ON THE IMPLICIT BIAS TOWARDS DEPTH MINIMIZATION IN DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent results in the literature suggest that the penultimate (second-to-last) layer representations of neural networks that are trained for classification exhibit a clustering property called neural collapse (NC). We study the implicit bias of stochastic gradient descent (SGD) in favor of low-depth solutions when training deep neural networks. We characterize a notion of effective depth that measures the first layer for which sample embeddings are separable using the nearest-class center classifier. Furthermore, we hypothesize and empirically show that SGD implicitly selects neural networks of small effective depths.

Secondly, while neural collapse emerges even when generalization should be impossible - we argue that the *degree of separability* in the intermediate layers is related to generalization. We derive a generalization bound based on comparing the effective depth of the network with the minimal depth required to fit the same dataset with partially corrupted labels. Remarkably, this bound provides non-trivial estimations of the test performance. Finally, we empirically show that the effective depth of a trained neural network monotonically increases when increasing the number of random labels in data.

1 INTRODUCTION

Deep learning systems have steadily advanced the state of the art in a wide range of benchmarks, demonstrating impressive performance in tasks ranging from image classification (Taigman et al., 2014; Zhai et al., 2021), language processing (Devlin et al., 2019; Brown et al., 2020), open-ended environments (Silver et al., 2016; Arulkumaran et al., 2019), to coding (Chen et al., 2021).

Recent findings show that deep neural networks can generalize well even when the number of parameters far exceeds the number of training samples (Zhang et al., 2017; Belkin, 2021). While it has been repetitively observed that training deeper networks achieves superior performance over their shallow counterparts (He et al., 2016; 2015; Wang et al., 2022), an effective theory for explaining the success of deep neural networks is still missing.

Traditional approaches for measuring generalization (Vapnik, 1998; Shalev-Shwartz & Ben-David, 2014; Mohri et al., 2012) typically bound the test error by the sum between the train error and the ratio between a complexity measure of the selected hypothesis class (e.g., neural network) and $\sqrt{m}$, where m is the number of training samples. For instance, the complexity may depend on the number of trainable parameters (Vapnik, 1998), their norms (e.g., (Bartlett et al., 2017; Golowich et al., 2017)) or the rank of the trained matrices (e.g., (Bartlett et al., 2017)). In many practical settings, however, the complexity of the learned hypothesis typically exceeds $\sqrt{m}$ making the bounds vacuous (Bartlett & Mendelson, 2001; Harvey et al., 2017; Neyshabur et al., 2015; Bartlett et al., 2017; Neyshabur et al., 2018).

A recent thread in the literature suggests that SGD exhibits an 'implicit regularization' during optimization, and that this may be key to generalization in deep learning (Neyshabur, 2017). For instance, (Belkin, 2021; Ali et al., 2019; Gunasekar et al., 2018) showed that gradient-based optimization implicitly minimizes the weight norms of linear models. Other papers (e.g., (Galanti & Poggio, 2022; Timor et al., 2022; Le & Jegelka, 2022)) demonstrated that when training neural networks, gradient-based optimization methods implicitly minimize the rank of the learned weight matrices. While these biases may be related to the performance of deep neural networks, it is un-

clear how to connect these results with traditional generalization bounds in a way that leads to non-vacuous estimations of test performance.

Contributions. In this paper, we propose a novel method for measuring generalization in deep learning. We propose a new type of generalization bound that is not based on comparing the trained model's complexity to the dataset's size. Our bound ensures that the model performs well at test time if its complexity is small *compared* to the complexity of a network required to fit the same dataset with partially random labels. In other words, even if a trained network has a complexity greater than $\sqrt{m}$, it may be less complex than a model that fits partially random labels. As a result, in such cases, our bound may still provide a non-trivial estimate of the test error.

To formally describe our notion of complexity for neural networks, we employ the notion of nearest class-center (NCC) separability. This property asserts that the feature embeddings associated with training samples belonging to the same class are separable according to the nearest class-center decision rule. While original results (Papyan et al., 2020) observed NCC separability at the penultimate layer of trained networks, recent results (Ben-Shaul & Dekel, 2022) observed NCC separability in intermediate layers. In this work, we introduce the notion of *'effective depth'* of neural networks that regards to the lowest layer for which its features are NCC separable (see Sec. 3.2).

We make multiple important observations regarding effective depths. **(i)** We empirically show that the effective depth of trained networks monotonically increases when increasing the amount of random labels in data. **(ii)** We observe that when training sufficiently deep networks, they converge to (approximately) the same effective depth L_0, i.e., regardless of the network's depth L, the feature embeddings of layers above layer L_0 tend to be NCC separable. Based on the first observation, we argue that the proposed bound provides non-trivial estimations of the test performance. Second, unlike traditional generalization bounds, the bound is empirically independent of depth due to the "Minimal Depth" observation. We empirically compare the proposed bound to baseline bounds in Tab. 2 and show that while other bounds are vacuous in the deep learning setting, the notion proposed overcomes this issue.

1.1 Related Work

Neural collapse and generalization. Our work is closely related to the recent line of work on Neural collapse (Papyan et al., 2020; Han et al., 2022). Neural collapse identifies training dynamics of deep networks for standard classification tasks, where the feature embeddings associated with training samples belonging to the same class tend to concentrate around their means.

While several papers analyzed the emergence of neural collapse from a theoretical standpoint (e.g., (Zhu et al., 2021; Rangamani et al., 2022; Lu & Steinerberger, 2020; Fang et al., 2021; Ergen & Pilanci, 2021)), its specific role in deep learning and its potential relationship with generalization is still unclear. Recent work (Galanti et al., 2022a; Xu et al., 2022; Galanti et al., 2022b) studied the conditions for when class features variation collapse generalizes from the train samples, to both test samples and new classes and in the transfer learning setting. Following that, Hui et al. (2022) further studied whether neural collapse generalizes to test samples.

In this work we focus on the following (independent) question: *is neural collapse a good indication of whether the network generalizes well?* As a counter argument, Mixon et al. (2020) provided empirical evidence that neural collapse emerges even when training the network with random labels. Therefore, the presence of neural collapse cannot indicate whether the network generalizes or not. However, this experiment does not invalidate the possibility of an indirect relationship between neural collapse and generalization. We argue that the degree of separability in the intermediate layers may be closely related to generalization.

Emergence of structure in deep networks. While various papers Papyan (2020); Tirer & Bruna (2022); Galanti et al. (2022a); Ben-Shaul & Dekel (2022); Cohen et al. (2018); Alain & Bengio (2017); Montavon et al. (2011); Papyan et al. (2017); Ben-Shaul & Dekel (2021); Shwartz-Ziv & Tishby (2017) investigated certain geometrical properties within intermediate layers (e.g., clustering and separability), this paper is the first to demonstrate that deep neural networks tend to converge to a minimal effective depth that is independent of the network's depth. Even though one can derive "effective depths" from the experiments of Cohen et al. (2018), we show that when training sufficiently deep networks they converge to (approximately) the same effective depth.

2 PROBLEM SETUP

We consider the problem of training a model for standard multi-class classification. Formally, the target task is defined by a distribution P over samples $(x, y) \in \mathcal{X} \times \mathcal{Y}_C$, where $\mathcal{X} \subset \mathbb{R}^d$ is the instance space, and $\mathcal{Y}_C$ is a label space with cardinality C. To simplify the presentation, we use one-hot encoding for the label space, that is, the labels are represented by the unit vectors in $\mathbb{R}^C$, and $\mathcal{Y}_C := \{e_c : c = 1, \ldots, C\}$ where $e_c \in \mathbb{R}^C$ is the cth standard unit vector in $\mathbb{R}^C$; with a slight abuse of notation, we allow ourselves to write $y = c$ instead of $y = e_c$. For a pair (x, y) distributed by P, we denote by P_c the class conditional distribution of x given $y = c$ (i.e., $P_c(\cdot) := \mathbb{P}[x \in \cdot \mid y = c]$).

A classifier $h_W : \mathcal{X} \to \mathbb{R}^C$ assigns a *soft* label to an input point $x \in \mathcal{X}$, and its performance on the distribution P is measured by the expected risk

$$L_P(h_W) := \mathbb{E}_{(x,y(x))\sim P}[\ell(h_W(x), y(x))],$$

where $\ell : \mathbb{R}^C \times \mathcal{Y}_C \to [0, \infty)$ is a non-negative loss function (e.g., L_2 or cross-entropy losses).

We typically do not have direct access to the full population distribution P. Therefore, we generally aim to learn a classifier, h, using some balanced training data $S := \{(x_i, y_i)\}_{i=1}^m = \cup_{c=1}^C S_c = \cup_{c=1}^C \{x_{ci}, y_{ci}\}_{i=1}^{m_0} \sim P_B(m)$ of $m = C \cdot m_0$ samples consisting m_0 independent and identically distributed (i.i.d.) samples drawn from P_c for each $c \in [C]$. Specifically, we intend to find W that minimizes the regularized empirical risk

$$L_S^\lambda(h_W) \;:=\; \tfrac{1}{m} \sum_{i=1}^{m} \ell(h_W(x_i), y_i) + \lambda \|W\|_2^2, \tag{1}$$

where the regularization controls the complexity of the function h_W and typically helps reducing overfitting. Finally, the performance of the trained model is evaluated using the train and test error rates, which are computed as follows: $\mathrm{err}_S(h_W) := \sum_{i=1}^m \mathbb{I}[\arg\max_c h_W(x_i)_c \neq y_i]$ and $\mathrm{err}_P(h_W) := \mathbb{E}_{(x,y)\sim P}[\mathbb{I}[\arg\max_c h_W(x)_c \neq y]]$. Here, $\mathbb{I} : \{\text{True}, \text{False}\} \to \{0, 1\}$ the indicator function.

Neural networks. In this work, the classifier h_W is a neural network, decomposed into a set of parametric layers. Formally, we write $h_W := e_{W_e} \circ f_{W_f}^L := e_{W_e} \circ g_{W_L}^L \circ \cdots \circ g_{W_1}^1$, where $g_{W_i}^i \in \{g' : \mathbb{R}^{p_i} \to \mathbb{R}^{p_{i+1}}\}$ are parametric functions and $e_{W_e} \in \{e' : \mathbb{R}^{p_{L+1}} \to \mathbb{R}^C\}$ is a linear function. For example, $g_{W_i}^i$ could be a standard linear or convolutional layer, a residual block or a pooling layer. Here, σ is an element-wise ReLU activation function. With a slight abuse of notation, we omit specifying the sub-scripted weights, $f_i := g^i \circ \cdots \circ g^1$ and $h := h_W$.

In this work, we give special attention to the following architectures. The first architecture is a convolutional network, denoted by CONV-L-H. The network starts with a stack of a 2×2 convolutional layer with stride 2, batch normalization, a convolution of the same structure, batch normalization, and ReLU. Following that we have a set of L stacks of 3×3 convolutional layers with H channels, stride 1 and padding 1, batch normalization, and ReLU. The last layer is linear. The output tensors of these layers share the same shape as their input's shape. The second architecture is an MLP, denoted by MLP-L-H consisting of L hidden layers, where each layer contains a linear layer of width H, followed by batch normalization and ReLU. The last layer is linear.

