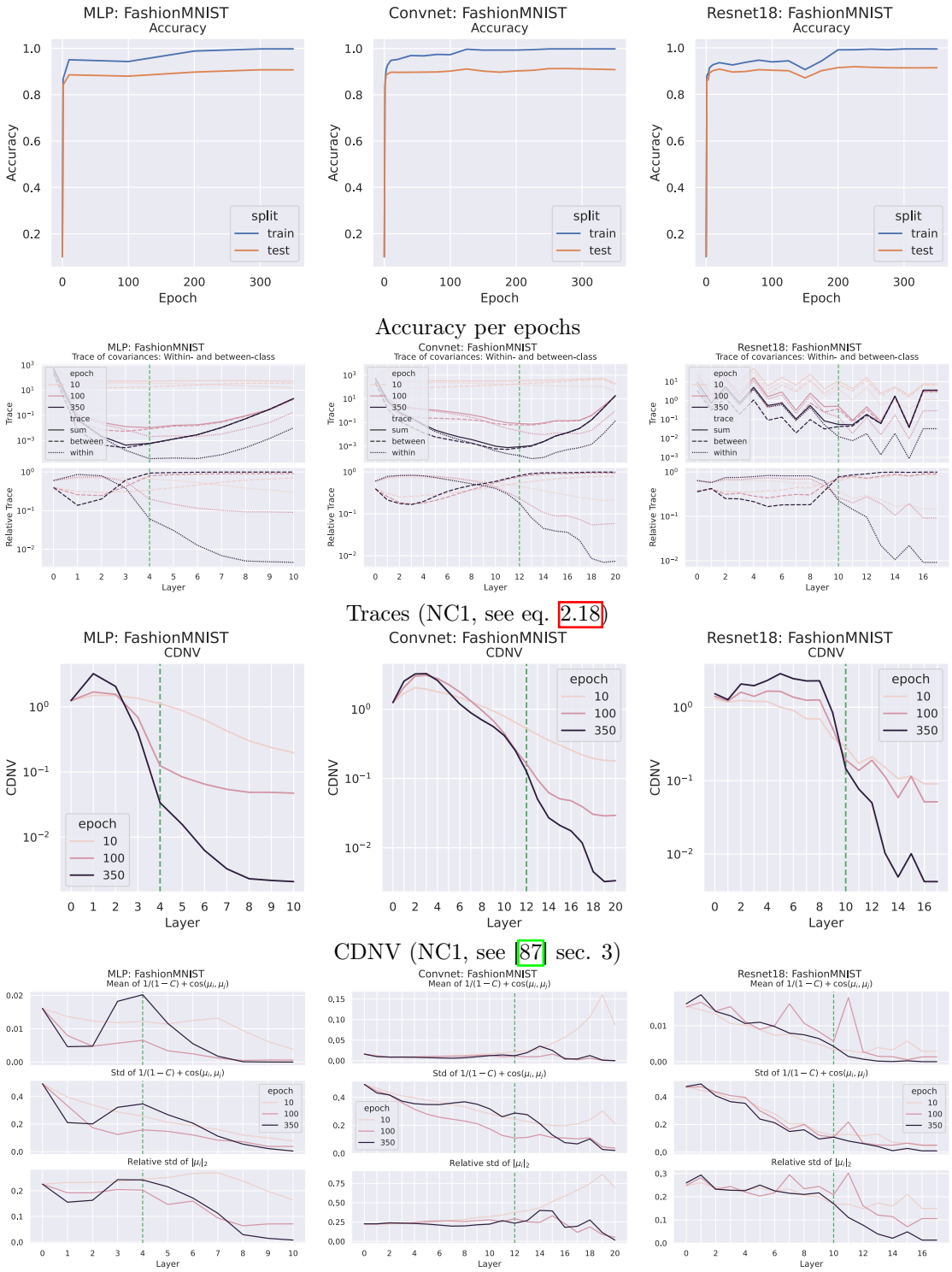


Figure B.0.5: Accuracy and NC1-2 on MLP-10, ConvNet-20 and Resnet50 trained with FashionMNIST.

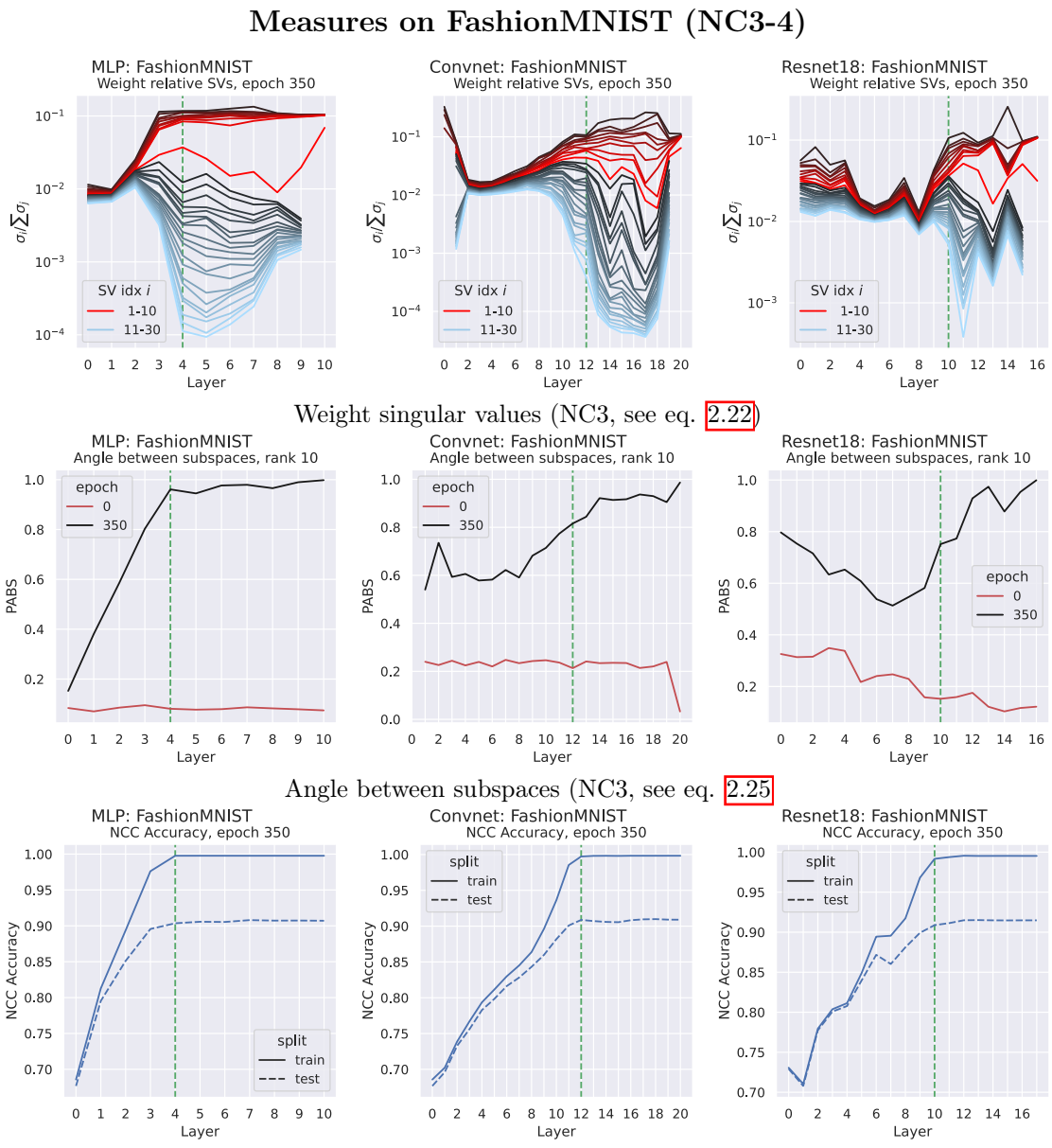


Figure B.0.6: NC3-4 on MLP-10, ConvNet-20 and Resnet50 trained with FashionMNIST.

