Prevalence of neural collapse during the terminal phase of deep learning training

Vardan Papyan[a,1], X. Y. Han[b,1], and David L. Donoho[a,2]

[a]Department of Statistics, Stanford University, Stanford, CA 94305-4065; and [b]School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14850

Contributed by David L. Donoho, August 18, 2020 (sent for review July 22, 2020; reviewed by Helmut Boelsckei and Stéphane Mallat)

Modern practice for training classification deepnets involves a terminal phase of training (TPT), which begins at the epoch where training error first vanishes. During TPT, the training error stays effectively zero, while training loss is pushed toward zero. Direct measurements of TPT, for three prototypical deepnet architectures and across seven canonical classification datasets, expose a pervasive inductive bias we call *neural collapse* (NC), involving four deeply interconnected phenomena. (NC1) Cross-example within-class variability of last-layer training activations collapses to zero, as the individual activations themselves collapse to their class means. (NC2) The class means collapse to the vertices of a simplex equiangular tight frame (ETF). (NC3) Up to rescaling, the last-layer classifiers collapse to the class means or in other words, to the simplex ETF (i.e., to a self-dual configuration). (NC4) For a given activation, the classifier's decision collapses to simply choosing whichever class has the closest train class mean (i.e., the nearest class center [NCC] decision rule). The symmetric and very simple geometry induced by the TPT confers important benefits, including better generalization performance, better robustness, and better interpretability.

deep learning | inductive bias | adversarial robustness | simplex equiangular tight frame | nearest class center

1. Introduction

Over the last decade, deep learning systems have steadily advanced the state of the art in benchmark competitions, culminating in superhuman performance in tasks ranging from image classification to language translation to game play. One might expect the trained networks to exhibit many particularities—making it impossible to find any empirical regularities across a wide range of datasets and architectures. On the contrary, in this article we present extensive measurements across image classification datasets and architectures, exposing a common empirical pattern.

Our observations focus on today's standard training paradigm in deep learning, an accretion of several fundamental ingredients that developed over time. Networks are trained beyond zero misclassification error, approaching negligible cross-entropy loss and interpolating the in-sample training data; networks are overparameterized, making such memorization possible; and these parameters are layered in ever-growing depth, allowing for sophisticated feature engineering. A series of recent works (1–5) highlighted the paradigmatic nature of the practice of training well beyond zero error, seeking zero loss. We call the postzero-error phase the terminal phase of training (TPT).

A scientist with standard preparation in mathematical statistics might anticipate that the linear classifier resulting from this paradigm, being a by-product of such training, would be quite arbitrary and vary wildly—from instance to instance, dataset to dataset, and architecture to architecture—thereby displaying no underlying cross-situational invariant structure. The scientist might further expect that the configuration of the fully trained decision boundaries—and the underlying linear classifier defining those boundaries—would be quite arbitrary and vary chaotically from situation to situation. Such expectations might be supported by appealing to the overparameterized nature of the model, and to standard arguments whereby any noise in the data propagates during overparameterized training to generate disproportionate changes in the parameters being fit.

Defeating such expectations, we show here that TPT frequently induces an underlying mathematical simplicity to the trained deepnet model—and specifically to the classifier and last-layer activations—across many situations now considered canonical in deep learning. Moreover, the identified structure naturally suggests performance benefits. Additionally, indeed, we show that convergence to this rigid structure tends to occur simultaneously with improvements in the network's generalization performance as well as adversarial robustness.

We call this process **neural collapse** (NC) and characterize it by four manifestations in the classifier and last-layer activations:

(NC1) Variability collapse: as training progresses, the within-class variation of the activations becomes negligible as these activations collapse to their class means.

(NC2) Convergence to simplex equiangular tight frame (ETF): the vectors of the class means (after centering by their global mean) converge to having equal length, forming equal-sized angles between any given pair, and being the maximally pairwise-distanced configuration constrained to the previous two properties. This configuration is identical to a previously studied configuration in the

Significance

Modern deep neural networks for image classification have achieved superhuman performance. Yet, the complex details of trained networks have forced most practitioners and researchers to regard them as black boxes with little that could be understood. This paper considers in detail a now-standard training methodology: driving the cross-entropy loss to zero, continuing long after the classification error is already zero. Applying this methodology to an authoritative collection of standard deepnets and datasets, we observe the emergence of a simple and highly symmetric geometry of the deepnet features and of the deepnet classifier, and we document important benefits that the geometry conveys—thereby helping us understand an important component of the modern deep learning training paradigm.

Author contributions: V.P., X.Y.H., and D.L.D. designed research, performed research, analyzed data, provided mathematical analysis, and wrote the paper.

Reviewers: H.B., ETH Zurich; and S.M., Collège de France.

The authors declare no competing interest.

See online for related content such as Commentaries.

[1]V.P. and X.Y.H. contributed equally to this work.

[2]To whom correspondence may be addressed. Email: donoho@stanford.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2015509117/-/DCSupplemental.

First published September 21, 2020.

mathematical sciences known as simplex ETF (6). See *Definition (Simplex ETF)*.

(NC3) **Convergence to self-duality**: the class means and linear classifiers—although mathematically quite different objects, living in dual-vector spaces—converge to each other, up to rescaling. Combined with (NC2), this implies a complete symmetry in the network classifiers' decisions: each isoclassifier-decision region is isometric to any other such region by rigid Euclidean motion; moreover the class means are each centrally located within their own specific regions, so there is no tendency toward higher confusion between any two classes than any other two.

(NC4) **Simplification to nearest class center (NCC)**: for a given deepnet activation, the network classifier converges to choosing whichever class has the nearest train class mean (in standard Euclidean distance).

We give a visualization of the phenomena (NC1) to (NC3) in Fig. 1* and define simplex ETFs (NC2) more formally as follows.

Definition (Simplex ETF): A standard simplex ETF is a collection of points in $\mathbb{R}^C$ specified by the columns of

$$\boldsymbol{M}^\star = \sqrt{\frac{C}{C-1}}\left(\boldsymbol{I} - \frac{1}{C}\mathbb{1}\mathbb{1}^\top\right), \qquad [1]$$

where $\boldsymbol{I} \in \mathbb{R}^{C\times C}$ is the identity matrix, and $\mathbb{1}_C \in \mathbb{R}^C$ is the ones vector. In this paper, we allow other poses, as well as rescaling, so the general simplex ETF consists of the points specified by the columns of $\boldsymbol{M} = \alpha\boldsymbol{U}\boldsymbol{M}^\star \in \mathbb{R}^{p\times C}$, where $\alpha \in \mathbb{R}_+$ is a scale factor, and $\boldsymbol{U} \in \mathbb{R}^{p\times C}$ $(p \geq C)$ is a partial orthogonal matrix $(\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I})$.

Properties (NC1) to (NC4) show that a highly symmetric and rigid mathematical structure with clear interpretability arises spontaneously during deep learning feature engineering, identically across many different datasets and model architectures.

(NC2) implies that the different feature means are "equally spaced" around the sphere in their constructed feature space; (NC3) says the same for the linear classifiers in their own dual space and moreover, that the linear classifiers are "the same as" the class means, up to possible rescaling. These mathematical symmetries and rigidities vastly simplify the behavior and analysis of trained classifiers, as we show in Section 5 below, which contrasts the kind of qualitative understanding previously available from theory against the precise and highly constrained predictions possible with (NC4).

(NC1) to (NC4) offer theoretically established performance benefits: stability against random noise and against adversarial noise. Additionally, indeed, this theory bears fruit. We show that during TPT, while NC is progressing, the trained models are improving in generalizability and in adversarial robustness.

In Section 7 below, we discuss the broader significance of (NC1) to (NC4) and their relation to recent advances across several rapidly developing "research fronts."