Optimization. We optimize our models to minimize the regularized empirical risk $L_S^\lambda(h)$ by applying SGD for a certain number of iterations T with coefficient $\lambda > 0$. Specifically, we initialize the weights $W_0 = \gamma$ of h using a standard initialization procedure and at each iteration, we update $W_{t+1} \leftarrow W_t - \mu_t \nabla_W L_{\tilde{S}}(h_t)$, where $\mu_t > 0$ is the learning rate at the t'th iteration and the subset $\tilde{S} \subset S$ of size B is selected uniformly at random. Throughout the paper, we denote by h_S^γ the output of the learning algorithm starting from the initialization $W_0 = \gamma$. When γ is irrelevant or obvious from context, we will simply write $h_S^\gamma = h_S = e_S \circ f_S$.

3 NEURAL COLLAPSE AND GENERALIZATION

In this section we theoretically explore the relationship between neural collapse and generalization. We start by introducing neural collapse, NCC separability, and effective depth of neural networks. Then, we connect these notions with the test-time performance of neural networks.

3.1 NEAREST CLASS-CENTER SEPARABILITY

Neural collapse identifies training dynamics of deep networks for standard classification tasks, in which the features of the penultimate layer associated with training samples belonging to the same class tend to concentrate around their class-means. This includes (NC1) class-features variability collapse, (NC2) the class means of the embeddings collapse to the vertices of a simplex equiangular tight frame, (NC3) the last-layer classifiers collapse to the class means up to scaling and (NC4) the classifier's decision collapses to simply choosing whichever class has the closest train class mean, while maintaining a zero classification error.

In this paper we focus on a weak form of NC4 we call *"nearest class-center separability"* (NCC separability). Formally, suppose we have a dataset $S = \cup_{c=1}^{C} S_c$ of samples and a mapping $f : \mathbb{R}^d \to \mathbb{R}^p$, we say that the features of f are NCC separable (w.r.t. S) if for all $i \in [m]$, we have $\hat{h}(x_i) = y_i$, where

$$\hat{h}(x) \ := \ \arg\min_{c \in [C]} \|f(x) - \mu_f(S_c)\|. \tag{2}$$

To measure the degree of NCC separability of a feature map f, we use the train and test classification error rates of the NCC classifier on top of the given layer, $\mathrm{err}_S(\hat{h})$ and $\mathrm{err}_P(\hat{h})$.

Essentially, NC4 asserts that during training, the feature embeddings in the penultimate layer become separable and the classifier h itself converges to the 'nearest class-center classifier' $\hat{h}$.

3.2 EFFECTIVE DEPTHS AND GENERALIZATION

In this section we study the effective depths of neural networks and their connection with generalization. To formally define this notion, we focus on neural networks whose L top-most layers are of the same size (e.g., CONV-L-H or MLP-L-H). We observe that neural networks trained for standard classification exhibit an implicit bias towards depth minimization.

Observation 1 (Minimal depth hypothesis)**.** *Suppose we have a dataset S. There exists an integer $L_0 \geq 1$, such that, if we train a CONV-L-H or MLP-L-H of any depth $L \geq L_0$ for cross-entropy minimization on S using SGD with weight decay, the learned features f^l become (approximately) NCC separable for all $l \in \{L_0, \ldots, L\}$.*

We note that if the L_0'th layer of f_L exhibits NCC separability, we could correctly classify the samples already in the L_0'th layer of f_L using a linear classifier (i.e., the nearest class-center classifier). Therefore, intuitively its depth is effectively upper bounded by L_0. The notion of effective depth of a neural network is formally defined as follows.

Definition 1 (ϵ-effective depth)**.** *Suppose we have a dataset S and a neural network $h = e \circ g^L \circ \cdots \circ g^1$ with $g^1 : \mathbb{R}^n \to \mathbb{R}^{p_2}$, $g^i : \mathbb{R}^{p_i} \to \mathbb{R}^{p_{i+1}}$ and linear classifier $e : \mathbb{R}^{p_{L+1}} \to \mathbb{R}^C$. Let $\hat{h}_i(x) := \arg\min_{c \in [C]} \|f_i(x) - \mu_{f_i}(S_c)\|$. The ϵ-effective depth $\mathscr{d}_S^{\epsilon}(h)$ of the network h is the minimal value $i \in [L]$, such that, $\mathrm{err}_S(\hat{h}_i) \leq \epsilon$ (and $\mathscr{d}_S^{\epsilon}(h) = L$ if such $i \in [L]$ is non-existent).*

To avoid confusion, we note that the ϵ-effective depth is a property of a neural network and not of the function it implements. That is, a function can be implemented by two different architectures of different effective depths. While our empirical observations in Sec. 4 suggest that the optimizer learns neural networks of low-depths, it is not necessarily the lowest depth that allows NCC separability. As a next step, we define the *ϵ-minimal NCC depth*. Intuitively, the NCC depth of a given architecture is the minimal value $L \in \mathbb{N}$, for which there exists a neural network of depth L whose features are NCC separable. As we will show, the relationship between the ϵ-effective depth of a neural network and the ϵ-minimal NCC depth is connected with generalization.

Definition 2 (ϵ-Minimal NCC depth)**.** *Suppose we have a dataset $S = \cup_{c=1}^{C} S_c$ and a neural network architecture $f^L = g^L \circ \cdots \circ g^1$ with $g^1 : \mathbb{R}^n \to \mathbb{R}^{n_0}$ and $g^i \in \mathcal{G} \subset \{g' \mid g' : \mathbb{R}^{n_0} \to \mathbb{R}^{n_0}\}$ for all $i = 2, \ldots, L$. The ϵ-minimal NCC depth of $\mathcal{G}$ is the minimal depth L for which there exist parameters $W = \{W_i\}_{i=1}^{L}$, such that, $f' := f_W^L = g_{W_L}^L \circ \cdots \circ g_{W_1}^1$ satisfies $\mathrm{err}_S(\hat{h}) \leq \epsilon$, where $\hat{h}(x) := \arg\min_{c \in [C]} \|f'(x) - \mu_{f'}(S_c)\|$. We denote the ϵ-minimal NCC depth by $\mathscr{d}_{\min}^{\epsilon}(\mathcal{G}, S)$.*

To study the performance of a given model, we consider the following setup. Let $S_1 = \{(x_i^1, y_i^1)\}_{i=1}^m$ and $S_2 = \{(x_i^2, y_i^2)\}_{i=1}^m$ be two balanced datasets. We think of them as two splits of the training

dataset S. We assume that the classifier $h^{\gamma}_{S_1}$ is trained on S_1 and we use S_2 to evaluate its performance. We denote by $X_j = \{x^j_i\}_{i=1}^m$ and $Y_j = \{y^j_i\}_{i=1}^m$ the instances and labels in S_j.

To formally state our bound, we make two technical assumptions. The first is that the misclassified labels that $h^{\gamma}_{S_1}$ produces over the samples $X_2 = \cup_{c=1}^{C}\{x^2_{ci}\}_{i=1}^{m_0}$ are distributed uniformly.

Definition 3 (δ_m-uniform mistakes)**.** *We say that the mistakes of a learning algorithm $A : (S_1, \gamma) \mapsto h^{\gamma}_{S_1}$ are δ_m-uniform, if with probability $\geq 1 - \delta_m$ over the selection of $S_1, S_2 \sim P_B(m)$, the values and indices of the mistaken labels of $h^{\gamma}_{S_1}$ over X_2 are uniformly distributed (as a function of γ).*

The above definition provides two conditions regarding the learning algorithm. It assumes that with a high probability (over the selection of S_1, S_2), $h^{\gamma}_{S_1}$ makes the same number of mistakes on S_2 across all initializations γ. In addition, it assumes that the mistakes are distributed uniformly across the samples in S_2 and their (incorrect) values are also distributed uniformly. While these assumptions may be violated in practice, the train error typically has a small variance and the mistakes are almost distributed uniformly when the classes are non-hierarchical (e.g., CIFAR10, MNIST).

For the second assumption, we consider the following term. Let $p \in (0, 1/2), \alpha \in (0, 1)$, we denote

$$\delta^2_{m,p,\alpha} \;:=\; \mathbb{P}_{S_1,S_2,\tilde{Y}_2,\hat{Y}_2}\left[\exists\, q \geq (1+\alpha)\, p :\; \mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2) > \mathbb{E}_{\hat{Y}_2}[\mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \hat{S}_2)]\right], \quad (3)$$

where $\tilde{Y}_2 = \{\tilde{y}_i\}_{i=1}^m$ and $\hat{Y}_2 = \{\hat{y}_i\}_{i=1}^m$ are uniformly selected to be sets of labels that disagree with Y_2 on pm and qm values (resp.) and $\tilde{S}_2$ and $\hat{S}_2$ are datasets obtained by replacing the labels of S_2 with $\tilde{Y}_2$ and $\hat{Y}_2$ (resp.). We assume that $\delta^2_{m,p,\alpha}$ is small. Meaning, with a high probability, the minimal depth to fit $(2-p)m$ correct labels and pm random labels is upper bounded by the expected minimal depth to fit $(2-q)m$ correct labels and qm random labels for any $q \geq (1+\alpha)p$. To understand this assumption, we note that in both cases, the model has to fit at least m correct labels and pm (or qm) random labels. However, we typically need to increase the capacity of the model in order to fit extended amounts of random labels (see Figs. 3 and 4).

Following the setting above, we are prepared to formulate our generalization bound.

Proposition 1. *Let $m \in \mathbb{N}$, $p \in (0, 1/2)$, $\alpha \in (0, 1)$ and $\epsilon \in (0, 1)$. Assume that the error of the learning algorithm is δ^1_m-uniform. Assume that $S_1, S_2 \sim P_B(m)$. Let $h^{\gamma}_{S_1}$ be the output of the learning algorithm given access to a dataset S_1 and initialization γ. Then,*

$$\begin{aligned}\mathbb{E}_{S_1}\mathbb{E}_{\gamma}[\mathrm{err}_P(h^{\gamma}_{S_1})] \;\leq\; \mathbb{P}_{S_1,S_2,\tilde{Y}_2}&\left[\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})] \;\geq\; \mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right] \\ &+ (1+\alpha)p + \delta^1_m + \delta^2_{m,p,\alpha},\end{aligned} \quad (4)$$

where $\tilde{Y}_2 = \{\tilde{Y}_i\}_{i=1}^m$ is uniformly selected to be a set of labels that disagrees with Y_2 on pm values.

The above proposition provides an upper bound on the expected test error of the classifier $h^{\gamma}_{S_1}$ which is the term that we would like to bound. The proposition assumes that the mistakes $h^{\gamma}_{S_1}$ generates on X_2 are distributed uniformly (with probability $\geq 1 - \delta^1_m$). To account the likelihood that this assumption fails, our bound includes the term δ^1_m, which is assumed to be small.

Informally, the bound suggests the following idea to evaluate the performance of $h^{\gamma}_{S_1}$. We start with an initial guess $p_m = p \in (0, 1/2)$ of the test error of $h^{\gamma}_{S_1}$. Using this guess, we compare its ϵ-effective depth with the ϵ-minimal NCC depth $\mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)$ required to NCC separate the samples in $S_1 \cup \tilde{S}_2$, where $\tilde{S}_2$ is the result of randomly relabeling $p_m m$ of S_2's labels. Intuitively, if the mistakes of $h^{\gamma}_{S_1}$ are uniformly distributed and its ϵ-effective depth is smaller than $\mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)$, then, we expect $h^{\gamma}_{S_1}$ to make at most p_m mistakes on S_2. Therefore, in a sense, the choice of p_m serves as a 'guess' whether the effective depth of a model trained with S_1 is likely to be smaller than the ϵ-minimal NCC depth required to NCC separate the samples in $S_1 \cup \tilde{S}_2$.