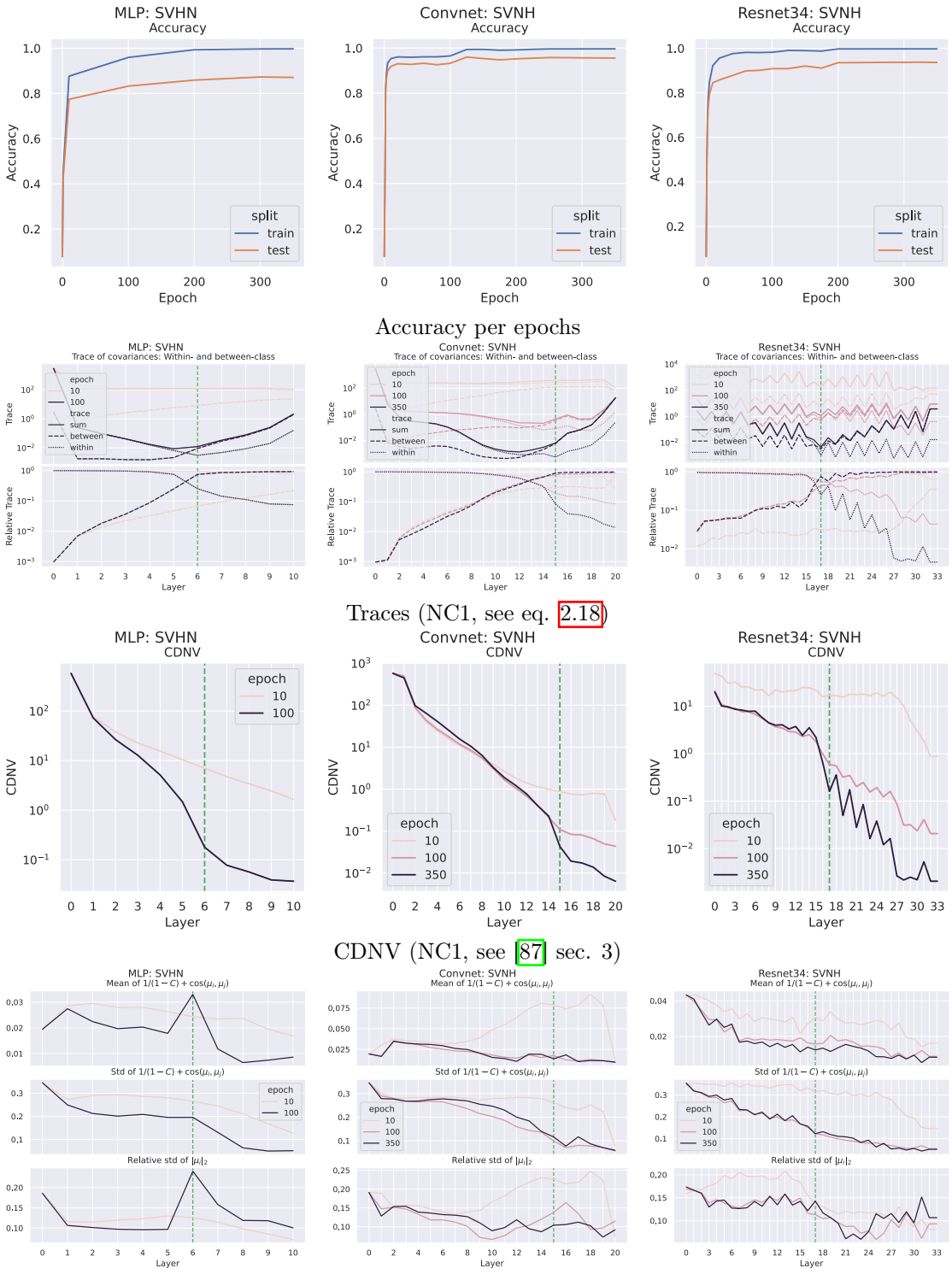


Figure B.0.7: Accuracy and NC1-2 on MLP-10, ConvNet-20 and Resnet50 trained with SVHN.

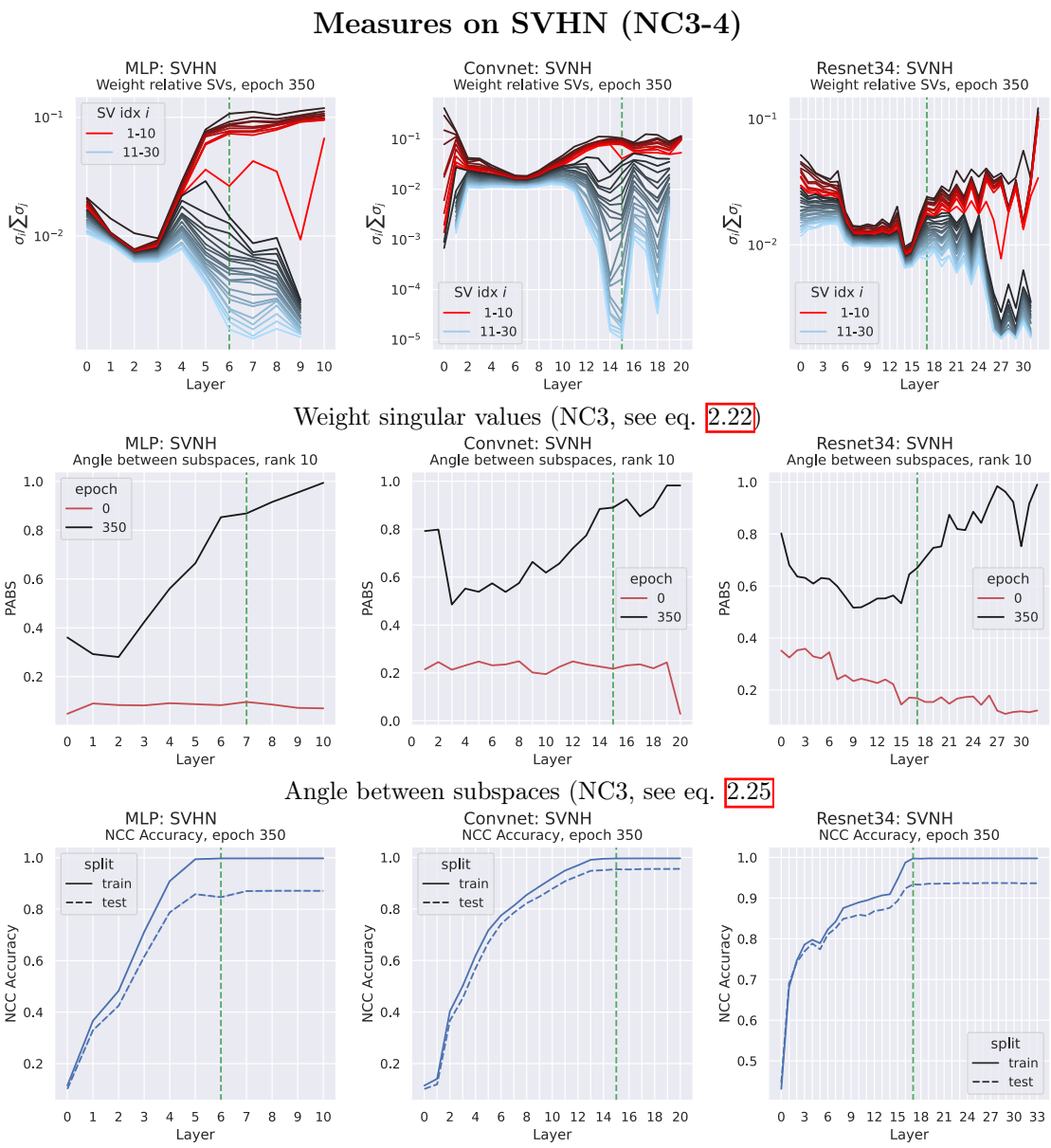


Figure B.0.8: NC3-4 on MLP-10, ConvNet-20 and Resnet50 trained with SVHN.

APPENDIX

C

FEATURE ANALYSIS

C.1 Further data on layer of Intermediate Neural Collapse from hyperparameters

These larger figures are larger and more readable versions of the ones presented in section 4.2, and additional data for Resnet50 models.

Note that models with lower than 0.35% accuracy are still not presented in the final results. The full data is available as part of the code given in section A.1.

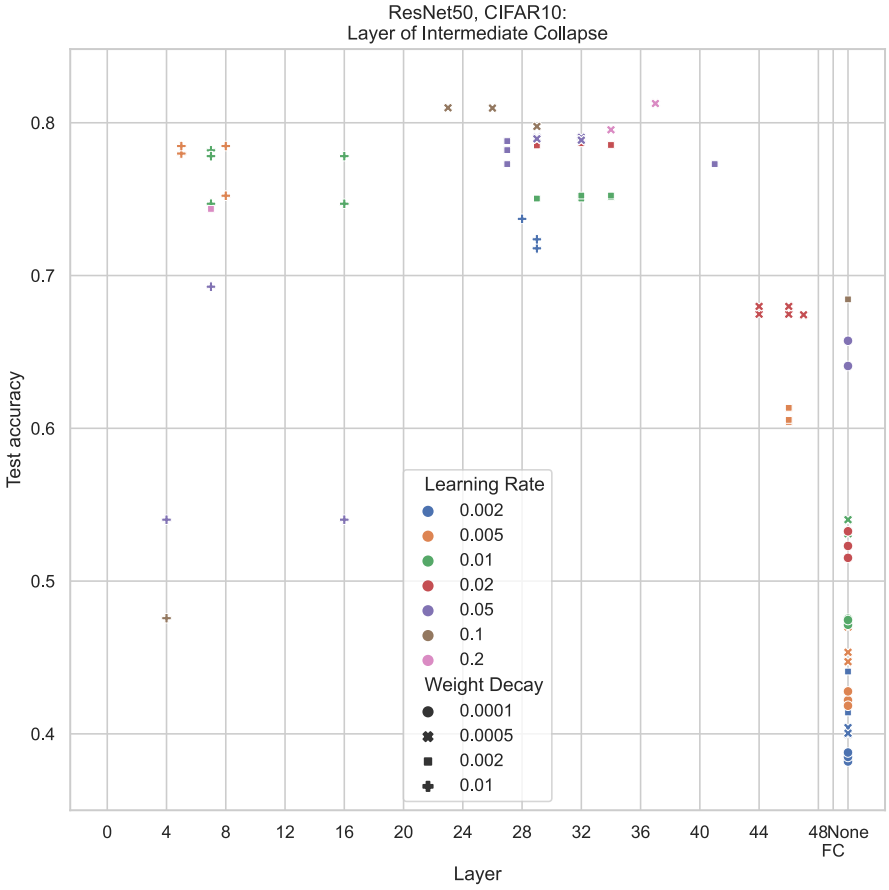


Figure C.1.1: INC for Resnet50 on CIFAR10 depending on hyperparameters.