To support our conclusions, we conduct empirical studies that range over seven canonical classification datasets (7–9), including ImageNet (10), and three prototypical, contest-winning architectures (11–13). These datasets and networks were chosen for their prevalence in the literature as benchmarks (14–16), reaffirmed by their easy availability as part of the popular deep learning framework PyTorch (17). As explained below, these observations have important implications for our understand-

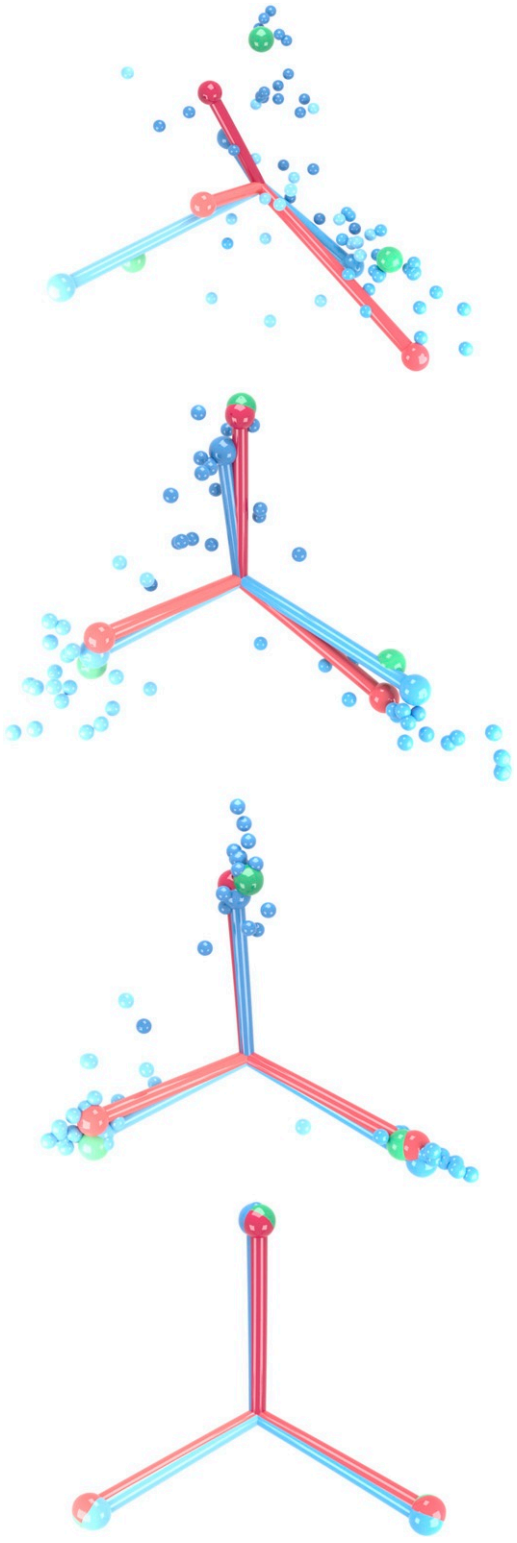

Fig. 1. Visualization of NC. The figure depicts, in three dimensions, NC as training proceeds from top to bottom. Green spheres represent the vertices of the standard simplex, see Definition (Simplex ETF), red ball and sticks represent linear classifiers, blue ball and sticks represent class means, and small blue spheres represent last-layer features. For all objects, we distinguish different classes via the shade of the color. As training proceeds, last-layer features collapse onto their class means (NC1), class means converge to the vertices of the simplex ETF (NC2), and the linear classifiers approach their corresponding class means (NC3). An animation can be found at https://purl.stanford.edu/br193mh4244.

*Fig. 1 is, in fact, generated using real measurements collected while training the VGG13 deepnet on CIFAR10. For three randomly selected classes, we extract the linear classifiers, class means, and a subsample of 20 last-layer features at epochs 2, 16, 65, and 350. These entities are then rotated, rescaled, and represented in three dimensions by leveraging the singular-value decomposition of the class means. We omit further details as Fig. 1 serves only to illustrate NC on an abstract level.

ing of numerous theoretical and empirical observations in deep learning.

2. Setting and Methodology

All subsequent experiments are built upon the general setting and methodology described below.

A. Image Classification. In the image classification problem, we are given a training dataset of d-dimensional images; the goal is to train a predictor to identify the class—of C total classes—to which any input image $\boldsymbol{x} \in \mathbb{R}^d$ belongs.

B. Deep Learning for Classification. In this work, we consider the predictor to be a deep neural network, which typically consists of numerous layers followed by a linear classifier. We view the layers before the classifier as computing a function, $\boldsymbol{x} \to \boldsymbol{h}(\boldsymbol{x})$, where $\boldsymbol{h} : \mathbb{R}^d \to \mathbb{R}^p$ outputs a p-dimensional feature vector. We refer to $\boldsymbol{h}(\boldsymbol{x})$ as the last-layer activations or last-layer features. The linear classifier takes as inputs the last-layer activations and outputs the class label. In detail, the linear classifier is specified by weights $\boldsymbol{W} \in \mathbb{R}^{C \times p}$ and biases $\boldsymbol{b} \in \mathbb{R}^C$, and the label the network attaches to image $\boldsymbol{x}$ is a simple function of $\boldsymbol{W}\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{b}$. In fact, it is $\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'}$ [i.e., the label is the index of the largest element in the vector $\boldsymbol{W}\boldsymbol{h}(\boldsymbol{x}) + \boldsymbol{b}$].

C. Network Architecture and Feature Engineering. The network is generally specified in two stages. First, an architecture is prescribed, and then, for a given architecture, there is a large number of parameters that determine the deep network's feature engineering, $\boldsymbol{h}(\boldsymbol{x})$. Collecting these parameters in a vector $\boldsymbol{\theta}$, we may also write $\boldsymbol{h} = \boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{x})$.

When the architecture specifies a truly deep network—and not merely a shallow one—the variety of behaviors that the different choices of $\boldsymbol{\theta}$ can produce is quite broad. To evoke, in quite concrete terms, the process of specifying the nonlinear transformation $\boldsymbol{x} \mapsto \boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{x})$, we speak of feature engineering. In contrast, traditional machine learning often dealt with a fixed collection of feature vectors that were not data adaptive.

D. Training. Viewing the induced class labels as the network outputs and the architecture and problem size as fixed in advance, the underlying class labeling algorithm depends on the parameter vector $(\boldsymbol{\theta}, \boldsymbol{W}, \boldsymbol{b})$. We think of $\boldsymbol{\theta}$ as determining the features to be used and $(\boldsymbol{W}, \boldsymbol{b})$ as determining the linear classifier that operates on the features to produce the labels. The number of parameters that must be determined is quite large. In practice, these parameters must be learned from data, by the process commonly known as training.

More concretely, consider a balanced dataset, having exactly N training examples in each class, $\bigcup_{c=1}^{C} \{\boldsymbol{x}_{i,c}\}_{i=1}^{N}$, where $\boldsymbol{x}_{i,c}$ denotes the ith example in the cth class. The parameters $(\boldsymbol{\theta}, \boldsymbol{W}, \boldsymbol{b})$ are fit by minimizing, usually using stochastic gradient descent (SGD), the objective function:

$$\min_{\boldsymbol{\theta}, \boldsymbol{W}, \boldsymbol{b}} \sum_{c=1}^{C} \sum_{i=1}^{N} \mathcal{L}\left(\boldsymbol{W}\boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{x}_{i,c}) + \boldsymbol{b}, \boldsymbol{y}_c\right). \qquad [2]$$

Above, we denote by $\mathcal{L} : \mathbb{R}^C \times \mathbb{R}^C \to \mathbb{R}^+$ the cross-entropy loss function and by $\boldsymbol{y}_c \in \mathbb{R}^C$ one-hot vectors (i.e., vectors containing one in the cth entry and zero elsewhere). We refer to this quantity as the training loss and the number of incorrect class predictions made by the network as the training error. Notice that, in TPT, the loss is nonzero even if the classification error is zero.

E. Datasets. We consider the MNIST, FashionMNIST, CIFAR10, CIFAR100, SVHN, STL10, and ImageNet datasets (7–10). MNIST was subsampled to $N = 5{,}000$ examples per class, SVHN to $N = 4{,}600$ examples per class, and ImageNet to $N = 600$ examples per class. The remaining datasets are already balanced. The images were preprocessed, pixel-wise, by subtracting the mean and dividing by the SD. No data augmentation was used.

F. Networks. We train the VGG, ResNet, and DenseNet architectures (11–13). For each of the three architecture types, we chose the network depth through trial and error in a series of preparatory experiments in order to adapt to the varying difficulties of the datasets. The final chosen networks were VGG19, ResNet152, and DenseNet201 for ImageNet; VGG13, ResNet50, and DenseNet250 for STL10; VGG13, ResNet50, and DenseNet250 for CIFAR100; VGG13, ResNet18, and DenseNet40 for CIFAR10; VGG11, ResNet18, and DenseNet250 for FashionMNIST; and VGG11, ResNet18, and DenseNet40 for MNIST and SVHN. DenseNet201 and DenseNet250 were trained using the memory-efficient implementation proposed in ref. 18. We replaced the dropout layers in VGG with batch normalization and set the dropout rate in DenseNet to zero.