Next, we interpret each term separately. The term $\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})]$ depends on both the complexity of the classification problem and the implicit bias of SGD to favor networks of small ϵ-effective depths. For example, if SGD does not minimize the ϵ-effective depth or the labels in S_1 are statistically independent of the inputs for sufficiently large m, we expect $\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})] = L$. Simply put, $\mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)$ measures the complexity of a task that involves fitting a dataset of size $2m$ samples,

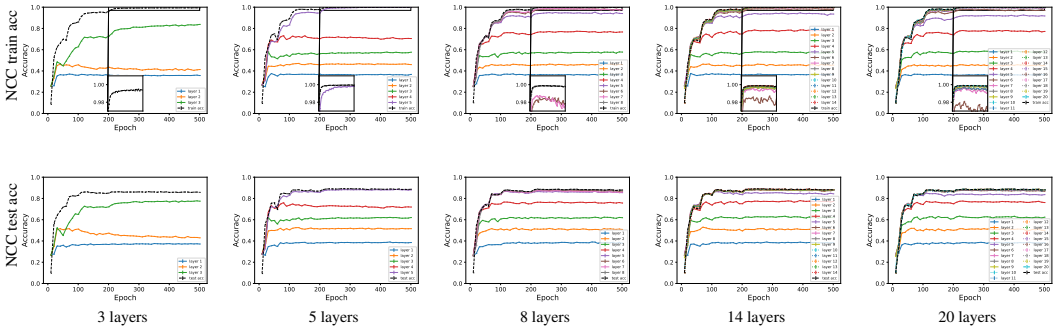

Figure 1: **Intermediate neural collapse of CONV-L-400 trained on CIFAR10.** We plot the NCC train/test accuracy rates of neural networks with varying numbers of hidden layers evaluated on the train data (plotted in lin-log scale). Each curve stands for a different layer within the network.

where $(2-p_m)m$ of the labels are correct and $p_m m$ are random labels. By decreasing p_m, we expect $\mathcal{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)$ to decrease, making the first term in the bound larger. In addition, if $h = e \circ f^L$ is a neural network of a fixed width, it is impossible to fit an increasing amount of random labels without increasing L. Therefore, when $p_m m \to \infty$, the dataset $S_1 \cup \tilde{S}_2$ becomes increasingly harder to fit, and we expect $\mathcal{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)$ to tend to infinity. On the other hand, if $\mathbb{E}_{\gamma}[\mathcal{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})]$ is bounded as a function of L and m and if $p_m = \frac{1}{\sqrt{m}}$, we obtain that $\mathbb{P}\left[\mathbb{E}_{\gamma}[\mathcal{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})] \geq \mathcal{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right] \underset{m\to\infty}{\longrightarrow} 0$ and $p_m \underset{m\to\infty}{\longrightarrow} 0$, giving us $\mathbb{E}_{S_1}[\text{err}_P(h_{S_1})] \leq \delta^1_m + \delta^2_{m,p,\alpha} + o_m(1)$.

Interestingly, whenever our minimal depth hypothesis (Obs. 1) holds, then $\mathbb{E}_{\gamma}[\mathcal{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})]$ should be (relatively) unaffected by the depth L of $h^{\gamma}_{S_1}$ as long as $L \geq L_0$. Therefore, in this regime, according to Prop. 1, the test performance of $h^{\gamma}_{S_1}$ should not decrease when increasing L beyond L_0.

We note that the proposed generalization bound is fairly different from traditional generalization bounds (Vapnik, 1998; Shalev-Shwartz & Ben-David, 2014; Mohri et al., 2012). Typically the expected test error is bounded by the sum between the train error and the ratio between the complexity of the learned hypothesis (e.g., number of trainable parameters) and $\sqrt{m}$. However, in many practical scenarios, the complexity of the learned hypothesis exceeds $\sqrt{m}$. We note that even in the presence of an implicit depth minimization, a standard parameter counting generalization bound would be vacuous. That is because, the overall number of parameters of the network after replacing the top, redundant, layers with a nearest class-center classifier would typically still exceed $\sqrt{m}$. On the other hand, Prop. 1 offers a different way to measure generalization. We do not require that the network's size be small in comparison to $\sqrt{m}$; rather, the bound guarantees generalization if the network's effective size is smaller than that of a network that fits partially random labels.

In general, computing the expectation over S_1, S_2 in the bound is impossible, due to the limited access of the training data. However, instead, we empirically estimate this term using a set of k pairs (S^i_1, S^i_2) of m samples, yielding an additional term that scales as $\mathcal{O}(1/\sqrt{k})$ to the bound (see Prop. 2 in the appendix).

4 Experiments

In this section, we experimentally analyze the emergence of neural collapse in the intermediate layers of neural networks. First, we validate the "Minimal Depth Hypothesis" (Obs. 1). Following that, we look at how corrupted labels affect the extent of intermediate layer NCC separability and the $epsilon$-effective depth. We show that as the number of corrupted labels in the data increases, so does the $epsilon$-effective depth. Finally, using the bound in Prop. 1, we provide non-trivial estimates of the test error. In Tab. 2, we empirically compare our bound with relevant baselines and show that, unlike other bounds, it achieves non-vacuous estimations of the test error. Throughout the experiments, we used Tesla-k80 GPUs for several hundred runs. Each run took between 5-20 hours. For additional experiments, see the appendix.

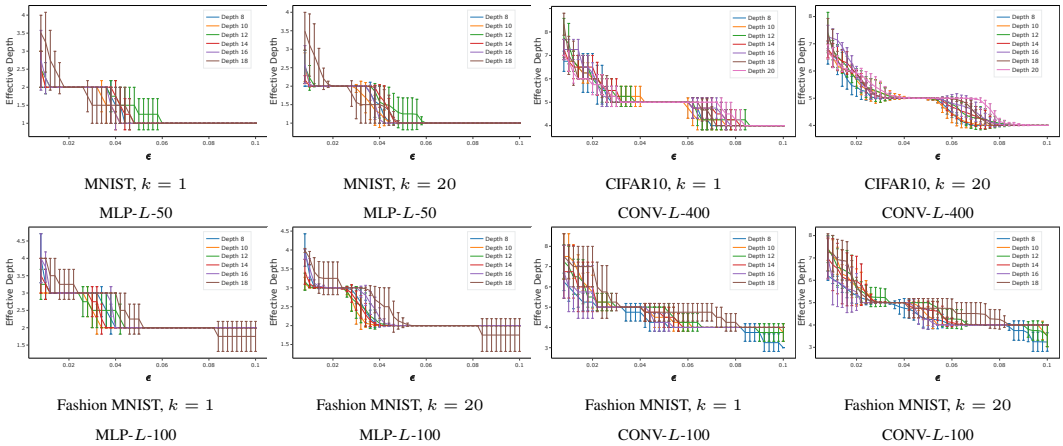

Figure 2: **Averaged ϵ-effective depths over the last few epochs.** We plot the ϵ-effective depth (y-axis) as a function of ϵ (x-axis). Each line specifies the ϵ-effective depth of a neural network of a certain depth L. We show the averaged ϵ-effective depth over the last $k = 1, 20$ epochs across 5 initializations. The network's architecture, dataset and k are specified below each plot.

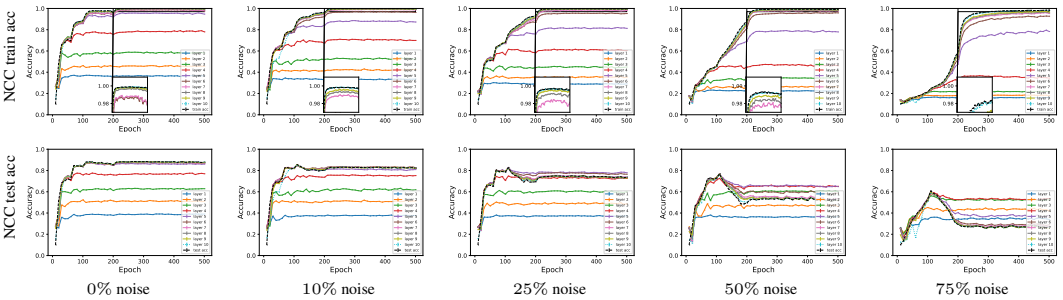

Figure 3: **Intermediate neural collapse of CONV-10-400 trained on CIFAR10 with partially corrupted labels.** We plot the NCC train/test accuracy rates of the various layers of a network trained with a certain amount of corrupted labels (see titles).

4.1 Setup

Training process. We consider k-class classification problems (e.g., CIFAR10) and train multi-layered neural networks $h = e \circ f^L = e \circ g^L \circ \cdots \circ g^1 : \mathbb{R}^n \to \mathbb{R}^C$ on the corresponding training dataset S. The models are trained with SGD for cross-entropy loss minimization between its logits and the one-hot encodings of the labels. We consistently use batch size 128, learning rate schedule with an initial learning rate 0.1, decayed three times by a factor of 0.1 at epochs 60, 120, and 160, momentum 0.9 and weight decay $5\mathrm{e}{-4}$. Each model is trained for 500 epochs.

Datasets. We consider various datasets: MNIST, Fashion MNIST, and CIFAR10. For CIFAR10 we used random cropping, random horizontal flips, and random rotations (by $15k$ degrees for k uniformly sampled from [24]). All datasets were standardized.

4.2 Results

Intermediate neural collapse. To study the bias towards minimal depth, we trained a set of CONV-L-400 networks with varying depths. In Fig. 1 we report the train and test NCC classification accuracy rates for networks of varying depths L on the CIFAR10 dataset. We make multiple interesting observations; **(i)** For networks with 8 or higher hidden layers, the eighth and higher layers exhibit NCC train accuracy of approximately 100%, and therefore, are effectively of depth 7. **(ii)** We observe that neural collapse strengthens when increasing the network's depth, on both train and

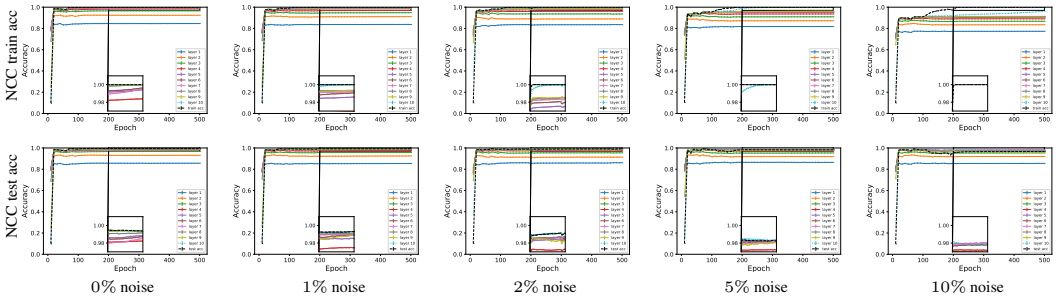

Figure 4: **Intermediate neural collapse of CONV-10-50 trained on MNIST with partially corrupted labels.** See Fig. 3 for details.

Dataset	MNIST			Fashion MNIST			CIFAR10		
Architecture	CONV-10-50			CONV-10-100			CONV-16-100		
$\mathbb{E}_{S_1,\gamma}[\text{err}_P(h^{\gamma}_{S_1})]$	0.0075			0.0996			0.2676		
p	0.05	0.075	0.1	0.05	0.15	0.2	0.4	0.45	0.5
Bound	1.05	0.475	**0.1**	1.05	0.75	**0.2**	**0.66**	0.72	0.7

Table 1: Estimating the bound in Prop. 1. We used $\epsilon = 0.005$ to measure the effective depths.

test data. **(iii)** The embeddings of the top layers become NCC separable approximately at the same epoch. **(iv)** We also observe that the final epoch's NCC train/test accuracy rates of any intermediate layer converges when increasing L. The results of this experiment are substantially extended and repeated with different architectures and datasets in the appendix.