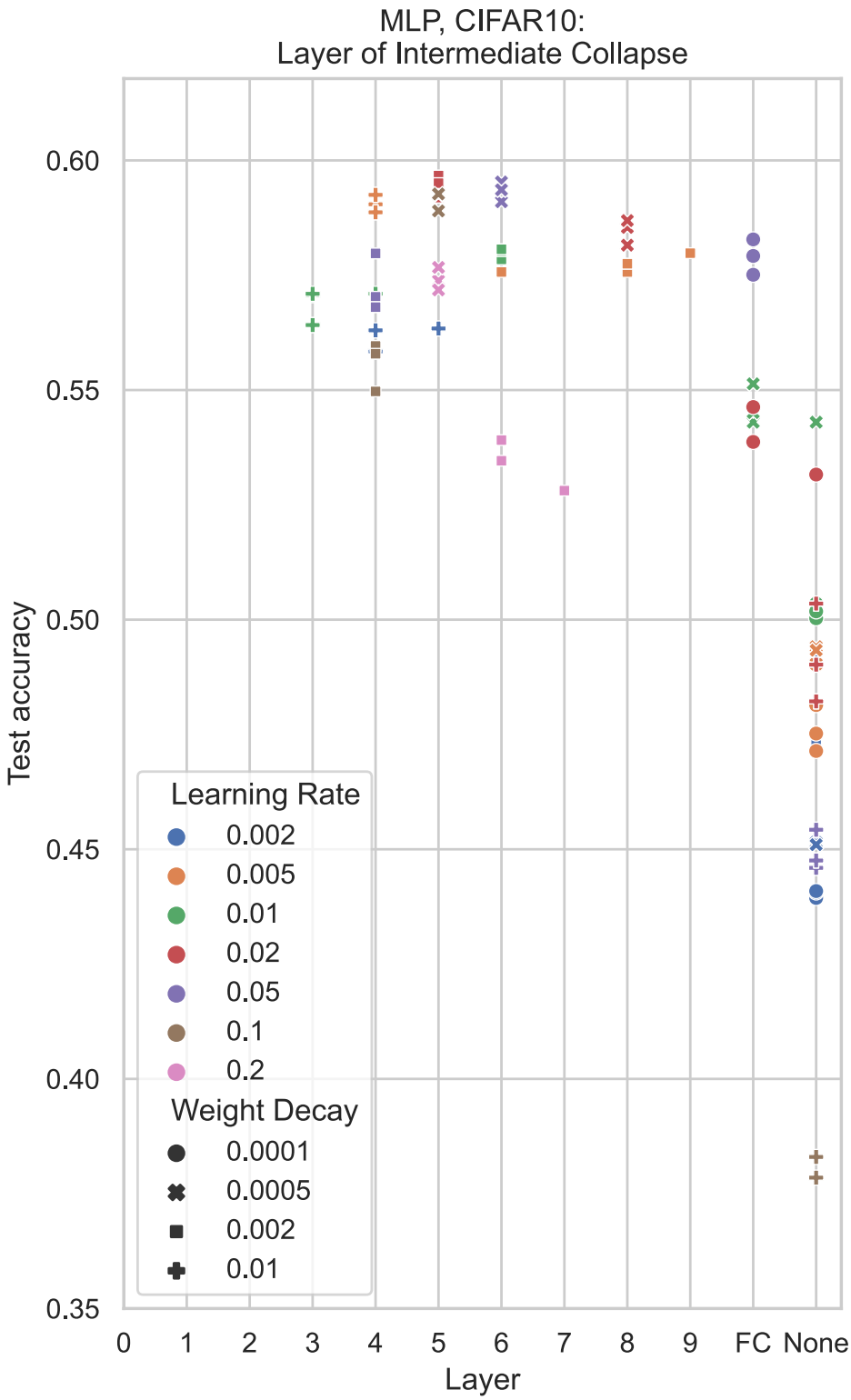


Figure C.1.2: INC for MLP on CIFAR10 depending on learning rate and weight decay.