G. Optimization Methodology. Following common practice, we minimize the cross-entropy loss using SGD with momentum 0.9. The weight decay is set to 1×10^{-4} for ImageNet and 5×10^{-4} for the other datasets. ImageNet is trained with a batch size of 256, across eight graphical processing units (GPUs), and the other datasets are trained on a single GPU with a batch size of 128. We train ImageNet for 300 epochs and the other datasets for 350 epochs. The initial learning is annealed by a factor of 10 at $1/2$ and $3/4$ for ImageNet and $1/3$ and $2/3$ for the other datasets. We sweep over 10 logarithmically spaced learning rates for ImageNet between 0.01 and 0.25 and 25 learning rates for the remaining datasets between 0.0001 and 0.25—picking the model resulting in the best test error in the last epoch.

H. Large-Scale Experimentation. The total number of models fully trained for this paper is tallied below:

ImageNet: 1 dataset × 3 nets × 10 lrs = 30 models.
Remainder: 6 datasets × 3 nets × 25 lrs = 450 models.
Total: 480 models.

The massive computational experiments reported here were run painlessly using ClusterJob and ElastiCluster (19–21) on the Stanford Sherlock HPC (high performance computing) cluster and Google Compute Engine virtual machines.

I. Moments of Activations. During training, we snapshot the network parameters at certain epochs. For each snapshotted epoch, we pass the train images through the network, extract their last-layer activations [using PyTorch hooks (22)], and calculate these activations' first and second moment statistics.

For a given dataset–network combination, we calculate the train global mean $\boldsymbol{\mu}_G \in \mathbb{R}^p$:

$$\boldsymbol{\mu}_G \triangleq \operatorname*{Ave}_{i,c} \{\boldsymbol{h}_{i,c}\},$$

and the train class means $\boldsymbol{\mu}_c \in \mathbb{R}^p$:

$$\boldsymbol{\mu}_c \triangleq \operatorname*{Ave}_{i} \{\boldsymbol{h}_{i,c}\}, \quad c = 1, \ldots, C,$$

where Ave is the averaging operator.

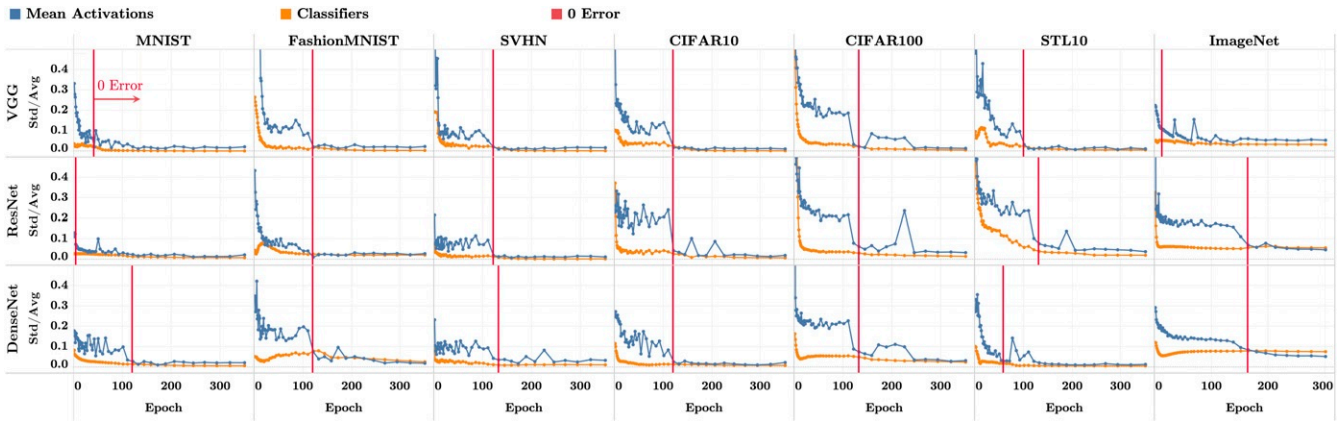

Fig. 2. Train class means become equinorm. The formatting and technical details are as described in Section 3. In each array cell, the vertical axis shows the coefficient of variation of the centered class-mean norms as well as the network classifiers norms. In particular, the blue lines show $\text{Std}_c(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2)/\text{Avg}_c(\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2)$ where $\{\boldsymbol{\mu}_c\}$ are the class means of the last-layer activations of the training data and $\boldsymbol{\mu}_G$ is the corresponding train global mean; the orange lines show $\text{Std}_c(\|\boldsymbol{w}_c\|_2)/\text{Avg}_c(\|\boldsymbol{w}_c\|_2)$ where $\boldsymbol{w}_c$ is the last-layer classifier of the cth class. As training progresses, the coefficients of variation of both class means and classifiers decrease.

Unless otherwise specified, for brevity, we refer in the text to the globally centered class means, $\{\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\}_{c=1}^C$, as just class means since the globally centered class means are of more interest.

Given the train class means, we calculate the train total covariance $\boldsymbol{\Sigma}_T \in \mathbb{R}^{p\times p}$,

$$\boldsymbol{\Sigma}_T \triangleq \underset{i,c}{\text{Ave}} \left\{ (\boldsymbol{h}_{i,c} - \boldsymbol{\mu}_G)(\boldsymbol{h}_{i,c} - \boldsymbol{\mu}_G)^\top \right\},$$

the between-class covariance, $\boldsymbol{\Sigma}_B \in \mathbb{R}^{p\times p}$,

$$\boldsymbol{\Sigma}_B \triangleq \underset{c}{\text{Ave}} \{ (\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)(\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)^\top \}, \qquad [3]$$

and the within-class covariance, $\boldsymbol{\Sigma}_W \in \mathbb{R}^{p\times p}$,

$$\boldsymbol{\Sigma}_W \triangleq \underset{i,c}{\text{Ave}} \{ (\boldsymbol{h}_{i,c} - \boldsymbol{\mu}_c)(\boldsymbol{h}_{i,c} - \boldsymbol{\mu}_c)^\top \}. \qquad [4]$$

Recall from multivariate statistics that

$$\boldsymbol{\Sigma}_T = \boldsymbol{\Sigma}_B + \boldsymbol{\Sigma}_W.$$

J. Formalization of NC. With the above notation, we now present a more mathematical description of NC, where $\to$ indicates convergence as training progresses:

(NC1) Variability collapse: $\boldsymbol{\Sigma}_W \to \boldsymbol{0}$.

(NC2) Convergence to simplex ETF:

$$|\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 - \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2| \to 0 \quad \forall\, c, c'$$
$$\langle \tilde{\boldsymbol{\mu}}_c, \tilde{\boldsymbol{\mu}}_{c'} \rangle \to \frac{C}{C-1}\delta_{c,c'} - \frac{1}{C-1} \quad \forall\, c, c'.$$

(NC3) Convergence to self-duality:

$$\left\| \frac{\boldsymbol{W}^\top}{\|\boldsymbol{W}\|_F} - \frac{\dot{\boldsymbol{M}}}{\|\dot{\boldsymbol{M}}\|_F} \right\|_F \to 0. \qquad [5]$$

(NC4) Simplification to NCC:

$$\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'} \to \arg\min_{c'} \|\boldsymbol{h} - \boldsymbol{\mu}_{c'}\|_2,$$

where $\tilde{\boldsymbol{\mu}}_c = (\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)/\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2$ are the renormalized the class means, $\dot{\boldsymbol{M}} = [\boldsymbol{\mu}_c - \boldsymbol{\mu}_G, c = 1, \ldots, C] \in \mathbb{R}^{p\times C}$ is the matrix