The effect of the depth on the ϵ-effective depth. In Obs. 1 we claimed that the ϵ-effective depth is insensitive to the actual depth of the network (once it exceeds a certain threshold). To validate this hypothesis we conducted the following experiments. We trained models on MNIST, Fashion MNIST and CIFAR10 with varying depth L. In Fig. 2 we plotted the averaged ϵ-effective depths of each network's last $k = 1, 20$ epochs as a function of ϵ. We also average the results across 5 different weight initializations and plot them along with error bar standard deviations. As can be seen, the ϵ-effective depth is almost unaffected by the choice of L for a given ϵ. Remarkably, for each ϵ, the averaged effective depth varies very little across the various networks. This means that the ϵ-effective depths of two trained deep networks of different depths are more or less the same.

NCC separability with partially corrupted labels. Simply put, Prop. 1 compares the depths required to fit correct labels and partially corrupt labels. To better understand the effect of corrupted labels on the complexity of the task, we compare the ϵ-effective depths of models trained with varying amounts of corrupted labels. Namely, we study the *degree* of NCC separability in the intermediate layers of neural networks that are trained with varying amounts of corrupted labels.

For this experiment we trained instances of CONV-10-400 for CIFAR10 classification with 0%, 10%, 25%, 50% and 75% of the labels corrupted (e.g., uniformly distributed random labels). We plot the degrees of NCC separation on the train and test sets, $1 - \text{err}_S(\hat{h}_i)$ and $1 - \text{err}_P(\hat{h}_i)$, across the intermediate layers of the neural networks during the optimization procedure.

As can be seen in Fig. 3, we achieve NCC separability in the penultimate layer when training with or without corrupted labels, which is consistent with the experiments in (Mixon et al., 2020). However, we notice several differences between the two cases. For starters, a higher degree of NCC separability is achieved when training without corrupted labels across all layers. Furthermore, when training with 10% or 25% corrupted labels, the sixth layer's NCC accuracy rate drops lower than 98%, in comparison with training without corrupted labels that gives us $> 98\%$ accuracy. Therefore, the ϵ-effective depth of the former network is 6 while the latter's is 5, when $\epsilon = 0.02$ (see Def. 1).

In Fig. 4 we repeat the experiment with CONV-10-50 trained on MNIST with 0%, 1%, 2%, 5% and 10% corrupted labels. We note that, as long as there at most 10% corrupted labels, the models perfectly fit the training labels and achieve perfect NCC separability in their corresponding penultimate layers. On the other hand, by looking at the degree of intermediate NCC separability we can distin-

Dataset	MNIST			Fashion MNIST			CIFAR10		
Architecture	CONV-L-50			CONV-L-100			CONV-L-100		
Depth (L)	10	12	15	10	12	15	16	18	20
$\mathbb{E}_{S_1,\gamma}[\text{err}_P(h^{\gamma}_{S_1})]$	0.0075	0.0074	0.0074	0.0996	0.0996	0.0996	0.2659	0.2653	0.2648
p	0.1	0.1	0.1	0.2	0.2	0.2	0.4	0.4	0.4
Proposed Bound	**0.1**	**0.1**	**0.1**	**0.2**	**0.2**	**0.2**	0.66	0.66	**0.53**
$L_{1,\infty}$ (Bartlett & Mendelson, 2002)	8.911+14	1.74e+17	2.13e+22	3.613e+17	9.145e+18	4.088e+22	1.076e+23	6.682e+28	2.758e+35
$L_{3,1.5}$ (Neyshabur et al., 2015)	5.462e+05	1.6e+06	1.308e+06	7.523e+07	6.997e+07	2.636e+08	4.633e+08	2.275e+09	5.061e+09
Frobenius (Neyshabur et al., 2015)	1.848e+06	8.194e+06	2.216e+07	2.486e+08	2.335e+08	1.585e+09	1.967e+09	1.442e+10	3.038e+11
Spec L_1 (Bartlett et al., 2017)	2.861e+05	6.412e+05	9.566e+05	4.706e+06	3.516e+06	3.176e+06	1.19e+07	1.449e+08	1.272e+10
Spec Frob (Neyshabur et al., 2019)	**3948.414**	**11199.209**	**1.538e+04**	**40229.583**	**2.884e+04**	**2.543e+04**	**94833.424**	**1.011e+06**	**1.033e+08**

Table 2: **Comparing our bound with baseline bounds in the literature for networks of varying depths.** While traditional bounds are extremely vacuous, our bound provides non-trivial estimations of the test error. Furthermore, unlike traditional bounds, our bound is generally independent of the depth of the network. This is consistent with the minimal depth hypothesis described in Obs. 1.

guish between the two training regimes (with/without corrupted labels). For example, the ϵ-effective depth of the network trained with 0% corrupted labels is 5 and for 2% it is 10 (for $\epsilon = 0.01$).

As a side note, we also notice (Figs. 3 and 4, second row) that the NCC classifiers corresponding to intermediate layers tend to be more resilient to corrupted labels than the model itself.

Estimating the bound in equation 4. We estimate the bound in equation 4 for multiple architectures and datasets. In each case we used $\epsilon = 0.005$ by default and employed different 'guesses' p (see Tab. 1) depending on the complexity of the learning task. We report an estimation of the expected test error of the models, $\mathbb{E}_{S_1,\gamma}[\text{err}_P(h^{\gamma}_{S_1})]$ and an estimation of the bound for each selection of p. For concrete technical details, see Appendix A.

As can be seen, for appropriate selections of p, we obtained non-trivial estimates to the test performance of the models, which are almost unheard of when it comes to standard bounds for deep neural networks. As expected, if the guess p is overoptimistic (e.g., close to $\mathbb{E}_{S_1,\gamma}[\text{err}_P(h^{\gamma}_{S_1})]$), then, the first term in the bound tends to be large compared to $\mathbb{E}_{S_1,\gamma}[\text{err}_P(h^{\gamma}_{S_1})]$.

Following that, given that the ϵ-effective depth of sufficiently deep neural networks is generally insensitive to depth (see Fig. 2), we expect the bound to be insensitive to depth as well. We estimate the bound in equation 4 for CONV-L-50 trained on MNIST and CONV-L-100 trained on Fashion MNIST and CIFAR10 with $L = 10, 12, 15$ for the first two and with $L = 15, 18, 20$ for CIFAR10. As shown in Tab. 2, we obtain similar bounds for each depth. Finally, we compare our bound to several baseline generalization bounds for deep networks to show that it outperforms traditional generalization bounds. We used the implementation of Neyshabur et al. (2019) to compute the bounds. While our bound is non-vacuous and generally independent of depth, the traditional bounds are extremely vacuous and rapidly grow when increasing the depth.

5 CONCLUSIONS

Understanding the ability of SGD to generalize well when training overparameterized neural network is attributed as one of the major open problems in deep learning theory (Zhang et al., 2017). In this paper we offer a new angle to study the role of depth in deep learning and the connection between neural collapse and generalization.

We characterize a notion of effective depth that measures the lowest layer that enjoys NCC separability. We introduce a novel generalization bound that measures the likelihood in which the effective depth of a trained neural network is (strictly) smaller than the minimal depth required to achieve NCC separability with partially corrupted labels. This criterion, as demonstrated empirically, is a good predictor of generalization. Furthermore, we characterize and empirically demonstrate that when sufficiently deep networks are trained, they converge to the same effective depth, implying that our bound does not worsen as the depth increases.

We hope that this work will spark further research into the generalization bounds discussed in this paper. It would be interesting to see if these bounds could be improved by replacing the effective depth as defined in this paper with a different notion of complexity. It would also be interesting to investigate the mathematical conditions under which Obs. 1 holds.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *ArXiv*, abs/1610.01644, 2017.

Alnur Ali, J. Zico Kolter, and Ryan J. Tibshirani. A continuous-time view of early stopping for least squares regression. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pp. 1370–1378. PMLR, 16–18 Apr 2019. URL https://proceedings.mlr.press/v89/ali19a.html.

Kai Arulkumaran, Antoine Cully, and Julian Togelius. Alphastar: An evolutionary computation perspective, 2019. URL http://arxiv.org/abs/1902.01724. cite arxiv:1902.01724.

Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. In *J. Mach. Learn. Res.*, 2001.

Peter L. Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *J. Mach. Learn. Res.*, 3:463–482, 2002.

Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 6241–6250, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Mikhail Belkin. Fit without fear: remarkable mathematical phenomena of deep learning through the prism of interpolation. *Acta Numerica*, 30:203 – 248, 2021.

Ido Ben-Shaul and Shai Dekel. Sparsity-probe: Analysis tool for deep learning models. *ArXiv*, abs/2105.06849, 2021.

Ido Ben-Shaul and Shai Dekel. Nearest class-center simplification through intermediate layers. *CoRR*, abs/2201.08924, 2022. URL https://arxiv.org/abs/2201.08924.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Dnn or k-nn: That is the generalize vs. memorize question, 2018. URL https://arxiv.org/abs/1805.06822.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, jun 2019.

Tolga Ergen and Mert Pilanci. Revealing the structure of deep neural networks via convex duality. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3004–3014. PMLR, 18–24 Jul 2021.

Cong Fang, Hangfeng He, Qi Long, and Weijie J. Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences*, 118(43), 2021.

Tomer Galanti and Tomaso Poggio. Sgd noise and implicit low-rank bias in deep neural networks, 2022. URL https://arxiv.org/abs/2206.05794.

Tomer Galanti, András György, and Marcus Hutter. On the role of neural collapse in transfer learning. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=SwIp410B6aQ.

Tomer Galanti, András György, and Marcus Hutter. Improved generalization bounds for transfer learning via neural collapse. *ICML Workshop on Pre-training: Perspectives, Pitfalls, and Paths Forward*, 2022b.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. *Information and Inference: A Journal of the IMA*, 9, 12 2017. doi: 10.1093/imaiai/iaz007.

Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1832–1841. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/gunasekar18a.html.

X.Y. Han, Vardan Papyan, and David L. Donoho. Neural collapse under MSE loss: Proximity to and dynamics on the central path. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=w1UbdvWH_R3.

Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension bounds for piecewise linear neural networks. *ArXiv*, abs/1703.02930, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Kaiming He et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet. In *ICCV*, 2015.

Like Hui, Mikhail Belkin, and Preetum Nakkiran. Limitations of neural collapse for understanding generalization in deep learning. *arXiv preprint arXiv:2202.08384*, 2022.

Thien Le and Stefanie Jegelka. Training invariances and the low-rank phenomenon: beyond linear networks. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=XEW8CQgArno.

Jianfeng Lu and Stefan Steinerberger. Neural collapse with cross-entropy loss. *CoRR*, abs/2012.08465, 2020.

Dustin G. Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features, 2020.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012. ISBN 026201825X.

Grégoire Montavon, Mikio L. Braun, and Klaus-Robert Müller. Kernel analysis of deep networks. *J. Mach. Learn. Res.*, 12:2563–2581, 2011.

Behnam Neyshabur. Implicit regularization in deep learning. *ArXiv*, abs/1709.01953, 2017.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In Peter Grünwald, Elad Hazan, and Satyen Kale (eds.), *Proceedings of The 28th Conference on Learning Theory*, volume 40 of *Proceedings of Machine Learning Research*, pp. 1376–1401, Paris, France, 03–06 Jul 2015. PMLR. URL https://proceedings.mlr.press/v40/Neyshabur15.html.