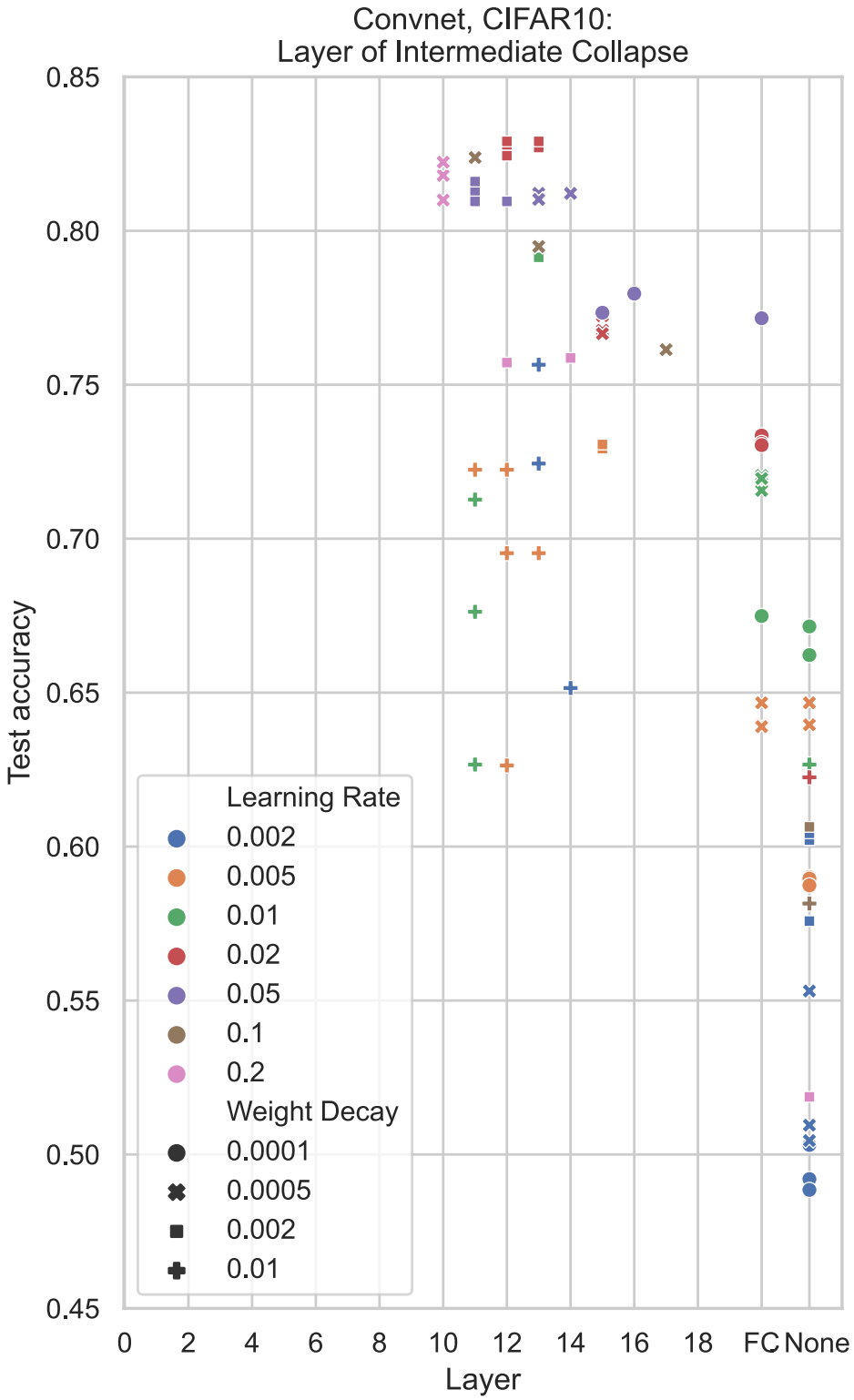


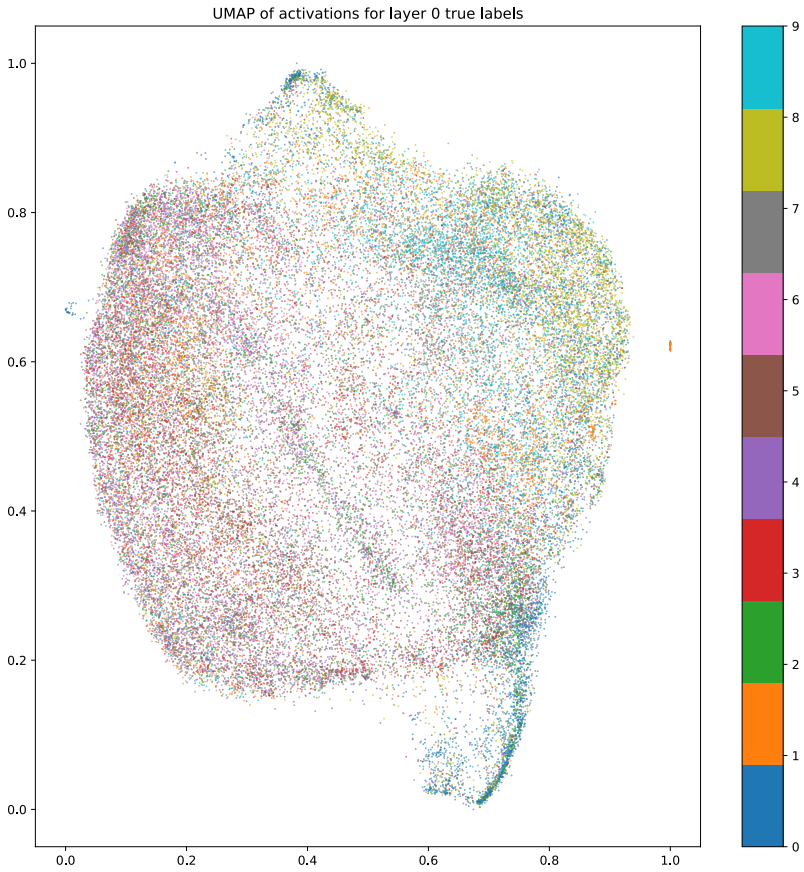
Figure C.1.3: INC for Convnet on CIFAR10 depending on learning rate and weight decay.

C.2 Plots of clusterings

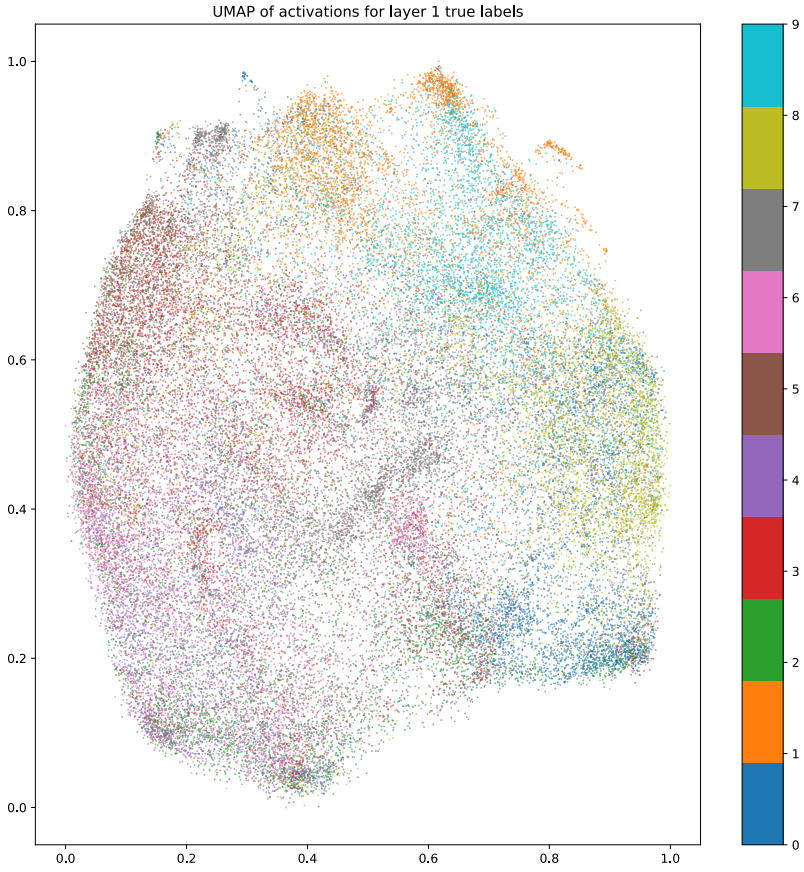
C.2.1 UMAP for true labels, enlarged figures

The following figure C.2.3 is the same figures as 4.3.1, but enlarged for readability.

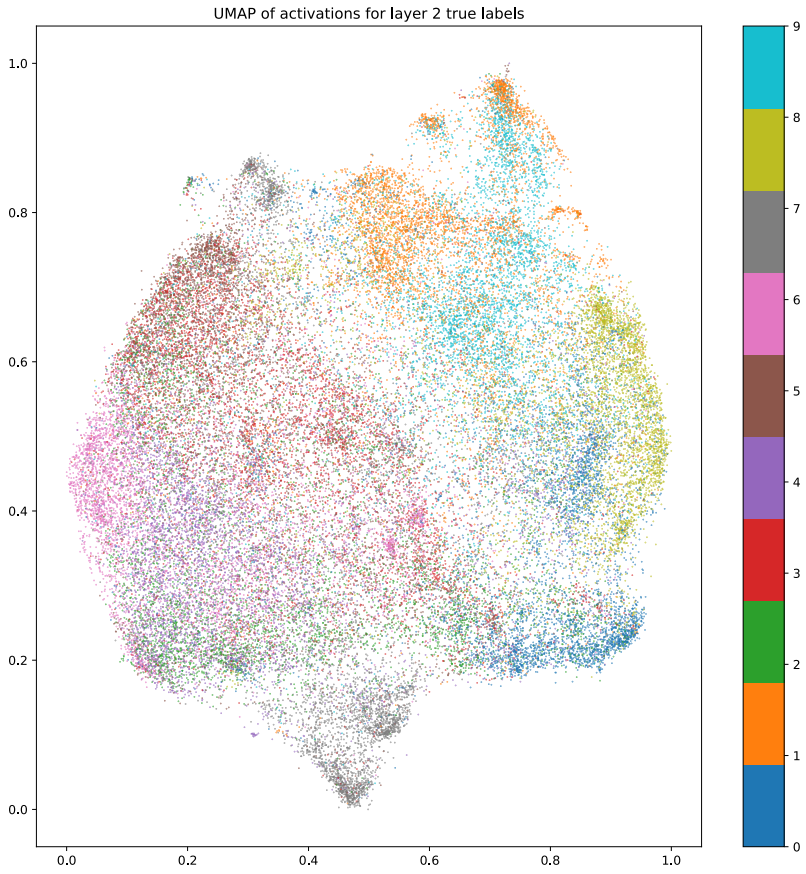
We also include figures where the coloring of the datapoints is based on the induced labels by the clustering algorithm as described in section 4.3. A second plot highlights the errors in classification. These two plots are shown stacked in figure C.2.4.