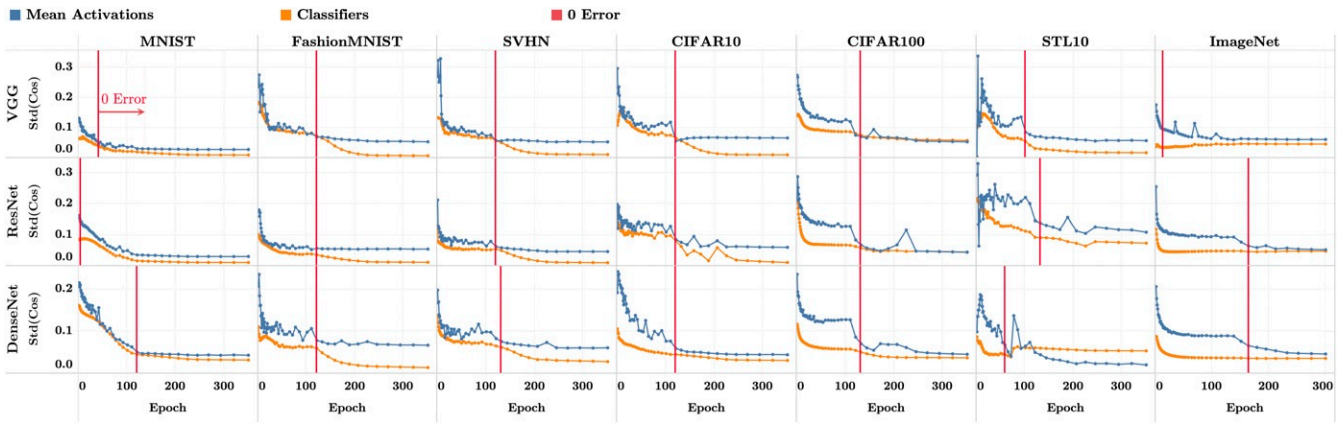

Fig. 3. Classifiers and train class means approach equiangularity. The formatting and technical details are as described in Section 3. In each array cell, the vertical axis shows the SD of the cosines between pairs of centered class means and classifiers across all distinct pairs of classes c and c'. Mathematically, denote $\cos_{\boldsymbol{\mu}}(c, c') = \langle \boldsymbol{\mu}_c - \boldsymbol{\mu}_G, \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G \rangle / (\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2$ and $\cos_{\boldsymbol{W}}(c, c') = \langle \boldsymbol{w}_c, \boldsymbol{w}_{c'} \rangle / (\|\boldsymbol{w}_c\|_2 \|\boldsymbol{w}_{c'}\|_2)$ where $\{\boldsymbol{w}_c\}_{c=1}^C$, $\{\boldsymbol{\mu}_c\}_{c=1}^C$, and $\boldsymbol{\mu}_G$ are as in Fig. 2. We measure $\text{Std}_{c,c'\neq c}(\cos_{\boldsymbol{\mu}}(c, c'))$ (blue) and $\text{Std}_{c,c'\neq c}(\cos_{\boldsymbol{W}}(c, c'))$ (orange). As training progresses, the SDs of the cosines approach zero, indicating equiangularity.

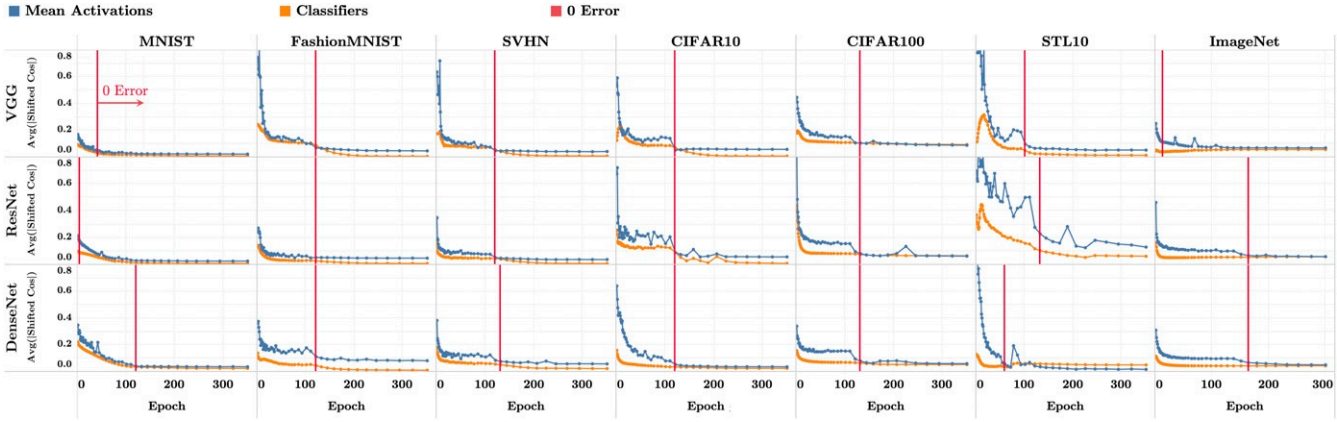

Fig. 4. Classifiers and train class means approach maximal-angle equiangularity. The formatting and technical details are as described in Section 3. We plot in the vertical axis of each cell the quantities $\text{Avg}_{c,c'}|\cos_\mu(c,c') + 1/(C-1)|$ (blue) and $\text{Avg}_{c,c'}|\cos_w(c,c') + 1/(C-1)|$ (orange), where $\cos_\mu(c,c')$ and $\cos_w(c,c')$ are as in Fig. 3. As training progresses, the convergence of these values to zero implies that all cosines converge to $-1/(C-1)$. This corresponds to the maximum separation possible for globally centered, equiangular vectors.

obtained by stacking the class means into the columns of a matrix, and $\delta_{c,c'}$ is the Kronecker delta symbol.

3. Results

To document the observations we make in Section 1, we provide a series of figures and a table below. We briefly list here our claims and identify the source of our evidence.

- Means and classifiers become equinorm: Fig. 2
- Means and classifiers become maximally equiangular: Figs. 3 and 4
- Means and classifiers become self-dual: Fig. 5
- Train within-class covariance collapses: Fig. 6
- Classifier approaches NCC: Fig. 7
- TPT improves robustness: Fig. 8
- TPT improves test error: Table 1

All figures in this article are formatted as follows. Each of the seven array columns is a canonical dataset for benchmarking classification performance—ordered left to right roughly by ascending difficulty. Each of the three array rows is a prototypical deep classifying network. On the horizontal axis of each cell is the epoch of training. For each dataset–network combination, the red vertical line marks the beginning of the effective beginning of TPT (i.e., the epoch when the training accuracy reaches 99.6% for ImageNet and 99.9% for the remaining datasets); we do not use 100% as it has been reported (24–26) that several of these datasets contain inconsistencies and mislabels, which sometimes prevent absolute memorization. Additionally, orange lines denote measurements on the network classifier, while blue lines denote measurements on the activation class means.

4. Discussion

Taken together, Figs. 2–7 give evidence for NC. First, Fig. 2 shows how, as training progresses, the variation in the norms of the class means (and classifiers) decreases—indicating that the class means (and classifiers) are converging to an equinormed state.

Then, Fig. 3 indicates that all pairs of class means (or classifiers) tend toward forming equal-sized angles. Fig. 4 additionally reveals that the cosines of these angles converge to $-\frac{1}{C-1}$—the maximum possible given the constraints. This maximal equiangularity, combined with equinormness, implies that the class means and classifiers converge to simplex ETFs.