Behnam Neyshabur, Srinadh Bhojanapalli, David A. McAllester, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *ArXiv*, abs/1707.09564, 2018.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *ArXiv*, abs/1805.12076, 2019.

Vardan Papyan. Traces of class/cross-class structure pervade deep learning spectra. *Journal of Machine Learning Research*, 21(252):1–64, 2020. URL http://jmlr.org/papers/v21/20-933.html.

Vardan Papyan, Yaniv Romano, and Michael Elad. Convolutional neural networks analyzed via convolutional sparse coding. *J. Mach. Learn. Res.*, 18:83:1–83:52, 2017.

Vardan Papyan, X. Y. Han, and David L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40): 24652–24663, 2020.

Akshay Rangamani, Mengjia Xu, Andrzej Banburski, Qianli Liao, Tomer Galanti, and Tomaso Poggio. Dynamics and neural collapse in deep classifiers trained with the square loss. Technical report, Center for Brains, Minds and Machines (CBMM), 2022.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms.* Cambridge University Press, 2014. ISBN 978-1-10-705713-5.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *ArXiv*, abs/1703.00810, 2017.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016. ISSN 0028-0836. doi: 10.1038/nature16961.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Nadav Timor, Gal Vardi, and Ohad Shamir. Implicit regularization towards rank minimization in relu networks. *CoRR*, abs/2201.12760, 2022. URL https://arxiv.org/abs/2201.12760.

Tom Tirer and Joan Bruna. Extended unconstrained features model for exploring deep neural collapse, 2022. URL https://arxiv.org/abs/2202.08087.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *arXiv preprint arXiv:2203.00555*, 2022.

Chenfeng Xu, Shijia Yang, Tomer Galanti, Bichen Wu, Xiangyu Yue, Bohan Zhai, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Image2point: 3d point-cloud understanding with 2d image pretrained models. In *IEEE European Conference on Computer Vision (ECCV)*, 2022.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers, 2021.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy8gdB9xx.

Zhihui Zhu, Tianyu DING, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=KRODJAa6pzE.

A Additional Experiments

A.1 Estimating the Generalization Bound

In this section we describe how we empirically estimate the bound in Prop. 1.

Estimating the bound. We would like to estimate the first term in the bound,

$$\mathbb{P}_{S_1,S_2,\tilde{Y}_2}\left[\mathbb{E}_\gamma[\mathscr{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \geq \mathscr{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right]. \tag{5}$$

According to Prop. 2 in order to estimate this term we need to generate i.i.d. triplets $(S^i_1, S^i_2, \tilde{Y}^i_2)$. Since we have a limixted access to training data, we use a variation of cross-validation and generate $k_1 = 5$ i.i.d. disjoint splits (S^i_1, S^i_2) of the training data S. For each one of these pairs, we generate $k_2 = 3$ corrupted labelings $\tilde{Y}^{ij}_2$. We denote by $\tilde{S}^{ij}_2$ the set obtained by replacing the labels of S^i_2 with $\tilde{Y}^{ij}_2$ and $\tilde{S}^{ij}_3 := S^i_1 \cup \tilde{S}^{ij}_2$.

As a first step, we would like to estimate $\mathbb{E}_\gamma[\mathscr{d}^\epsilon_{S^i_1}(h^\gamma_{S^i_1})]$ for each $i \in [k_1]$. For this purpose, we randomly select $T_1 = 5$ different initializations $\gamma_1, \dots, \gamma_{T_1}$ and for each one, we train the model $h^{\gamma_t}_{S^i_1}$ using the training protocol described in Sec. 4.1. Once trained, we compute $\mathscr{d}^\epsilon_{S_1}(h^{\gamma_t}_{S^i_1})$ for each $t \in [T_1]$ (see Def. 1) and approximate $\mathbb{E}_\gamma[\mathscr{d}^\epsilon_{S^i_1}(h^\gamma_{S^i_1})]$ using $d_i := \frac{1}{T_1}\sum_{t=1}^{T_1} \mathscr{d}^\epsilon_{S^i_1}(h^{\gamma_t}_{S^i_1})$.

As a next step, we would like to evaluate $\mathbb{I}[d_i \geq \mathscr{d}^\epsilon_{\min}(\mathcal{G}, \tilde{S}^{ij}_3)]$. We notice that $d_i \geq \mathscr{d}^\epsilon_{\min}(\mathcal{G}, S^i_1 \cup \tilde{S}^i_2)$ if and only if there is a d_i-layered neural network $f = g^{d_i} \circ \dots \circ g^1$ for which $\text{err}_{\tilde{S}^{ij}_3}(\hat{h}) \leq \epsilon$, where $\hat{h}(x) := \arg\min_{c\in[C]} \|f(x) - \mu_f(S_c)\|$. In general, computing this boolean value is computationally hard. Therefore, to estimate this boolean value, we simply train a (d_i+1)-layered network $h = e \circ f$ and check whether its penultimate layer is ϵ-NCC separable, i.e., $\text{err}_{\tilde{S}^{ij}_3}(\hat{h}) \leq \epsilon$, where $\hat{h}(x) := \arg\min_{c\in[C]} \|f(x) - \mu_f(S_c)\|$. If SGD implicitly optimizes neural networks to maximize NCC separability as observed in (Papyan et al., 2020) (and also in this paper), we should expect to obtain ϵ-NCC separability in the penultimate layer if that is possible with a d_i-layered network. Since training might be non-optimal, to obtain a robust estimation, we train $T_2 = 5$ models $h_t = e_t \circ f_t$ of depth $d_i + 1$ and pick the one with the best NCC separability in its penultimate layer. Namely, we replace $\mathscr{d}^\epsilon_{\min}(\mathcal{G}, \tilde{S}^{ij}_3)$ with $\min_{t\in[T_2]} \mathscr{d}^\epsilon_{\tilde{S}^{ij}_3}(h_t)$ and estimate $\mathbb{I}[d_i \geq \mathscr{d}^\epsilon_{\min}(\mathcal{G}, \tilde{S}^{ij}_3)]$ using $\mathbb{I}[d_i \geq \min_{t\in[T_2]} \mathscr{d}^\epsilon_{\tilde{S}^{ij}_3}(h_t)]$.

Our final estimation is the following

$$\frac{1}{k_1}\sum_{i=1}^{k_1}\frac{1}{k_2}\sum_{j=1}^{k_2}\mathbb{I}\left[d_i \geq \min_{t\in[T_2]} \mathscr{d}^\epsilon_{\tilde{S}^{ij}_3}(h_t)\right] \approx \mathbb{P}_{S_1,S_2,\tilde{Y}_2}\left[\mathbb{E}_\gamma[\mathscr{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \geq \mathscr{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right]. \tag{6}$$

In order to estimate the bound we assume that δ^1_m and $\delta^2_{m,p,\alpha}$ are negligible constants and that $\alpha = 1$. The estimation of the bound is given by the sum of the left hand side in equation 6 and p.

Estimating the mean test error. To estimate the mean test error, $\mathbb{E}_{S_1,\gamma}[\text{err}_P(h^\gamma_{S_1})]$, as typically done in machine learning, we replace the population distribution P with the test set S_{test} and we replace the expectation over S_1 and γ with averages across the $k_1 = 5$ random selections of $\{S^i_1\}_{i=1}^{k_1}$ and $T_1 = 5$ random selections of $\{\gamma_t\}_{t=1}^{T_1}$. Namely, we compute the following $\frac{1}{k_1}\sum_{i=1}^{k_1}\frac{1}{T_1}\sum_{t=1}^{T_1}\text{err}_{S_{test}}(h^{\gamma_t}_{S^i_1}) \approx \mathbb{E}_{S_1,\gamma}[\text{err}_P(h^\gamma_{S_1})]$.

A.2 Neural Collapse

To obtain a comprehensive analysis of collapse across layers, we also estimate the degree of NC1.

To evaluate NC1, we follow the process suggested by Galanti et al. (2022a), which is a simplified version of the original approach of Papyan et al. (2020). For a feature map $f : \mathbb{R}^d \to \mathbb{R}^p$ and two (class-conditional) distributions[1] Q_1, Q_2 over $\mathcal{X} \subset \mathbb{R}^d$, we define their *class-distance normalized*

[1]The definition can be extended to finite sets $S_1, S_2 \subset \mathcal{X}$ by defining $V_f(S_1, S_2) = V_f(U[S_1], U[S_2])$.

variance (CDNV) to be

$$V_f(Q_1, Q_2) \;:=\; \frac{\mathrm{Var}_f(Q_1) + \mathrm{Var}_f(Q_2)}{2\|\mu_f(Q_1) - \mu_f(Q_2)\|^2},$$

where $\mu_u(Q) := \mathbb{E}_{x\sim Q}[u(x)]$ and by $\mathrm{Var}_u(Q) := \mathbb{E}_{x\sim Q}[\|u(x) - \mu_u(Q)\|^2]$ the mean and variance of $u(x)$ for $x \sim Q$. Essentially, this quantity measures to what extent the feature vectors of samples from Q_1 and Q_2 are separated and clustered in space.

To demonstrate the gradual evolution of collapse across the layers, for each sub-architecture $f^i = g^i \circ \cdots \circ g^1(x)$ we consider the train and test class features variations $\mathrm{Avg}_{c\neq c'}[V_{f^i}(S_c, S_{c'})]$ and $\mathrm{Avg}_{c\neq c'}[V_{f^i}(P_c, P_{c'})]$. The population distribution of each class, P_c, is replaced with the test samples of that class.

As shown by Galanti et al. (2022a), this definition is essentially the same as that of Papyan et al. (2020). Furthermore, they showed that the NCC classification error rate can be upper bounded in terms of the CDNV. However, the NCC error can be zero in cases where the CDNV is larger than zero. For example, if the two classes are uniformly distributed over the 1-radius circles around the points $(-1, 0)$ and $(1, 0)$ in $\mathbb{R}^2$, then they are perfectly NCC separable while the CDNV between the two distributions is 0.25.

Auxiliary experiments on the effective depth. In Figs. 5-8 we plot the CDNV and the NCC accuracy rates of neural networks with varying numbers of hidden layers evaluated on the train and test data. Each curve stands for a different layer within the network. As can be seen, in all cases, for networks deeper than a threshold we obtain (near perfect) NCC separability in all of the top layers. Furthermore, the degree of neural collapse seems to improve with the network's depth.

Auxiliary experiments with noisy labels. In Figs. 9-11 we repeat the experiment in Fig. 3 and plot the results of the same experiment, with different networks and datasets (see captions). As can be seen, the effective NCC depth of a neural network tends to increase as we train with increasing amounts of corrupted labels.