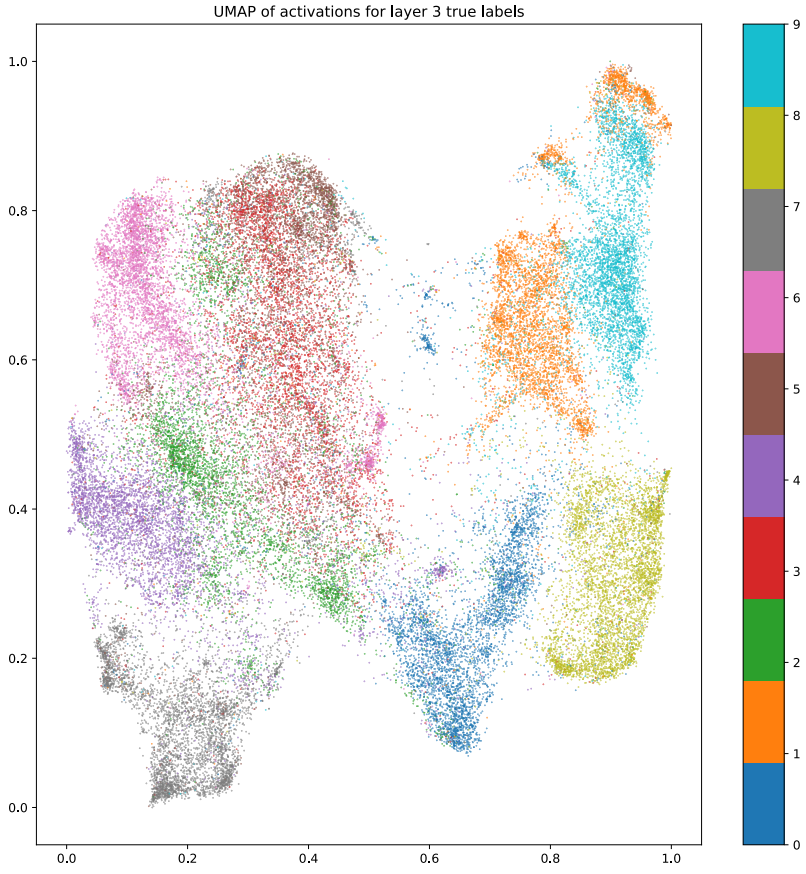
(a) UMAP clustering before layer 0



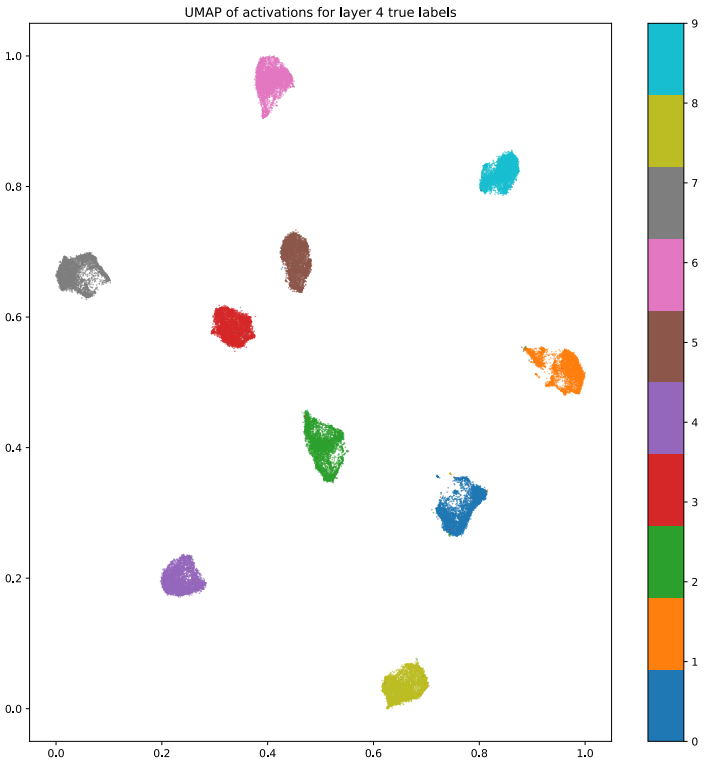
(b) UMAP clustering before layer 1



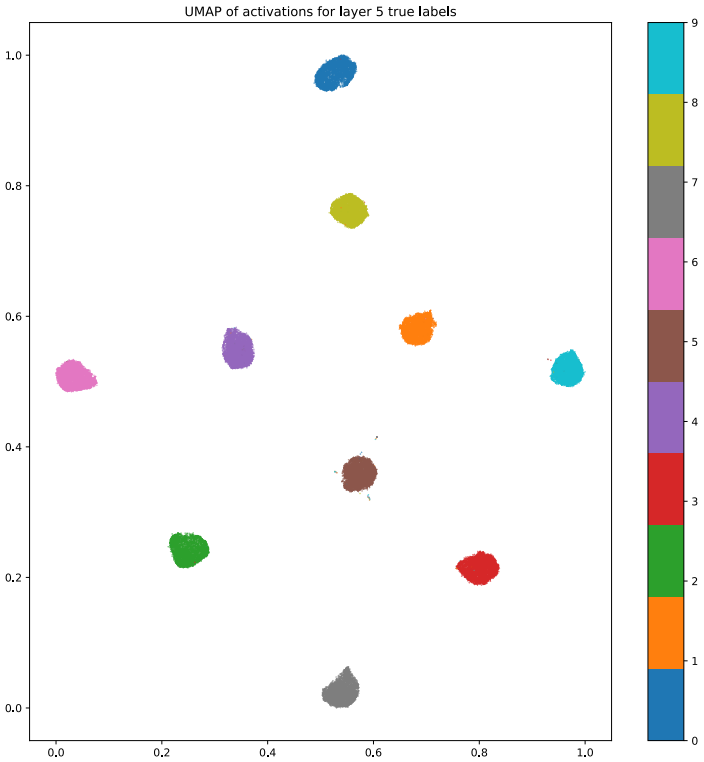
(a) UMAP clustering before layer 2



(b) UMAP clustering before layer 3

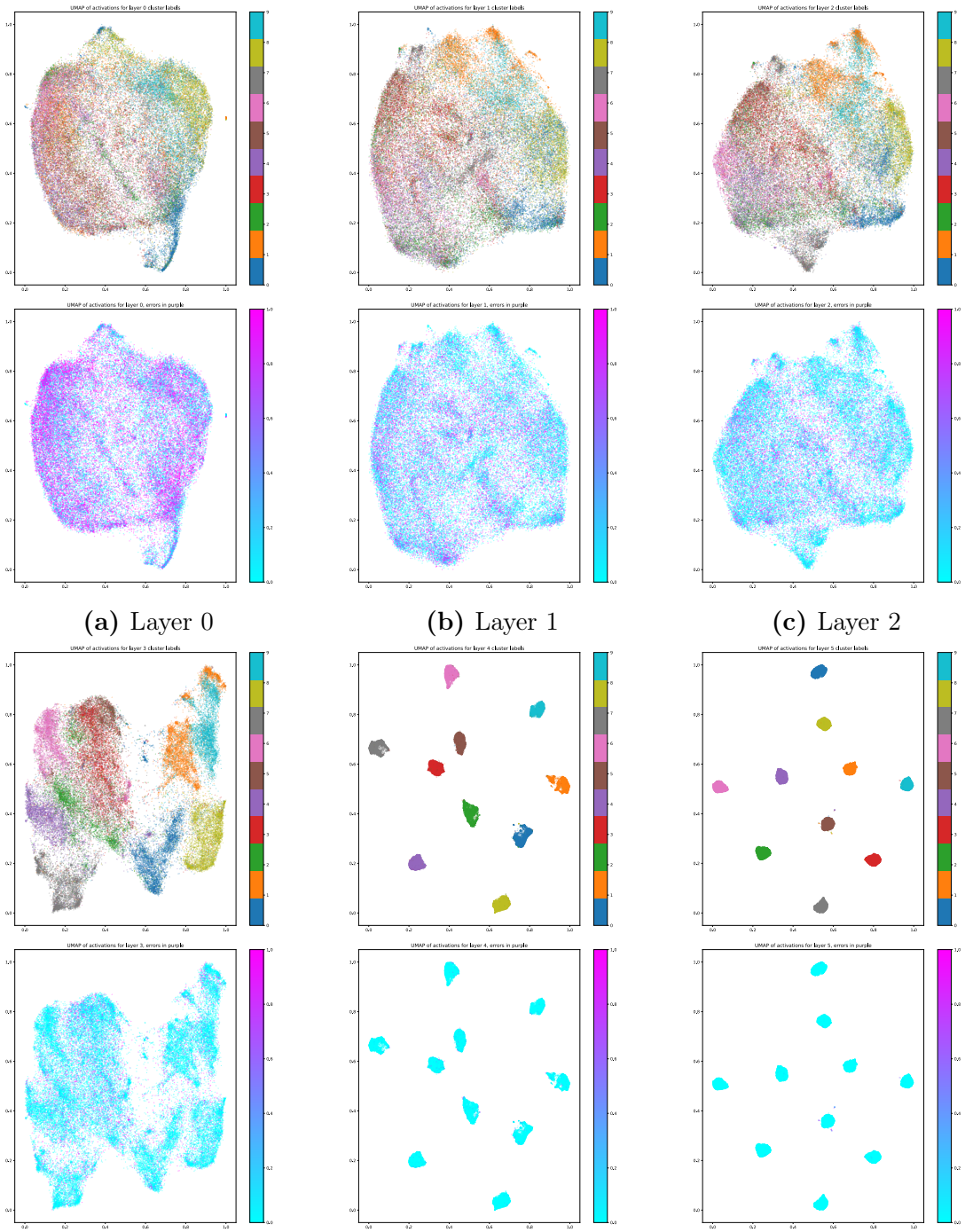


(a) UMAP clustering before layer 4



(b) UMAP clustering before layer 5

Figure C.2.3: UMAP clusterings of activations before the respective layers. See figure 4.3.1 for futher details.



(a) Layer 0 **(b)** Layer 1 **(c)** Layer 2

(d) Layer 3 **(e)** Layer 4 **(f)** Layer 5

Figure C.2.4: UMAP transformation with (upper) point-labelling by classification by clustering algorithm described in section 4.3, and (lower) erroneous classifications in purple, correct in blue. See also figure 4.3.1 for futher details.

Layer	Avg. #datapoints per cluster	Total Accuracy	Excluded datapoints
0	-	-	50,000
1	4.4	63%	13548
2	121	48%	0
3	1667	56%	0
4	3846	97.3%	0
5	5000	99.7%	6

Table C.2.1: Statistics on the clusterings before each layer using the euclidean distance metric.

C.2.2 Clusterings for euclidean metrics

This section contains all the same plots as displayed for the cosine-metric in the main section of the document. This is included for completeness, and to demonstrate that a change of metric is not hugely impactful and does in fact not give rise to any significant change in the results.

C.2.2.1 Euclidean metric clustering

The figures in this section are based on a clustering of the datapoints in activation-space with agglomerative clustering. The distance-metric used is the euclidean distance, ward linkage-method, and a distance threshold of 0.5. After this clustering, each cluster of 3 or more datapoints is given a label equal to the most prevalent class in that cluster. The clusters are then arranged based on their given label, and sorted within each label such that the cluster with the highest number of datapoints is listed first. The resulting confusion matrix from clusters to the element's true labels is plotted. See figure C.2.5.