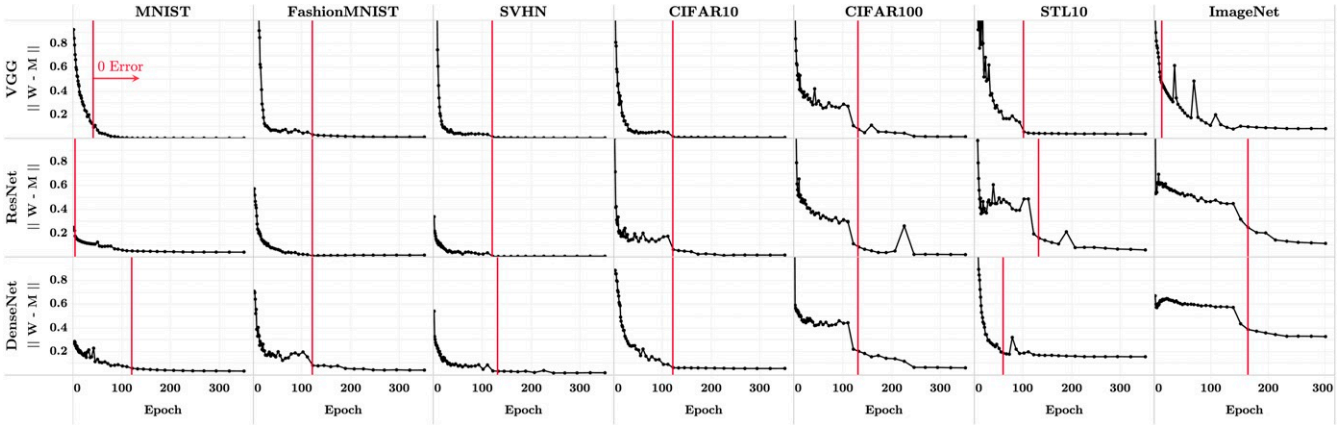

Fig. 5. Classifier converges to train class means. The formatting and technical details are as described in Section 3. In the vertical axis of each cell, we measure the distance between the classifiers and the centered class means, both rescaled to unit norm. Mathematically, denote $\widetilde{\boldsymbol{M}} = \dot{\boldsymbol{M}}/\|\dot{\boldsymbol{M}}\|_F$ where $\dot{\boldsymbol{M}} = [\boldsymbol{\mu}_c - \boldsymbol{\mu}_G : c = 1, \ldots, C] \in \mathbb{R}^{p\times C}$ is the matrix whose columns consist of the centered train class means; denote $\widetilde{\boldsymbol{W}} = \boldsymbol{W}/\|\boldsymbol{W}\|_F$ where $\boldsymbol{W} \in \mathbb{R}^{C\times p}$ is the last-layer classifier of the network. We plot the quantity $\|\widetilde{\boldsymbol{W}}^\top - \widetilde{\boldsymbol{M}}\|_F^2$ on the vertical axis. This value decreases as a function of training, indicating that the network classifier and the centered-means matrices become proportional to each other (self-duality).

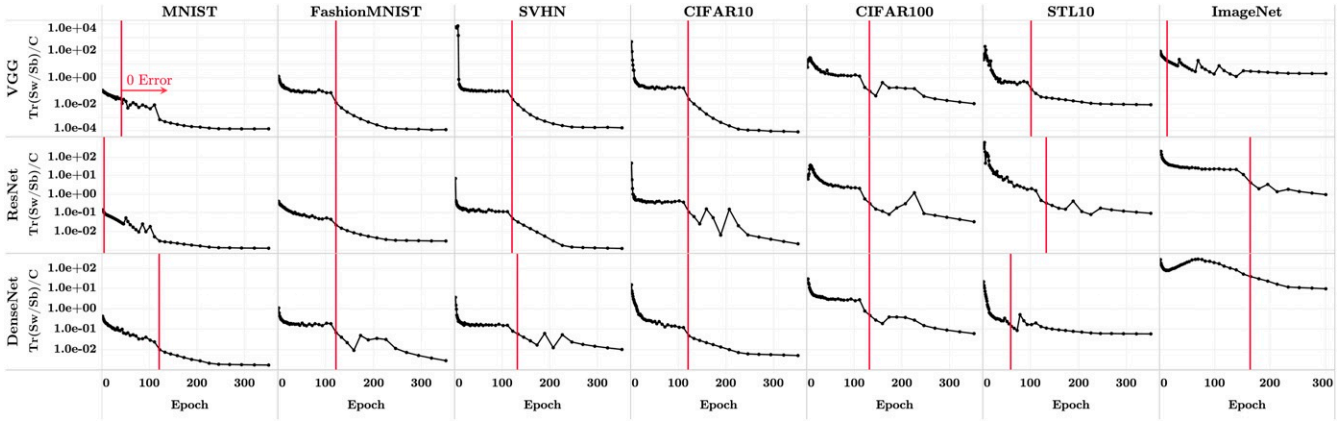

Fig. 6. Training within-class variation collapses. The formatting and technical details are as described in Section 3. In each array cell, the vertical axis (log scaled) shows the magnitude of the between-class covariance compared with the within-class covariance of the train activations. Mathematically, this is represented by $\mathrm{Tr}(\Sigma_W \Sigma_B^{\dagger}/C)$ where $\mathrm{Tr}(\cdot)$ is the trace operator, Σ_W is the within-class covariance of the last-layer activations of the training data, Σ_B is the corresponding between-class covariance, C is the total number of classes, and $[\cdot]^{\dagger}$ is Moore–Penrose pseudoinverse. This value decreases as a function of training—indicating collapse of within-class variation.

The above experiments by themselves do not indicate any relationship between the final converged states of the class means and classifiers, even though both converge to some simplex ETF. Such a relationship is revealed by Fig. 5—showing how they converge to the same simplex ETF, up to rescaling.

Moreover, the above concerns solely the class means. Yet, we can make a stronger claim about the activations themselves by looking at $\mathrm{Tr}(\Sigma_W \Sigma_B^{\dagger})$. This quantity, canonical in multivariate statistics, measures the inverse signal-to-noise ratio for classification problems and can be used to predict misclassification (27). The intraclass covariance matrix Σ_W (the noise) is best interpreted after scaled and rotated by pseudoinverse of the interclass covariance matrix Σ_B (the signal) since such transformation maps the noise into a common frame of reference across epochs. According to Fig. 6, the normalized variation of the activations becomes negligible as training proceeds, indicating that the activations collapse to their corresponding class means. This collapse continues well after the beginning of TPT.

A recurring theme across Figs. 2–7 is the continuing process of NC after zero error has been achieved. This explains TPT's paradigmatic nature: while continuing training after zero error has already been achieved seems counterintuitive, it induces significant changes in the underlying structure of the trained network.

This motivates Fig. 8 and Table 1, which explore two of the benefits of TPT. Table 1 shows how the test accuracy continues improving steadily, and Fig. 8 shows how adversarial robustness continues improving as well. In fact, most of the improvement in robustness happens during TPT.

5. NC Sharpens Previous Insights

Two notable prior works (28, 29) were able to significantly constrain the form of the structure of trained classifiers. However, in the presence of NC, it is possible to say dramatically more about the structure of trained classifiers. Moreover, the structure that emerges is extremely simple and symmetric.

In particular, the prior work assumed fixed features, not subject to data-adaptive feature engineering, which is so characteristic of deep learning training. In the modern context where deep learning features are trained and employing the assumption that the resulting last-layer entities undergo NC, the mathematical

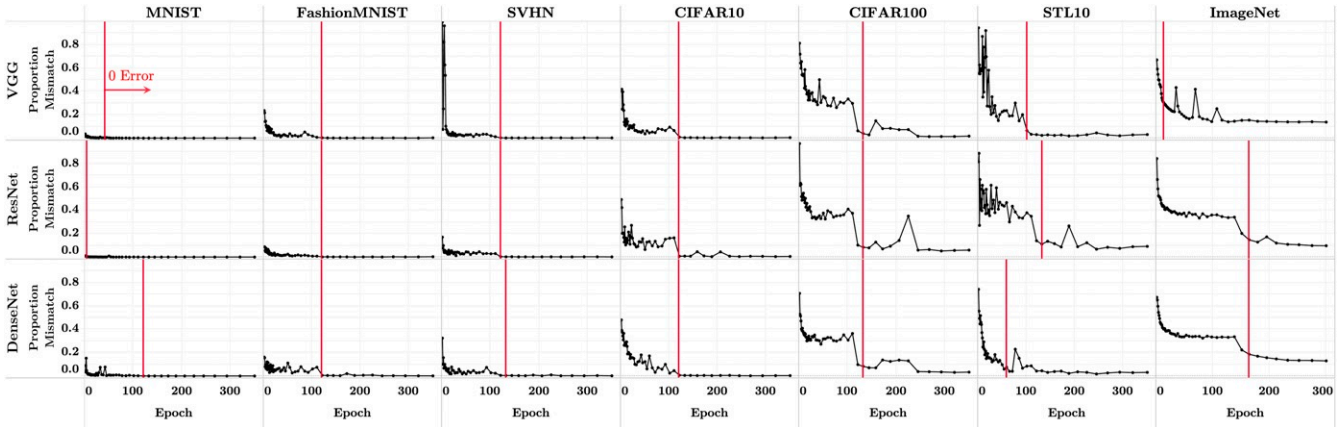

Fig. 7. Classifier behavior approaches that of NCC. The formatting and technical details are as described in Section 3. In each array cell, we plot the proportion of examples (vertical axis) in the testing set where network classifier disagrees with the result that would have been obtained by choosing $\arg\min_c \|\boldsymbol{h} - \boldsymbol{\mu}_c\|_2$ where $\boldsymbol{h}$ is a last-layer test activation, and $\{\boldsymbol{\mu}_c\}_{c=1}^{C}$ are the class means of the last-layer train activations. As training progresses, the disagreement tends to zero, showing the classifier's behavioral simplification to the nearest train class-mean decision rule.

constraints on the structure of trained classifiers tighten drastically.