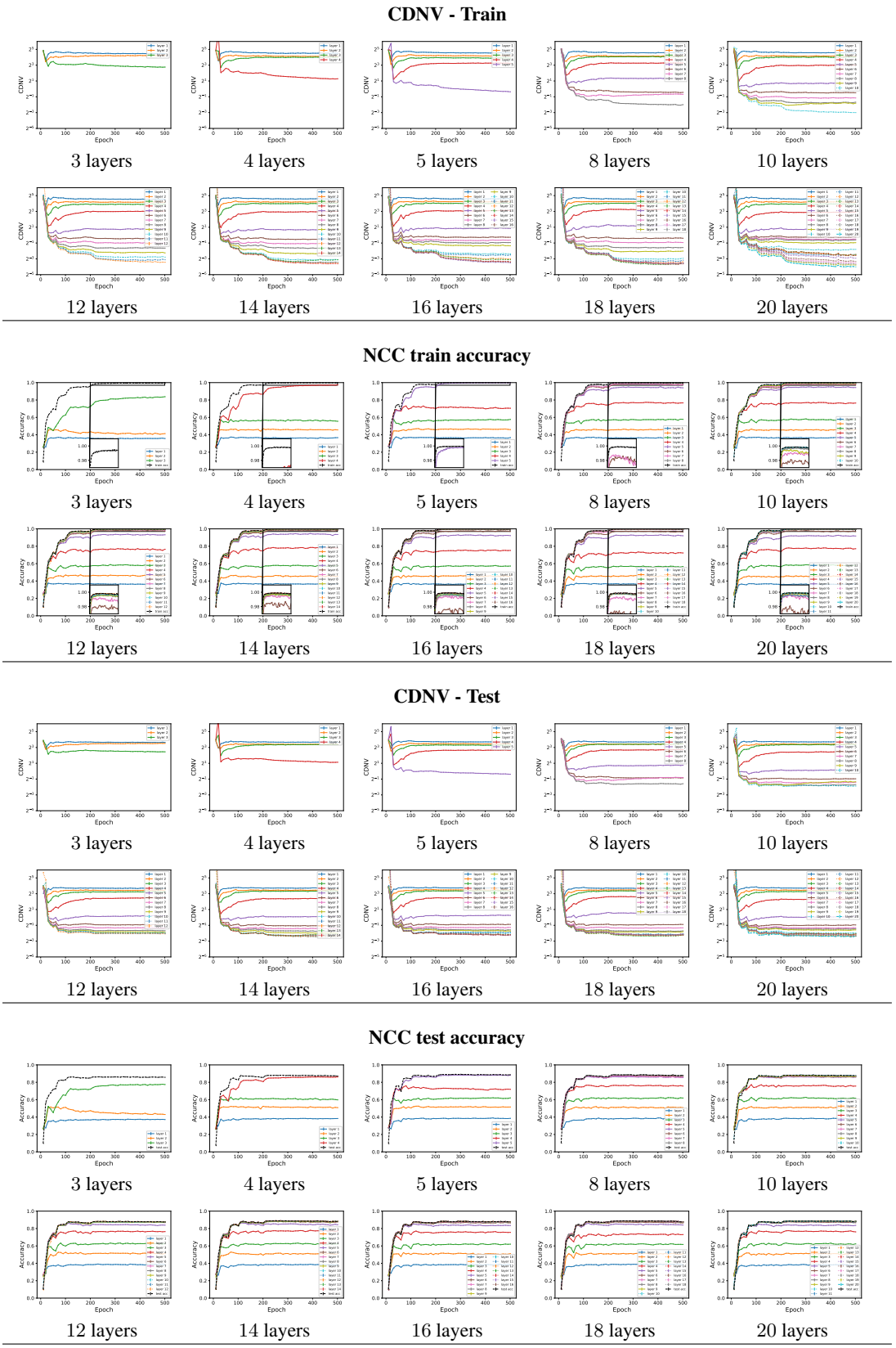

Figure 5: **Intermediate neural collapse of CONV-L-400 trained on CIFAR10.** See Fig. 1 in the main text for details.

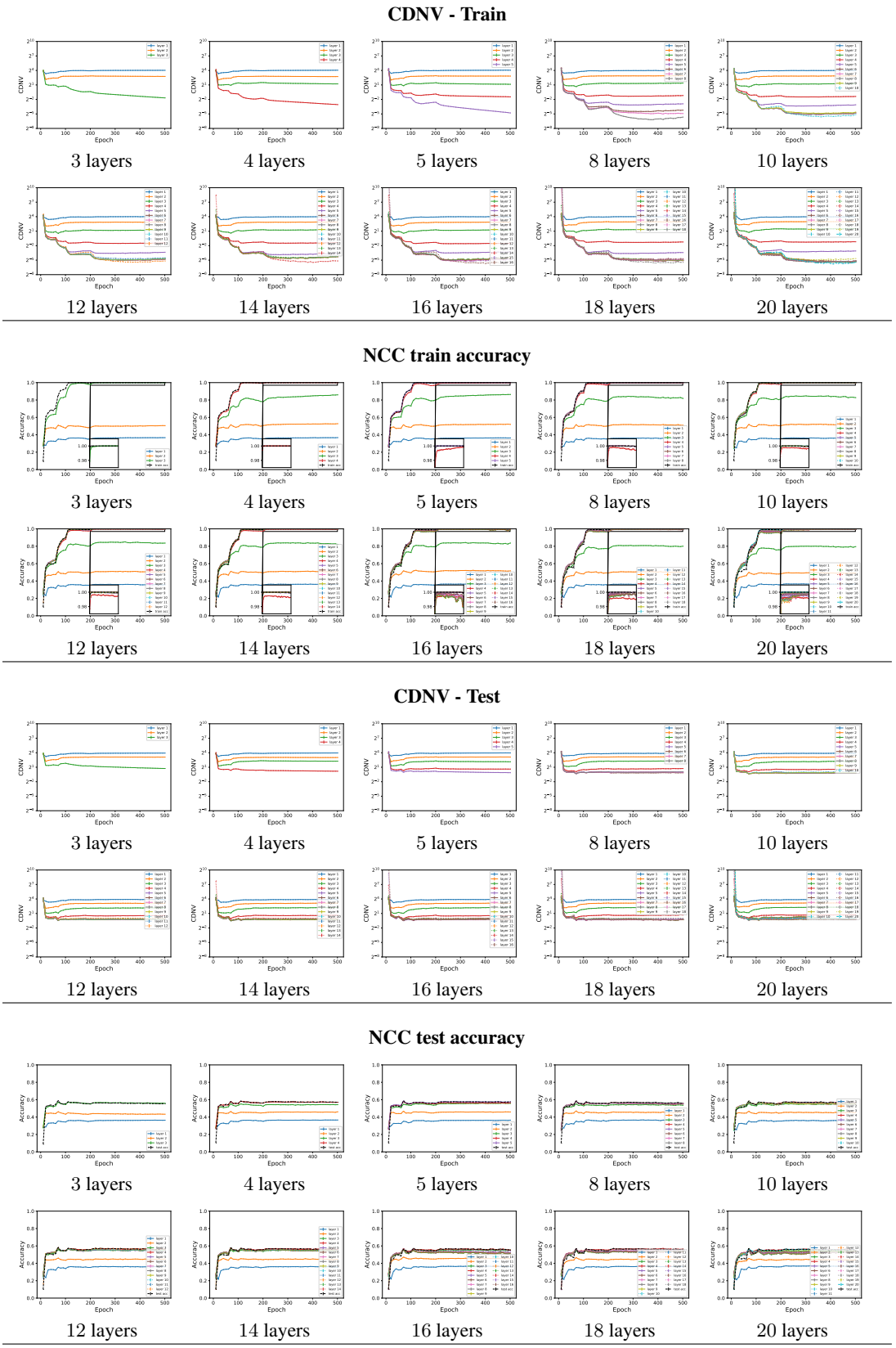

Figure 6: **Intermediate neural collapse of MLP-L-300 trained on CIFAR10.** See Fig. 1 in the main text for details.

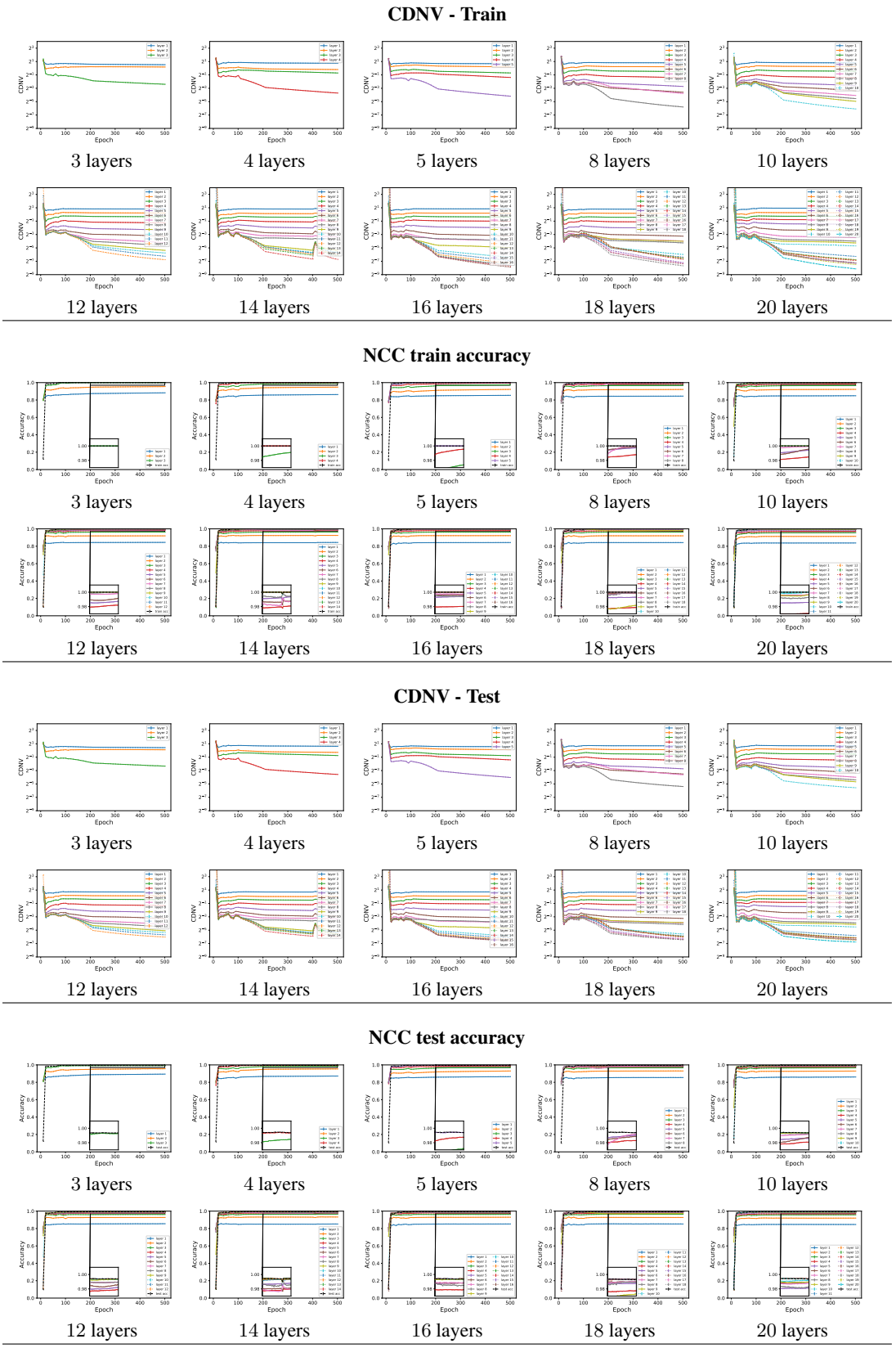

Figure 7: **Intermediate neural collapse of CONV-L-50 trained on MNIST.** See Fig. 1 in the main text for details.