Similarly defined as table 4.3.1, we have an accuracy for the euclidean metric clusters in C.2.1.

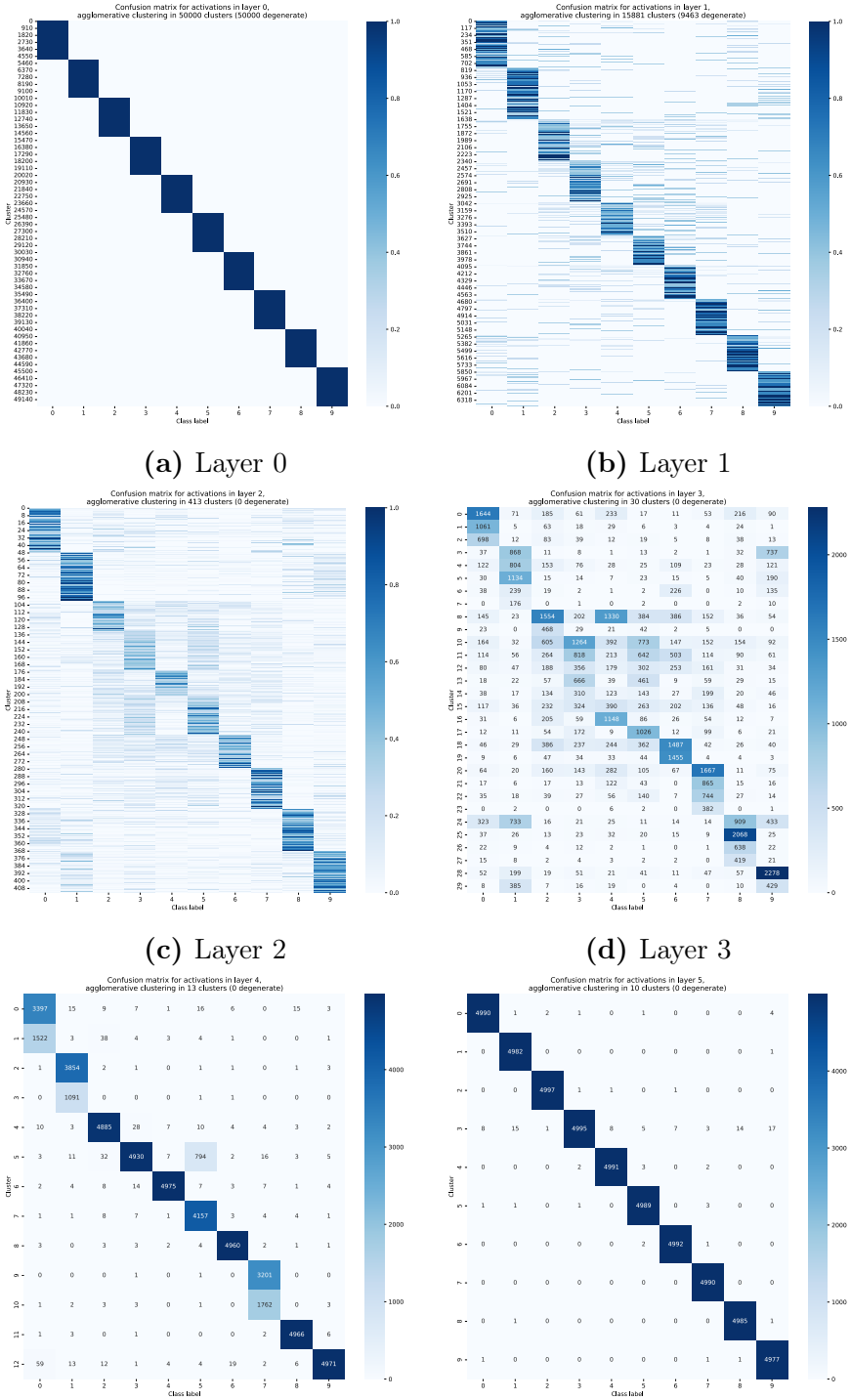
C.2.2.2 Euclidean metric errors in UMAP

We perform the same procedure as previously described in section 4.3 to obtain induced data-labels with the euclidean clustering.

C.2.2.3 Lack of hierarchical clustering

As in 4.3, we also report the extent to which the clusterings form hierarchical structures. The following table (C.2.2) reports the same numbers as table 4.3.2, but for the euclidean metric clustering.

Again we observe that there is little only very little hierarchical structure in the clustering of the data.

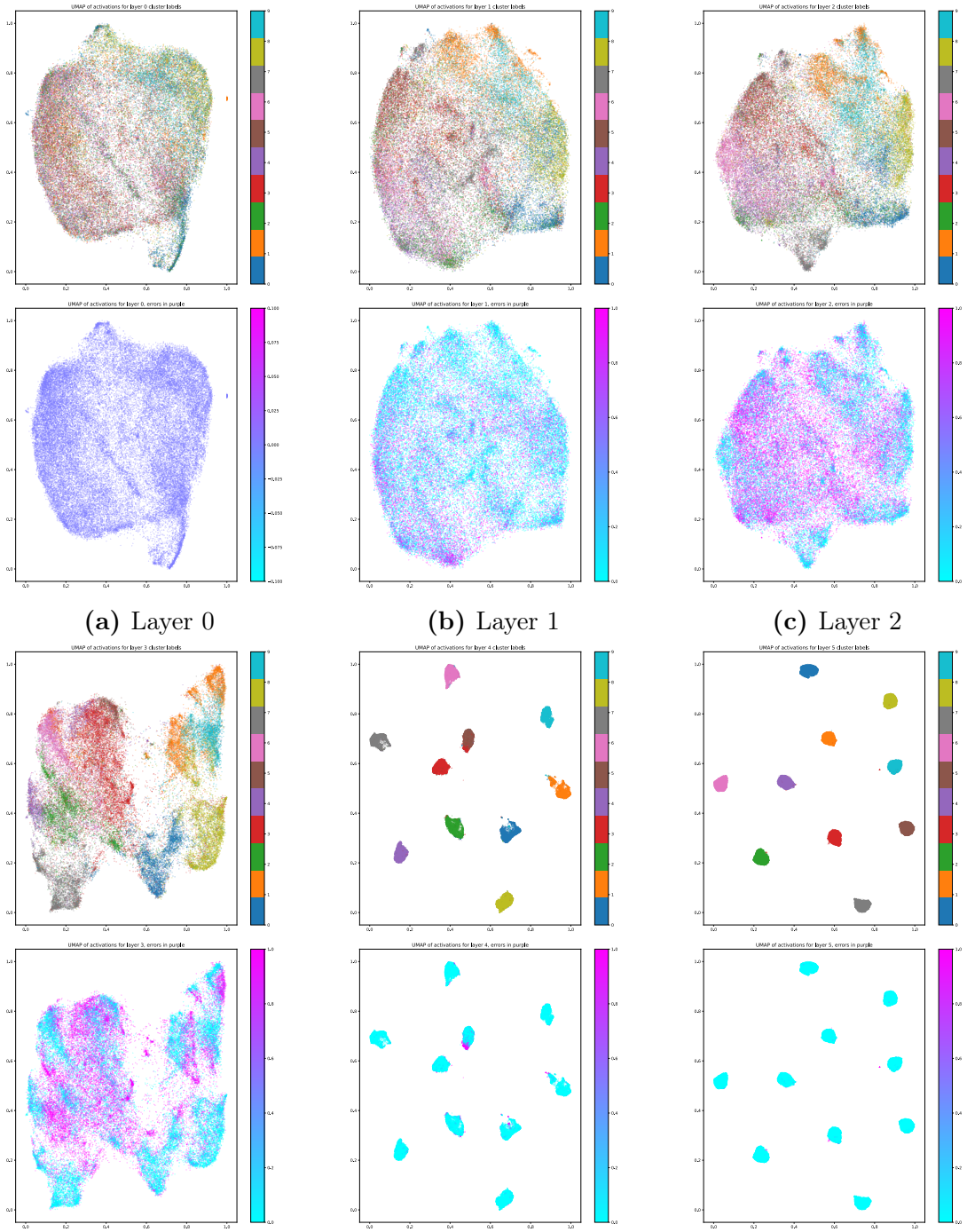


(a) Layer 0 **(b)** Layer 1

(c) Layer 2 **(d)** Layer 3

(e) Layer 4 **(f)** Layer 5

Figure C.2.5: Plots of clusters from agglomerative clustering with euclidean distance metric. Each row represents a cluster, and each column the class of the datapoints contained in that cluster. Each cluster (row) is normalized so it sums to 1, and as such the darker fields are the most prevalent in that cluster (row). See also figure 4.3.2.