As a preliminary step, we first formalize four possible properties of the end state toward which NC is tending:

$\overrightarrow{(\mathbf{NC1})}$ Variability collapse: $\boldsymbol{\Sigma}_W = \mathbf{0}$.

$\overrightarrow{(\mathbf{NC2})}$ Simplex ETF structure:

$$\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 = \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2 \quad \forall\, c, c'$$
$$\langle \tilde{\boldsymbol{\mu}}_c, \tilde{\boldsymbol{\mu}}_{c'} \rangle = \frac{C}{C-1}\delta_{c,c'} - \frac{1}{C-1}.$$

$\overrightarrow{(\mathbf{NC3})}$ Self-duality: $\frac{\boldsymbol{W}^\top}{\|\boldsymbol{W}\|_F} = \frac{\dot{\boldsymbol{M}}}{\|\dot{\boldsymbol{M}}\|_F}$.

$\overrightarrow{(\mathbf{NC4})}$ Behavioral equivalence to NCC:

$$\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'} = \arg\min_{c'} \|\boldsymbol{h} - \boldsymbol{\mu}_{c'}\|_2.$$

where $\tilde{\boldsymbol{\mu}}_c = (\boldsymbol{\mu}_c - \boldsymbol{\mu}_G)/\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2$ are the renormalized and centered class means, $\dot{\boldsymbol{M}} = [\mu_c - \mu_G : c = 1, \ldots, C] \in \mathbb{R}^{p \times C}$ is the matrix obtained by stacking the centered class means into the columns of a matrix, and $\delta_{c,c'}$ is the Kronecker delta symbol.

A. Webb and Lowe (28). In ref. 28, Webb and Lowe proved the following important result, which reformulated for the modern setting, could be written as follows.

Proposition 1 (Section 3 in ref. 28). *Fix the deepnet architecture and the underlying tuning parameters $\boldsymbol{\theta}$, so that the activations $\boldsymbol{h}_{\boldsymbol{\theta}}(\boldsymbol{x})$ involve no training and so that only the classifier weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ need to be trained. Maintain the same definitions—$\boldsymbol{\Sigma}_T$, $\boldsymbol{\mu}_c$, $\boldsymbol{\mu}_G$, etc.—as in Section 2. Adopting the mean squared error loss in place of the cross-entropy loss, the optimal classifier weights and biases are given by*

$$\boldsymbol{W} = \frac{1}{C}\dot{\boldsymbol{M}}^\top \boldsymbol{\Sigma}_T^\dagger$$
$$\boldsymbol{b} = \frac{1}{C}\mathbb{1}_C - \frac{1}{C}\dot{\boldsymbol{M}}^\top \boldsymbol{\Sigma}_T^\dagger \boldsymbol{\mu}_G, \qquad \mathbf{[6]}$$

where † denotes the Moore–Penrose pseudoinverse, $\dot{\boldsymbol{M}} = [\boldsymbol{\mu}_c - \boldsymbol{\mu}_G : c = 1, \ldots, C] \in \mathbb{R}^{p \times C}$ is the matrix obtained by stacking the centered class means into the columns of a matrix, and $\mathbb{1}_C \in \mathbb{R}^C$ is a vector of ones.

The form in **[6]** is similar to the one first developed by R. A. Fisher in 1936 (30)—commonly referred to as linear discriminant analysis (LDA)—although Fisher's version uses $\boldsymbol{\Sigma}_W$ in lieu of $\boldsymbol{\Sigma}_T$. In other words, the above theorem states that a modified LDA is the optimal solution for the last-layer classifier.

The result of Webb and Lowe (28) admirably elucidates the structure of the optimal classifier; however, it also leaves a great deal unspecified about possible properties of the classifier. In our Theorem 1, immediately following, we supplement the assumptions of Webb and Lowe (28) by adding the variability collapse and simplex ETF properties; the result significantly narrows the possible structure of optimal classifiers, obtaining both self-duality and behavioral agreement with NCC.

Theorem 1 [Proposition 1 + $\overrightarrow{(\text{NC1-2})}$ Imply $\overrightarrow{(\text{NC3-4})}$]. *Adopt the framework and assumptions of Proposition 1, as well as the end state implied by* (NC1) *and* (NC2) [*i.e.*, $\overrightarrow{(\text{NC1})}$ *and* $\overrightarrow{(\text{NC2})}$]. *The Webb–Lowe classifier* **[6]**, *in this setting, has the additional properties* $\overrightarrow{(\text{NC3})}$ *and* $\overrightarrow{(\text{NC4})}$.

Proof: By $\overrightarrow{(\text{NC1})}$, $\boldsymbol{\Sigma}_W = \mathbf{0}$, and we have $\boldsymbol{\Sigma}_T = \boldsymbol{\Sigma}_B$. Using now Proposition 1, we obtain

$$\boldsymbol{W} = \frac{1}{C}\dot{\boldsymbol{M}}^\top \boldsymbol{\Sigma}_B^\dagger$$
$$\boldsymbol{b} = \frac{1}{C}\mathbb{1}_C - \frac{1}{C}\dot{\boldsymbol{M}}^\top \boldsymbol{\Sigma}_B^\dagger \boldsymbol{\mu}_G.$$

[3] implies that $\boldsymbol{\Sigma}_B = \frac{1}{C}\dot{\boldsymbol{M}}\dot{\boldsymbol{M}}^\top$. Thus,

$$\boldsymbol{W} = \dot{\boldsymbol{M}}^\top \left(\dot{\boldsymbol{M}}\dot{\boldsymbol{M}}^\top\right)^\dagger = \dot{\boldsymbol{M}}^\dagger$$
$$\boldsymbol{b} = \frac{1}{C}\mathbb{1}_C - \dot{\boldsymbol{M}}^\top \left(\dot{\boldsymbol{M}}\dot{\boldsymbol{M}}^\top\right)^\dagger \boldsymbol{\mu}_G = \frac{1}{C}\mathbb{1}_C - \dot{\boldsymbol{M}}^\dagger \boldsymbol{\mu}_G.$$

$\overrightarrow{(\text{NC2})}$ implies that $\dot{\boldsymbol{M}}$ has exactly $C-1$ nonzero and equal singular values, so $\dot{\boldsymbol{M}}^\dagger = \alpha\dot{\boldsymbol{M}}^\top$ for some constant α. Combining the previous pair of displays, we obtain

$$\boldsymbol{W} = \alpha\dot{\boldsymbol{M}}^\top \qquad \mathbf{[7a]}$$
$$\boldsymbol{b} = \frac{1}{C}\mathbb{1}_C - \alpha\dot{\boldsymbol{M}}^\top \boldsymbol{\mu}_G; \qquad \mathbf{[7b]}$$

[7a] demonstrates the asserted self-duality $\overrightarrow{(\text{NC3})}$, up to rescaling. The class predicted by the above classifier is given by

$$\begin{aligned}
&\arg\max_{c'} \langle \boldsymbol{w}_{c'}, \boldsymbol{h} \rangle + b_{c'} \\
&= \arg\max_{c'} \alpha \langle \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G, \boldsymbol{h} \rangle + \frac{1}{C} - \alpha \langle \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G, \boldsymbol{\mu}_G \rangle \\
&= \arg\max_{c'} \langle \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G, \boldsymbol{h} - \boldsymbol{\mu}_G \rangle .
\end{aligned}$$

Using the equal norm property of $\overrightarrow{(\text{NC2})}$, this display becomes

$$\begin{aligned}
&\arg\max_{c'} \langle \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G, \boldsymbol{h} - \boldsymbol{\mu}_G \rangle \\
&= \arg\min_{c'} \|\boldsymbol{h} - \boldsymbol{\mu}_G\|_2^2 - 2\langle \boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G, \boldsymbol{h} - \boldsymbol{\mu}_G \rangle + \|\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G\|_2^2 \\
&= \arg\min_{c'} \|(\boldsymbol{h} - \boldsymbol{\mu}_G) - (\boldsymbol{\mu}_{c'} - \boldsymbol{\mu}_G)\|_2 \\
&= \arg\min_{c'} \|\boldsymbol{h} - \boldsymbol{\mu}_{c'}\|_2. \qquad \mathbf{[8]}
\end{aligned}$$

In words, the decision of the linear classifier based on $(\boldsymbol{W}, \boldsymbol{b})$ is identical to that made by NCC $\overrightarrow{(\text{NC4})}$.