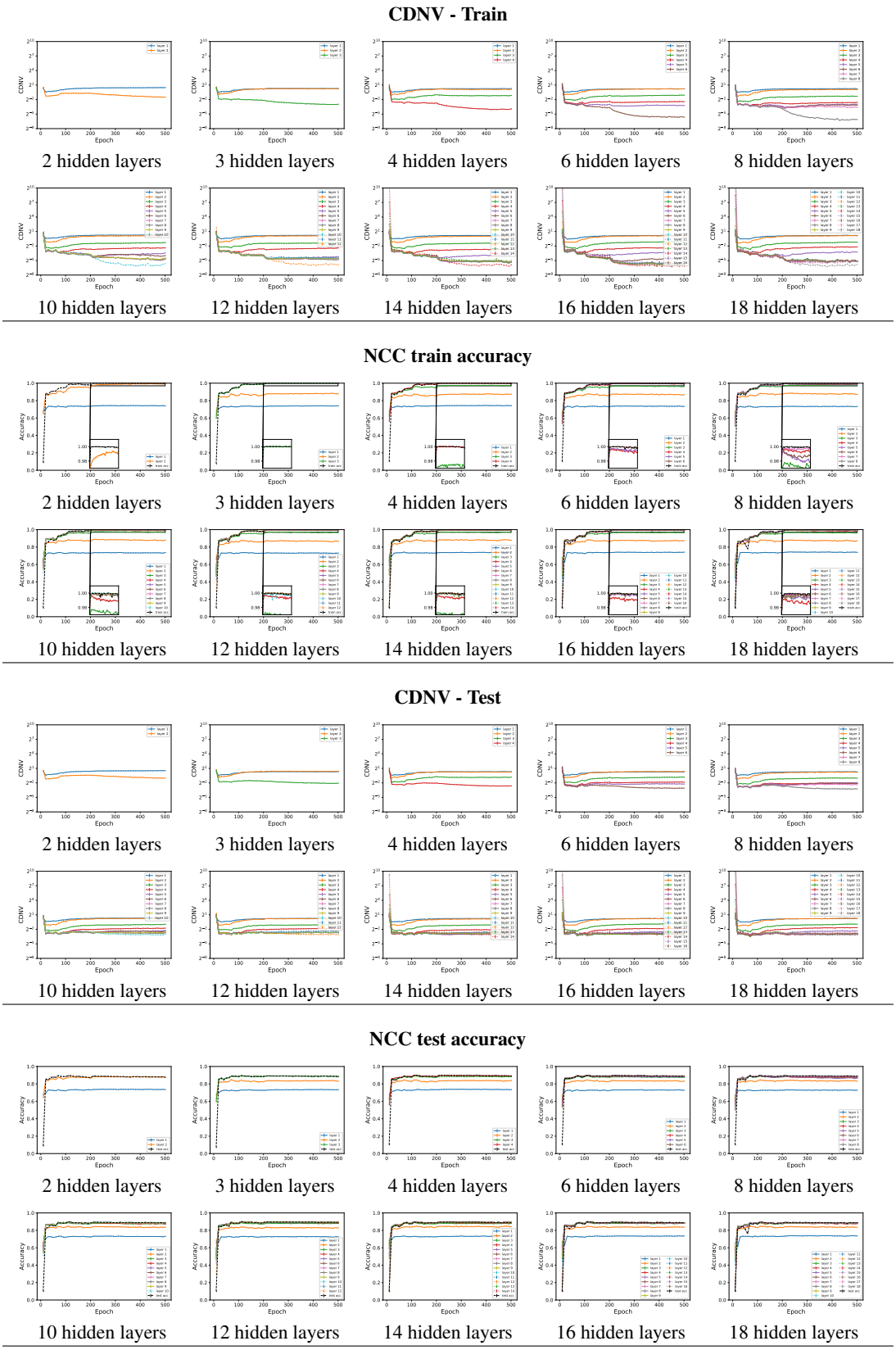

Figure 8: **Intermediate neural collapse of MLP-L-100 trained on Fashion MNIST.** See Fig. 1 in the main text for details.

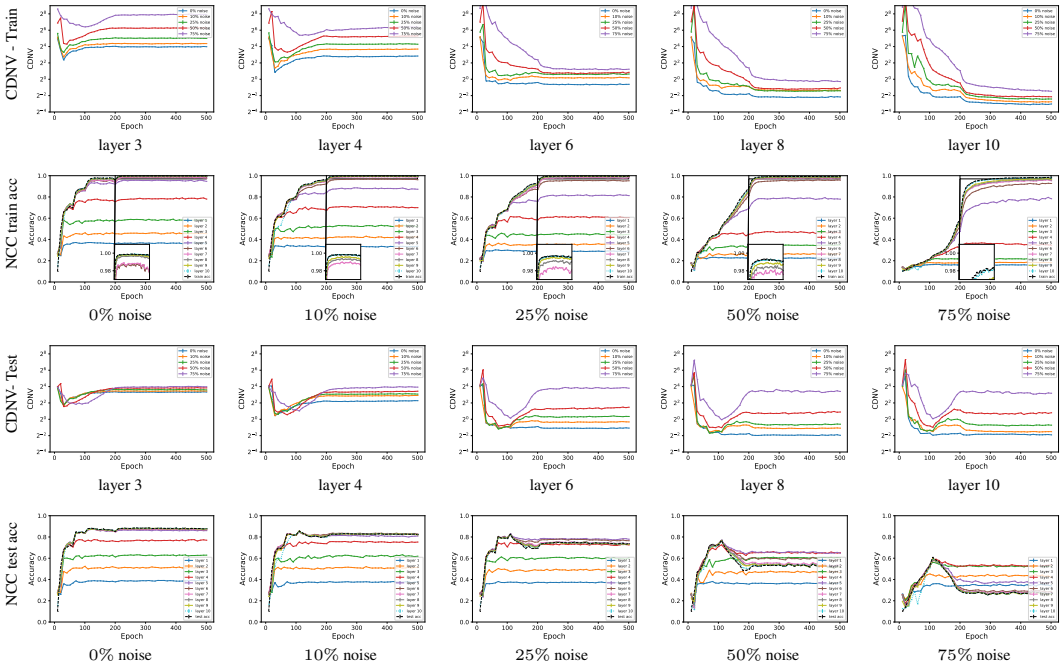

Figure 9: **Intermediate neural collapse of CONV-10-400 trained on CIFAR10 with partially corrupted labels.** In the first (third) row, we plot the CDNV on the train (test) data for intermediate layers of networks trained with varying amounts of corrupted labels (see legend). In the second (fourth) row, we plot the NCC accuracy rates of the various layers of a network trained with a certain amount of corrupted labels (see titles).

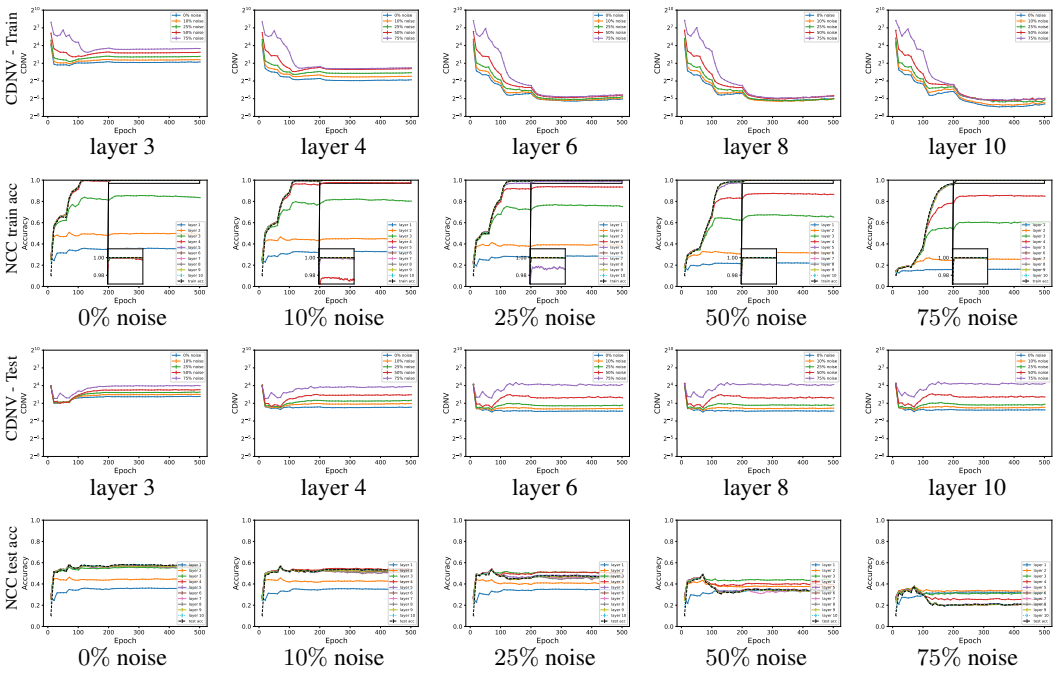

Figure 10: **Intermediate neural collapse of MLP-10-500 trained on CIFAR10 with noisy labels.** See Fig 3 in the main text for details.

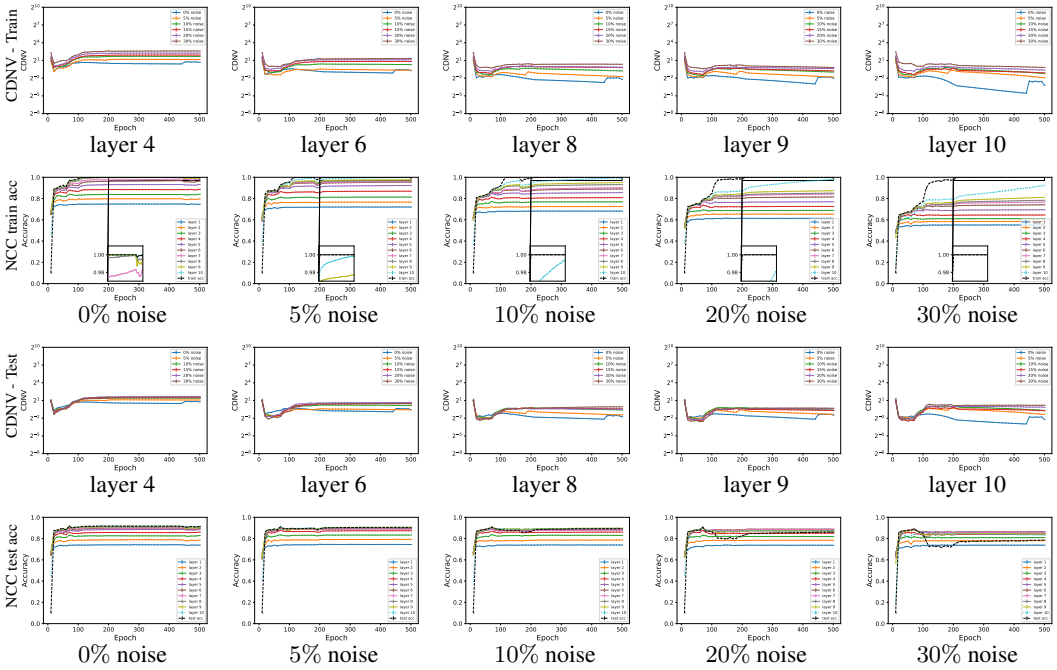

Figure 11: **Intermediate neural collapse of CONV-10-100 trained on Fashion MNIST with noisy labels.** See Fig. 3 in the main text for details.