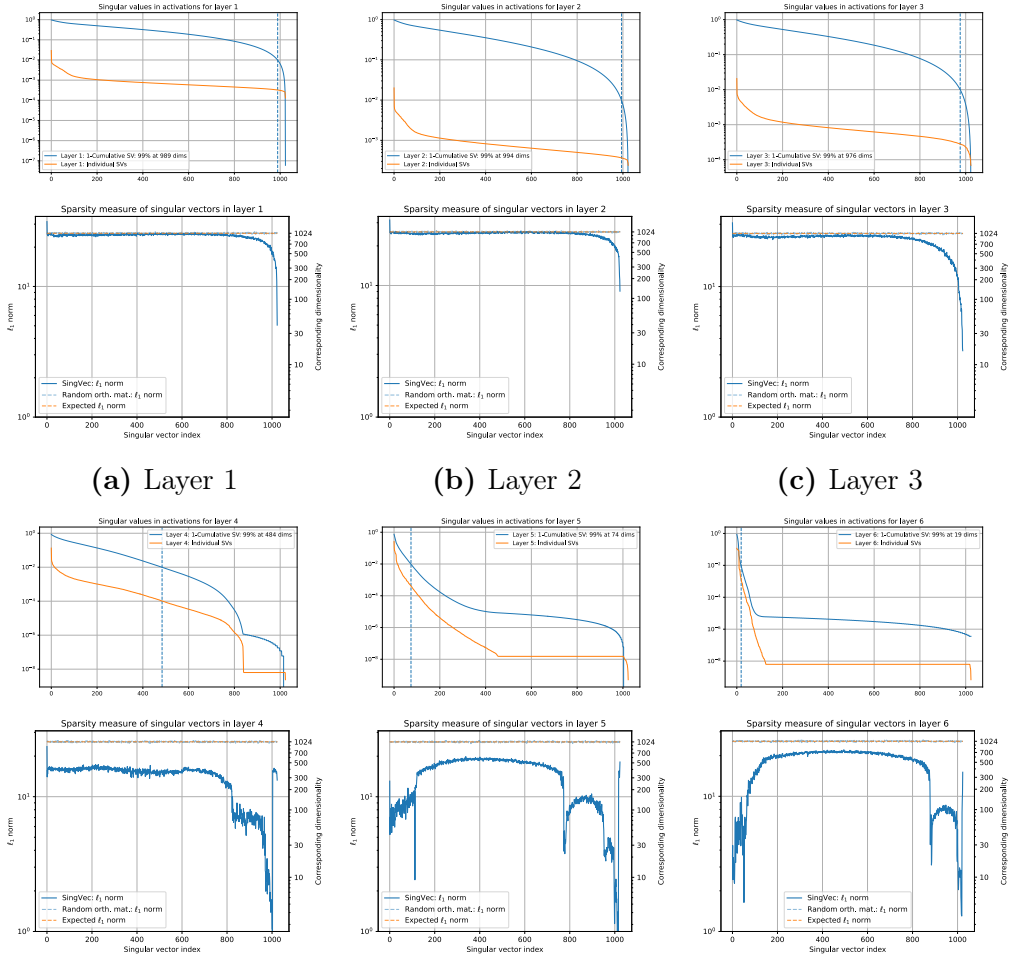
(a) Layer 0 **(b)** Layer 1 **(c)** Layer 2

(d) Layer 3 **(e)** Layer 4 **(f)** Layer 5

Figure C.2.6: UMAP transformation with (upper) point-labelling by classification by clustering algorithm described in section 4.3 *with euclidean distance as described in C.2.2*, and (lower) erroneous classifications in purple, correct in blue. Similar to figure C.2.4.

Layers	Hierarchical accuracy
0→ 1	-%
1→ 2	62%
2→ 3	42%
3→ 4	54%
4→ 5	97%

Table C.2.2: Accuracy of assuming hierarchical clusterings through each layer for euclidean metric.



(a) Layer 1 **(b)** Layer 2 **(c)** Layer 3

(d) Layer 4 **(e)** Layer 5 **(f)** Layer 6

Figure C.2.7: Singular values of activations and sparsity of corresponding singular vectors for layer 1-6. An extension of figure 4.4.1 and 4.4.2.

C.2.3 Singular vectors and ℓ_1 sparsity

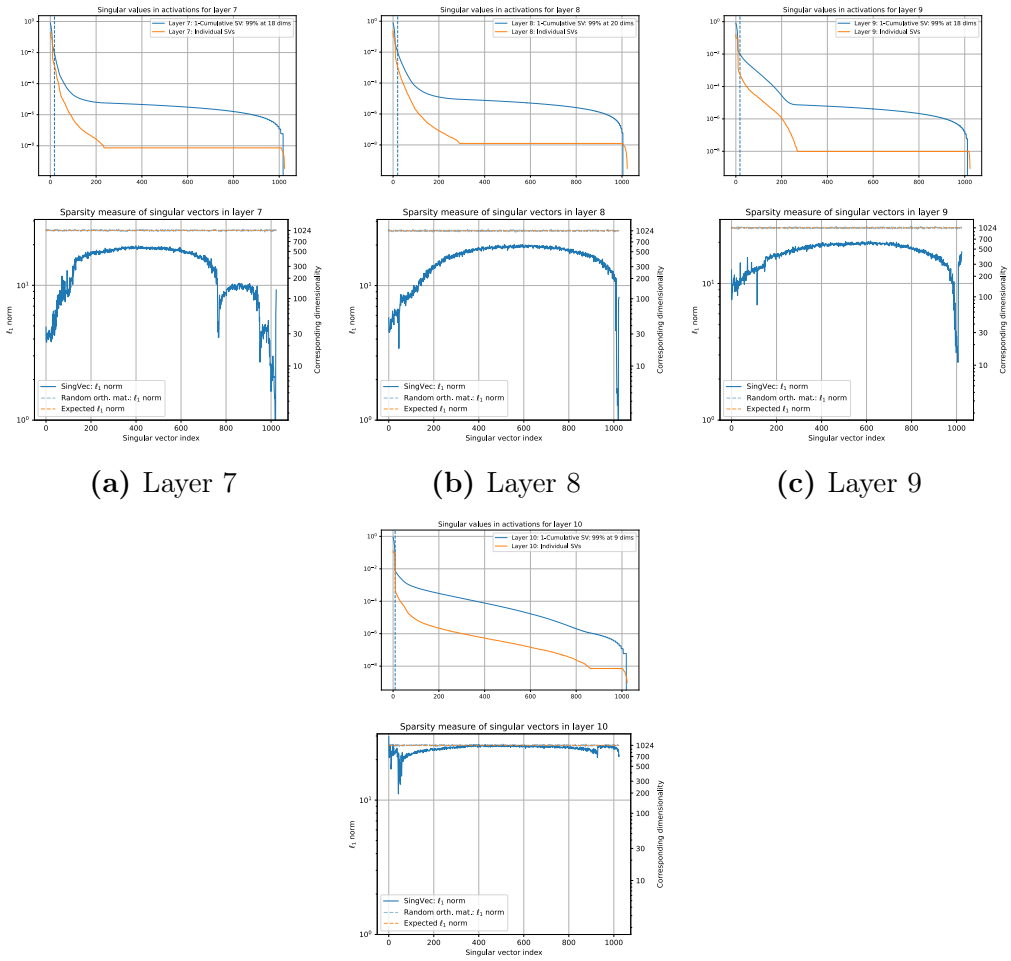
The full plots of the singular values of the covariance matrix of the data (top) as well as the sparsity of the corresponding singular vectors (bottome) are given in figures C.2.7 and C.2.8.

C.2.4 Sparsity and Means of Neurons

In this section we give all plots from all relevant layers 1-10 corresponding to figures 4.4.3 and 4.4.4. In this section, they are are arranged to highlight the correspondence between different paradigms of sparsity and different paradigms of mean-activations.

We also note a few observations from these plots not mentioned in the main text for brevity:

- In the post-collapse layers (4-10), there seems to be a clear pattern where the neurons exhibiting sparsity have orders of magnitude larger activations than the neurons not exhibiting sparsity.