Our theorem predicts that evidence of (NC1) and (NC2) as shown in Figs. 2–4 and 6 should deterministically accompany both (NC3) and (NC4)—exactly as observed in Figs. 5 and 7.

B. Soudry et al. (29). The authors of ref. 29 consider C-class classification, again in a setting where the parameter vector $\boldsymbol{\theta}$ is not trained, so that the last-layer activations $\boldsymbol{h}_{i,c} = \boldsymbol{h}(\boldsymbol{x}_{i,c})$ are fixed and not subject to feature engineering.

They proved an important result, which explicitly addresses our paper's focus on cross-entropy classifier loss minimization in the zero-error regime.

Proposition 2 (Theorem 7 in ref. 29). *Let $\mathbb{R}^{NC \times p}$ denote the vector space spanning all last-layer activation datasets, $\boldsymbol{H} = (\boldsymbol{h}_{i,c} : 1 \le i \le N, 1 \le c \le C)$, and let $\mathcal{H}$ denote the measurable subset of $\mathbb{R}^{NC \times p}$ consisting of linearly separable datasets: that is, consisting of datasets $\boldsymbol{H}$ where, for some linear classifiers $\{\boldsymbol{w}_c\}_{c=1}^C$ (possibly depending on dataset $\boldsymbol{H}$), separability holds:*

$$\langle \boldsymbol{w}_c - \boldsymbol{w}'_c, \boldsymbol{h}_{i,c} \rangle \ge 1, \qquad \boldsymbol{h}_{i,c} \in \boldsymbol{H}.$$

For (Lebesgue) almost every dataset $\boldsymbol{H} \in \mathcal{H}$, gradient descent minimizing the cross-entropy loss, as a function of the classifier weights,

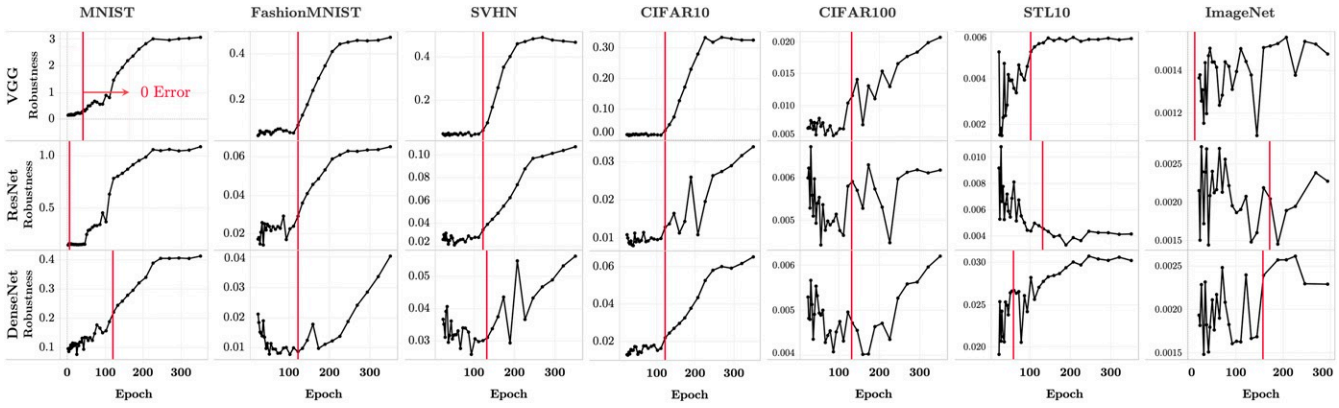

Fig. 8. Training beyond zero error improves adversarial robustness. The formatting and technical details are as described in Section 3. For each dataset and network, we sample without replacement 100 test images—constructing for each an adversarial example using the DeepFool method proposed in ref. 23. In each array cell, we plot on the vertical axis the robustness measure, $\text{Ave}_i \|\boldsymbol{r}(\boldsymbol{x}_i)\|_2 / \|\boldsymbol{x}_i\|_2$, from the same paper—where $\boldsymbol{r}(\boldsymbol{x}_i)$ is the minimal perturbation required to change the class predicted by the classifier, for a given input image $\boldsymbol{x}_i$. As training progresses, larger $\boldsymbol{x}_i$. As training progresses, larger perturbations are required to fool the deepnet. Across all array cells, the median improvement in the robustness measure in the last epoch over the first epoch achieving zero training error is 0.0252; the mean improvement is 0.2452.

tends to a limit. This limit is identical to the solution of the max-margin classifier problem:

$$\min_{\{\boldsymbol{w}_c\}_{c=1}^{C}} \sum_{c=1}^{C} \|\boldsymbol{w}_c\|_2^2 \text{ s.t. } \forall i,\, c,\, c' \neq c : \langle \boldsymbol{w}_c - \boldsymbol{w}_{c'}, \boldsymbol{h}_{i,c} \rangle \geq 1. \qquad [9]$$

This inspiring result significantly constrains the form of the trained classifier in precisely the cross-entropy loss setting relevant to deep learning. However, because feature activations are here fixed, not learned, this result is only able to give indirect, implicit information about a deepnet trained model involving feature engineering and about the classification decisions that it makes.

Some authors of ref. 29 have, in a series of highly influential talks and papers, laid great emphasis on the notion of an emergent, not superficially evident, "inductive bias" as a reason for the surprising success of deep learning training and have pointed to Proposition 2 as a foundational result indicating that inductive bias can be implicit in the behavior of a training procedure that superficially shows no behavioral tendencies in the indicated direction.

We agree wholeheartedly with the philosophy underlying ref. 29; our results support the further observation that inductive bias is far more constraining on the outcome of modern deepnet training than was previously known.

In effect, out of all possible max-margin classifiers that could be consistent with Proposition 2, the modern deepnet training paradigm is producing linear classifiers approximately belonging to the very tiny subset with the additional property of being simplex ETFs. Moreover, such classifiers exhibit very striking behavioral simplicity in decision making.

Theorem 2 (Proposition 2 + $\overrightarrow{\text{(NC1-2)}}$ Imply $\overrightarrow{\text{(NC3-4)}}$). *Adopt the framework and assumptions of Proposition* 2, *as well as the end state implied by* (NC1) *and* (NC2) [*i.e.*, $\overrightarrow{\text{(NC1)}}$ *and* $\overrightarrow{\text{(NC2)}}$]. *The Soudry et al.* (29) *classifier* [**9**], *in this setting, has the additional properties* $\overrightarrow{\text{(NC3)}}$ *and* $\overrightarrow{\text{(NC4)}}$.