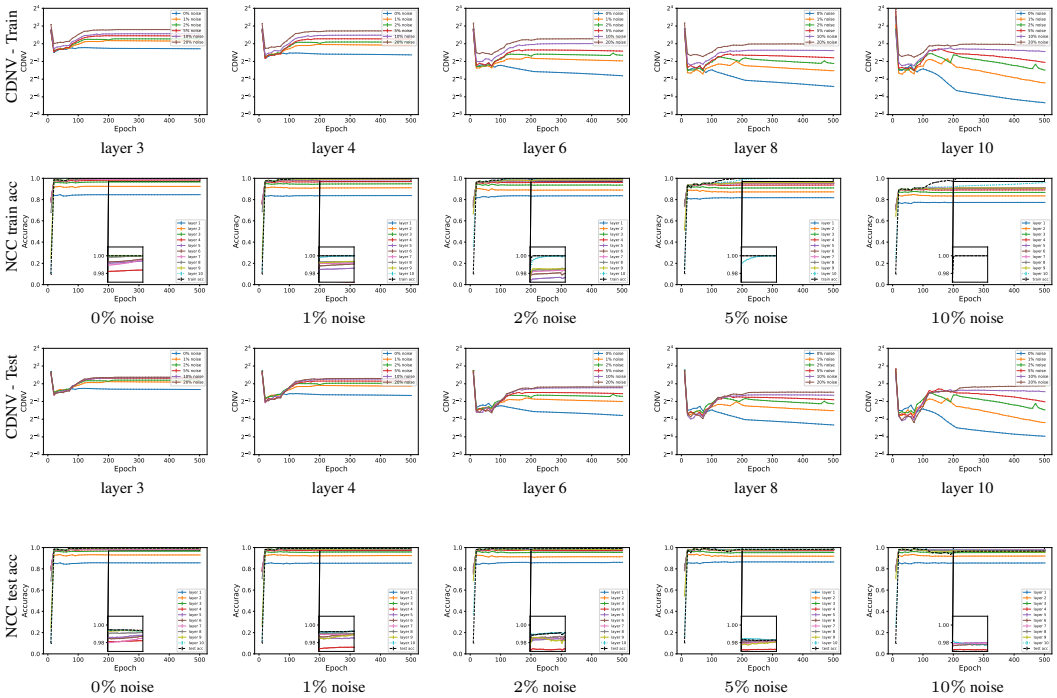

Figure 12: **Intermediate neural collapse of CONV-10-50 trained on MNIST with partially corrupted labels.** See Fig. 9 for details.

B PROOFS

Proposition 1. *Let $m \in \mathbb{N}$, $p \in (0, 1/2)$, $\alpha \in (0,1)$ and $\epsilon \in (0,1)$. Assume that the error of the learning algorithm is δ^1_m-uniform. Assume that $S_1, S_2 \sim P_B(m)$. Let $h^\gamma_{S_1}$ be the output of the learning algorithm given access to a dataset S_1 and initialization γ. Then,*

$$\begin{aligned}\mathbb{E}_{S_1}\mathbb{E}_\gamma[\text{err}_P(h^\gamma_{S_1})] \;\le\; \mathbb{P}_{S_1,S_2,\tilde{Y}_2}\left[\mathbb{E}_\gamma[\mathcal{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \;\ge\; \mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right] \\ + (1+\alpha)p + \delta^1_m + \delta^2_{m,p,\alpha},\end{aligned} \tag{4}$$

where $\tilde{Y}_2 = \{\tilde{Y}_i\}_{i=1}^m$ is uniformly selected to be a set of labels that disagrees with Y_2 on pm values.

Proof. Let $S_1 = \{(x^1_i, y^1_i)\}_{i=1}^m$ and $S_2 = \{(x^2_i, y^2_i)\}_{i=1}^m$ be two balanced datasets. Let $\epsilon > 0, p > 0$ and $q \ge (1+\alpha)\,p$. Let $\tilde{Y}_2$ and $\hat{Y}_2$ be a uniformly selected set of labels that disagree with Y_2 on pm and qm randomly selected labels (resp.). We denote by $\tilde{S}_2$ and $\hat{S}_2$ the relabeling of S_2 with the labels in $\tilde{Y}_2$ and in $\hat{Y}_2$ (resp.). We define four different events,

$$\begin{aligned}
A_1 &= \{(S_1, S_2, \tilde{Y}_2) \mid \exists\, q \ge (1+\alpha)\,p : \mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2) > \mathbb{E}_{\hat{Y}_2}[\mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \hat{S}_2)]\} \\
A_2 &= \{(S_1, S_2) \mid \text{the mistakes of } h^\gamma_{S_1} \text{ are not uniform over } S_2\} \\
A_3 &= \{(S_1, S_2, \tilde{Y}_2) \mid (S_1, S_2, \tilde{Y}_2) \notin A_1 \cup A_2 \text{ and } \mathbb{E}_\gamma[\mathcal{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \;<\; \mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\} \\
A_4 &= \{(S_1, S_2, \tilde{Y}_2) \mid (S_1, S_2, \tilde{Y}_2) \notin A_1 \cup A_2 \text{ and } \mathbb{E}_\gamma[\mathcal{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \;\ge\; \mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\} \\
B_1 &= \{(S_1, S_2, \tilde{Y}_2) \mid \mathbb{E}_\gamma[\mathcal{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \;\ge\; \mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\}
\end{aligned} \tag{7}$$

By the law of total expectation

$$\begin{aligned}
\mathbb{E}_{S_1}\mathbb{E}_\gamma[\text{err}_P(h^\gamma_{S_1})] &= \mathbb{E}_{S_1,S_2}\mathbb{E}_\gamma[\text{err}_{S_2}(h^\gamma_{S_1})] \\
&= \sum_{i=1}^4 \mathbb{P}[A_i] \cdot \mathbb{E}_{S_1,S_2,\tilde{Y}_2}[\mathbb{E}_\gamma[\text{err}_{S_2}(h^\gamma_{S_1})] \mid A_i] \\
&\le \mathbb{P}[A_1] + \mathbb{P}[A_2] + \mathbb{E}_{S_1,S_2,\tilde{Y}_2}[\text{err}_{S_2}(h^\gamma_{S_1}) \mid A_3] + \mathbb{P}[B_1],
\end{aligned} \tag{8}$$

where the last inequality follows from $\text{err}_{S_2}(h^\gamma_{S_1}) \le 1$, $\mathbb{P}[A_3] \le 1$ and $A_4 \subset B_1$.

We would like to upper bound each one of the above terms. First, we notice that since the mistakes of the network are δ^1_m-uniform, $\mathbb{P}[A_2] \le \delta^1_m$. In addition, by definition $\mathbb{P}[A_1] \le \delta^2_{m,p,\alpha}$.

As a next step, we upper bound $\mathbb{E}_{S_1,S_2,\tilde{Y}_2}[\text{err}_{S_2}(h^\gamma_{S_1}) \mid A_3]$. Assume that $(S_1, S_2, \tilde{Y}_2) \in A_3$. Hence, $(S_1, S_2, \tilde{Y}_2) \notin A_1 \cup A_2$. Then, the mistakes of $h^\gamma_{S_1}$ over S_2 are uniformly distributed (with respect to the selection of γ). Assume by contradiction that $q_m := \text{err}_{S_2}(h^\gamma_{S_1}) > (1+\alpha)\,p$ for some initialization γ. Then, since the mistakes of $h^\gamma_{S_1}$ over S_2 are uniformly distributed, $q_m = \text{err}_{S_2}(h^\gamma_{S_1}) > (1+\alpha)\,p$ for all initializations γ. Therefore, we have

$$\mathbb{E}_{\hat{Y}_2}[\mathcal{d}^\epsilon_{\min}(\mathcal{F}, S_1 \cup \hat{S}_2)] \;\le\; \mathbb{E}_\gamma[\mathcal{d}^\epsilon_{S_1}(h^\gamma_{S_1})] \;<\; \mathcal{d}^\epsilon_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2),$$

where the first inequality follows from the definition of $\mathcal{d}^\epsilon_{\min}(\mathcal{F}, S_1 \cup \hat{S}_2)$ and the second one by the assumption that $(S_1, S_2, \tilde{Y}_2) \in A_3$. However, this inequality contradicts the fact that $(S_1, S_2, \tilde{Y}_2) \notin A_1$. Therefore, we conclude that in this case, $q = \mathbb{E}_\gamma[\text{err}_{S_2}(h^\gamma_{S_1})] \le (1+\alpha)\,p$ and $\mathbb{E}_{S_1,S_2,\tilde{Y}_2}[\text{err}_{S_2}(h_{S_1}) \mid A_3] \le (1+\alpha)\,p$. □

Proposition 2. *Let $m \in \mathbb{N}$, $p \in (0, 1/2)$, $\alpha \in (0,1)$ and $\epsilon \in (0,1)$. Assume that the error of the learning algorithm is δ^1_m-uniform. Let $S_1, S_2, S^i_1, S^i_2 \sim P_B(m)$ (for $i \in [k]$). Let $\tilde{Y}^i_2 = \{\tilde{y}_i\}_{i=1}^m$ be a set of labels that disagrees with Y^i_2 on uniformly selected pm labels and $\tilde{S}^i_2$ is a relabeling of S_2 with the labels in $\tilde{Y}^i_2$. Let $h^\gamma_{S_1}$ be the output of the learning algorithm given access to a dataset S_1 and initialization γ. Then, with probability at least $1-\delta$ over the selection of $\{(S^i_1, S^i_2, \tilde{Y}^i_2)\}_{i=1}^k$, we*

have

$$\mathbb{E}_{S_1}\mathbb{E}_{\gamma}[\text{err}_P(h^{\gamma}_{S_1})] \;\leq\; \frac{1}{k}\sum_{i=1}^{k}\mathbb{I}\left[\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1^i}(h^{\gamma}_{S_1^i})] \;\geq\; \mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1^i \cup \tilde{S}_2^i)\right] + (1+\alpha)\,p + \delta^1_m + \delta^2_{m,p,\alpha} + \sqrt{\frac{\log(2/\delta)}{2k}}.$$

Proof. By Prop. 1, we have

$$\mathbb{E}_{S_1}\mathbb{E}_{\gamma}[\text{err}_P(h^{\gamma}_{S_1})] \;\leq\; \mathbb{P}_{S_1,S_2,\tilde{Y}_2}\left[\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})] \;\geq\; \mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right] + (1+\alpha)\,p_m + \delta^1_m + \delta^2_{m,p,\alpha}$$

We define i.i.d. random variables

$$V_i \;=\; \mathbb{I}\left[\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1^i}(h^{\gamma}_{S_1^i})] \;\geq\; \mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1^i \cup \tilde{S}_2^i)\right]. \tag{9}$$

Therefore, we can rewrite,

$$\mathbb{P}_{S_1,S_2,\tilde{Y}_2}\left[\mathbb{E}_{\gamma}[\mathscr{d}^{\epsilon}_{S_1}(h^{\gamma}_{S_1})] \;\geq\; \mathscr{d}^{\epsilon}_{\min}(\mathcal{G}, S_1 \cup \tilde{S}_2)\right] \;=\; \mathbb{E}[V_1] \tag{10}$$

By Hoeffding's inequality,

$$\mathbb{P}\left[\left|k^{-1}\sum_{i=1}^{k}V_i - \mathbb{E}[V_1]\right| \;\geq\; \epsilon\right] \;\leq\; 2\exp(-2k\epsilon^2). \tag{11}$$

By choosing $\epsilon = \sqrt{\log(1/2\delta)/2k}$, we obtain that with probability at least $1-\delta$, we have

$$\mathbb{E}[V_1] \;\leq\; \frac{1}{k}\sum_{i=1}^{k}V_i + \sqrt{\log(1/2\delta)/2k}. \tag{12}$$

When combined with Prop. 1, we obtain the desired bound. □