(a) Layer 7 **(b)** Layer 8 **(c)** Layer 9

(d) Layer 10

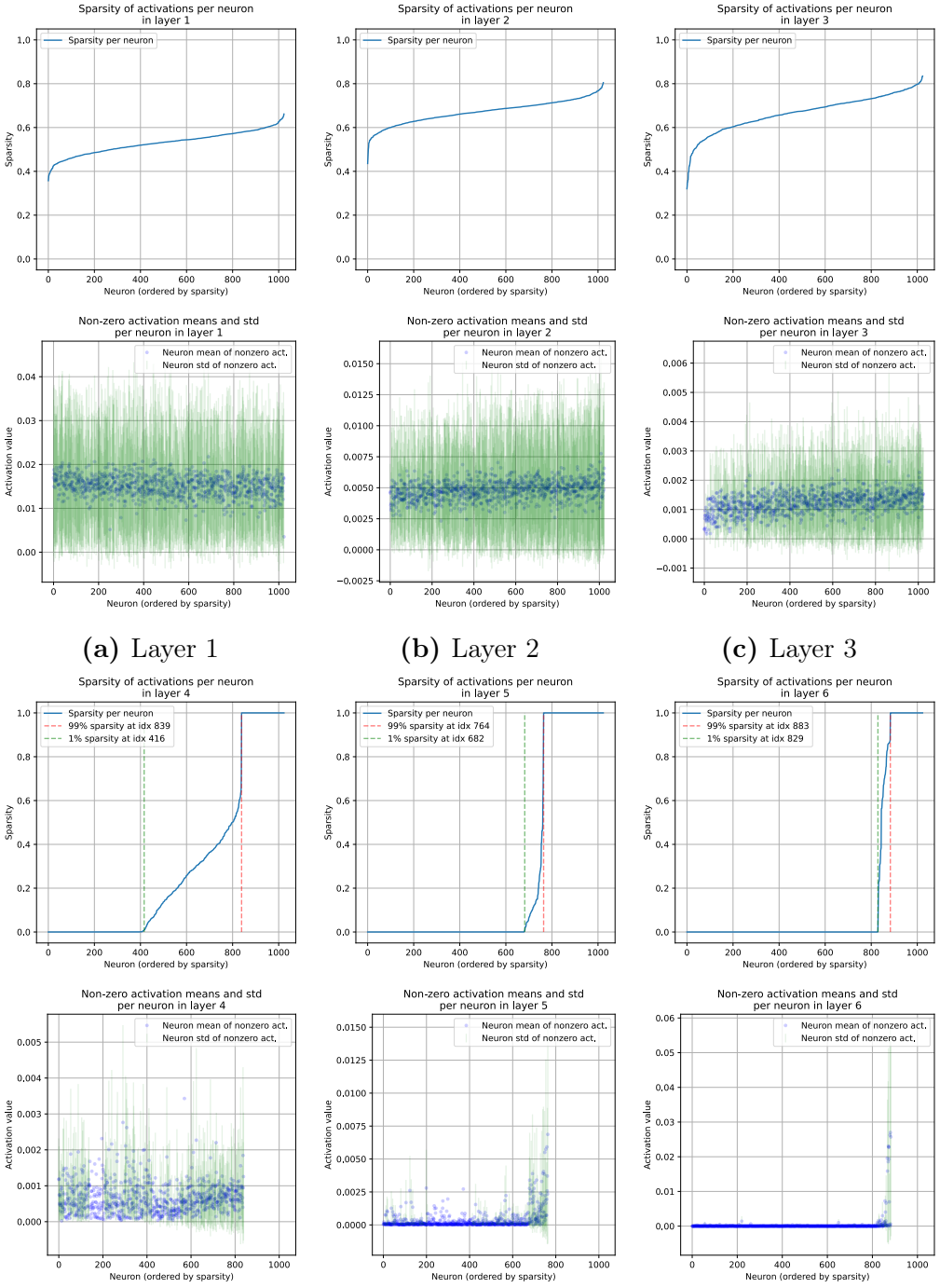
Figure C.2.8: Singular values of activations and sparsity of corresponding singular vectors for layer 7-10. An extension of figure 4.4.1 and 4.4.2.

- While layer 4 exhibits patters similar to the later layers, it seems that the first layer post-collapse is not as perfectly collapsed as the deeper layers. This fits with previous observations, showing how the collapse becomes stronger deeper in the network.

- In layer 10 (C.2.10d), and to some degree in layer 6-9, there is clear step-wise sparsity with steps at the $\frac{1}{10}$-marks. This likely corresponds to the only sparsity arising from the 10 class means forming essentially only 10 sets of internally equivalent datapoints.

- Seeing as this step-wise nature of the datapoints has long plateus (see e.g. neuron 200-750 in fig C.2.10d), it is clear that the class-mean vectors are not alligned with the neuron-axes but rather span some other 10-dimensional space with little preference for the canonical axes.

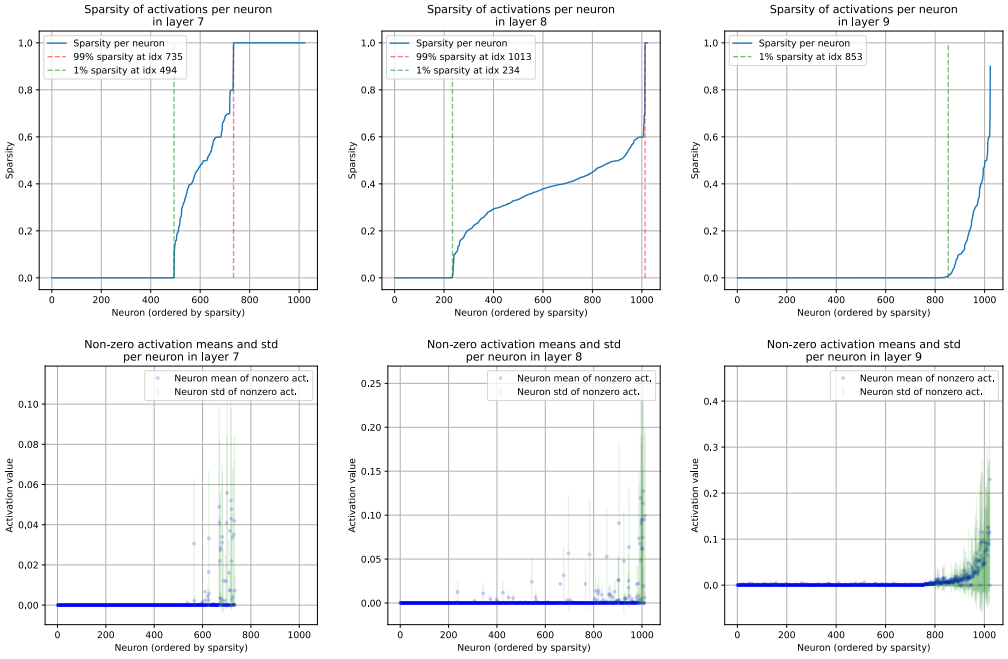
We also note, as mentioned in the main text, the stark contrast between pre- and post-collapse layers, and how the pre-collapse layers clearly occupy the full space.



(a) Layer 1 **(b)** Layer 2 **(c)** Layer 3

(d) Layer 4 **(e)** Layer 5 **(f)** Layer 6

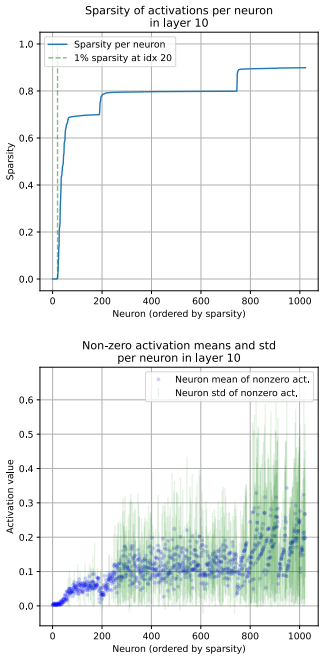
Figure C.2.9: Sparsity of neurons and means of all non-zero activations in hidden layers 1-6. An extension of figure 4.4.3 and 4.4.4.



(a) Layer 7

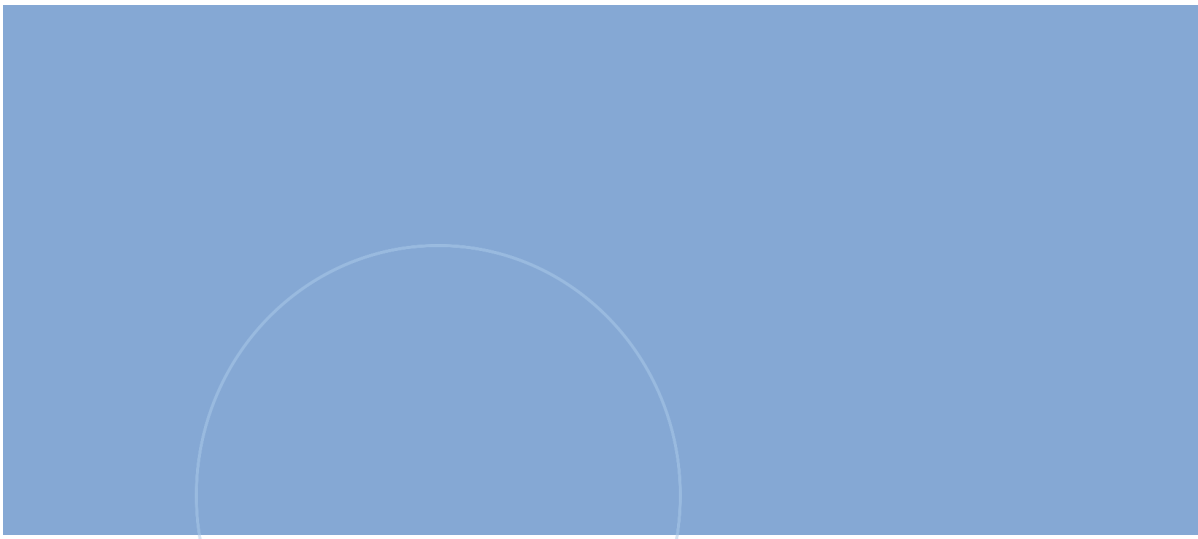
(b) Layer 8

(c) Layer 9



(d) Layer 10

Figure C.2.10: Sparsity of neurons and means of all non-zero activations in hidden layers 7-10. An extension of figure 4.4.3 and 4.4.4.



NTNU

Kunnskap for en bedre verden