Proof: Since $\dot{\boldsymbol{M}}$ is the matrix of a simplex ETF, it has $C-1$ equal-sized singular values with the remaining singular value being zero. Without loss of generality, we assume here that those singular values are 1 (i.e., $\|\dot{\boldsymbol{M}}\|_2 = 1$). Notice the singular value assumption implies that the columns are of norm $\|\boldsymbol{\mu}_c - \boldsymbol{\mu}_G\|_2 = \sqrt{(C-1)/C}$ (not unity) for all c. At the assumed end state of variability collapse $\overrightarrow{\text{(NC1)}}$, the activations all collapse to their respective class means, and the max-margin classifier problem reduces to

$$\min_{\{\boldsymbol{w}_c\}_{c=1}^{C}} \sum_{c=1}^{C} \frac{1}{2} \|\boldsymbol{w}_c\|_2^2 \text{ s.t. } \forall c,\, c' \neq c : \langle \boldsymbol{w}_c - \boldsymbol{w}_{c'}, \boldsymbol{\mu}_c - \boldsymbol{\mu}_G \rangle \geq 1.$$

Table 1. Training beyond zero error improves test performance

Dataset and network	Test accuracy at zero error	Test accuracy at last epoch
MNIST		
VGG	99.40	99.56
ResNet	99.32	99.71
DenseNet	99.65	99.70
FashionMNIST		
VGG	92.92	93.31
ResNet	93.29	93.64
DenseNet	94.18	94.35
SVHN		
VGG	93.82	94.53
ResNet	94.64	95.70
DenseNet	95.87	95.93
CIFAR10		
VGG	87.85	88.65
ResNet	88.72	89.44
DenseNet	91.14	91.19
CIFAR100		
VGG	63.03	63.85
ResNet	66.19	66.21
DenseNet	77.19	76.56
STL10		
VGG	65.15	68.00
ResNet	69.99	70.24
DenseNet	67.79	70.81
ImageNet		
VGG	47.26	50.12
ResNet	65.41	64.45
DenseNet	65.04	62.38

The median improvement of test accuracy at the last epoch over that at the first epoch achieving zero training error is 0.3495%; the mean improvement is 0.4984%.

Rewriting with matrix notation and using a Pythagorean decomposition of the objective, the above becomes

$$\min_{\boldsymbol{W}} \frac{1}{2}\|\boldsymbol{W}\dot{\boldsymbol{M}}\dot{\boldsymbol{M}}^{\dagger}\|_F^2 + \frac{1}{2}\|\boldsymbol{W}(\boldsymbol{I}-\dot{\boldsymbol{M}}\dot{\boldsymbol{M}}^{\dagger})\|_F^2$$
$$\text{s.t. } \forall c, c' \neq c : (\boldsymbol{e}_c - \boldsymbol{e}_{c'})^{\top}\boldsymbol{W}\dot{\boldsymbol{M}}\boldsymbol{e}_c \geq 1,$$

where † denotes the Moore–Penrose pseudoinverse of a matrix. Property $\overrightarrow{(\text{NC2})}$ fully specifies the Gram matrix of $\dot{\boldsymbol{M}}$, from which we know that $\dot{\boldsymbol{M}}$ has $C-1$ singular values all equal to one, WLOG (without loss of generality), and a right null space spanned by the vector of ones, $\mathbb{1}$, since its columns have zero mean. Thus, its singular value decomposition is given by $\dot{\boldsymbol{M}} = \boldsymbol{U}\boldsymbol{V}^{\top}$, where $\boldsymbol{U} \in \mathbb{R}^{p\times C-1}$ and $\boldsymbol{V} \in \mathbb{R}^{C\times C-1}$ are partial orthogonal matrices. Hence,

$$\min_{\boldsymbol{W}} \frac{1}{2}\|\boldsymbol{W}\boldsymbol{U}\boldsymbol{U}^{\top}\|_F^2 + \frac{1}{2}\|\boldsymbol{W}(\boldsymbol{I}-\boldsymbol{U}\boldsymbol{U}^{\top})\|_F^2$$
$$\text{s.t. } \forall c, c' \neq c : (\boldsymbol{e}_c - \boldsymbol{e}_{c'})^{\top}\boldsymbol{W}\boldsymbol{U}\boldsymbol{V}^{\top}\boldsymbol{e}_c \geq 1.$$

Observe that 1) the second term of the objective penalizes deviations of $\boldsymbol{w}_c$ from the column space of $\boldsymbol{U}$ and that 2) such deviations do not affect the constraints. Conclude that the optimal solution for the above optimization problem has the form $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{U}^{\top}$, where $\boldsymbol{A} \in \mathbb{R}^{C\times C-1}$. This simplification, as well as the fact that

$$\begin{aligned}\|\boldsymbol{W}\boldsymbol{U}\boldsymbol{U}^{\top}\|_F^2 &= \|\boldsymbol{A}\boldsymbol{U}^{\top}\boldsymbol{U}\boldsymbol{U}^{\top}\|_F^2 \\ &= \text{Tr}(\boldsymbol{A}\boldsymbol{U}^{\top}\boldsymbol{U}\boldsymbol{U}^{\top}\boldsymbol{U}\boldsymbol{U}^{\top}\boldsymbol{U}\boldsymbol{A}) = \|\boldsymbol{A}\|_F^2\end{aligned}$$

and $\boldsymbol{W}\boldsymbol{U} = \boldsymbol{A}\boldsymbol{U}^{\top}\boldsymbol{U} = \boldsymbol{A}$, transforms the optimization problem into the equivalent form

$$\min_{\boldsymbol{A}} \frac{1}{2}\|\boldsymbol{A}\|_F^2 \text{ s.t. } \forall c, c' \neq c : (\boldsymbol{e}_c - \boldsymbol{e}_{c'})^{\top}\boldsymbol{A}\boldsymbol{V}^{\top}\boldsymbol{e}_c \geq 1. \qquad [10]$$

Averaging the constraints of [10] over c and summing over $c' \neq c$, we obtain

$$\begin{aligned}C-1 &\leq \frac{1}{C}\sum_c ((C-1)\boldsymbol{e}_c - (\mathbb{1}-\boldsymbol{e}_c))^{\top}\boldsymbol{A}\boldsymbol{V}^{\top}\boldsymbol{e}_c \\ &= \frac{1}{C}\sum_c (C\boldsymbol{e}_c - \mathbb{1})^{\top}\boldsymbol{A}\boldsymbol{V}^{\top}\boldsymbol{e}_c \\ &= \sum_c \boldsymbol{e}_c^{\top}\left(\boldsymbol{I} - \frac{1}{C}\mathbb{1}\mathbb{1}^{\top}\right)\boldsymbol{A}\boldsymbol{V}^{\top}\boldsymbol{e}_c \\ &= \text{Tr}\left(\left(\boldsymbol{I} - \frac{1}{C}\mathbb{1}\mathbb{1}^{\top}\right)\boldsymbol{A}\boldsymbol{V}^{\top}\right) \\ &= \text{Tr}\left(\boldsymbol{A}\boldsymbol{V}^{\top}\left(\boldsymbol{I} - \frac{1}{C}\mathbb{1}\mathbb{1}^{\top}\right)\right) \\ &= \text{Tr}\left(\boldsymbol{A}\boldsymbol{V}^{\top}\right),\end{aligned}$$

where the last equality follows from $\boldsymbol{V}^{\top}\mathbb{1} = 0$. This leads to the following relaxation of [10]:

$$\min_{\boldsymbol{A}} \frac{1}{2}\|\boldsymbol{A}\|_F^2 \text{ s.t. } \text{Tr}\left(\boldsymbol{A}\boldsymbol{V}^{\top}\right) \geq C-1. \qquad [11]$$

Checking first-order conditions, the optimum occurs at $\boldsymbol{A} = \boldsymbol{V}$. Recalling that $\dot{\boldsymbol{M}}$ is a simplex ETF with singular values 1, $\boldsymbol{V}\boldsymbol{V}^{\top} = \boldsymbol{V}\boldsymbol{U}^{\top}\boldsymbol{U}\boldsymbol{V}^{\top} = \dot{\boldsymbol{M}}^{\top}\dot{\boldsymbol{M}} = \boldsymbol{I} - \frac{1}{C}\mathbb{1}\mathbb{1}^{\top}$. Because $\boldsymbol{A} = \boldsymbol{V}$, and $\boldsymbol{V}\boldsymbol{e}_c = \left(\boldsymbol{e}_c - \frac{1}{C}\mathbb{1}\right)$ for $c = 1, \ldots, C$,

$$(\boldsymbol{e}_c - \boldsymbol{e}_{c'})^{\top}\boldsymbol{A}\boldsymbol{V}^{\top}\boldsymbol{e}_c = (\boldsymbol{e}_c - \boldsymbol{e}_{c'})^{\top}\boldsymbol{V}\boldsymbol{V}^{\top}\boldsymbol{e}_c = 1. \qquad [12]$$

Since $\boldsymbol{A} = \boldsymbol{V}$ optimizes [11], which involves the same objective as [10] but over a possibly enlarged feasible set, feasibility of $\boldsymbol{A} = \boldsymbol{V}$ implies that $\boldsymbol{A} = \boldsymbol{V}$ optimizes [10] as well. The solution to [10] is unique since the problem minimizes a positive definite quadratic subject to a single nondegenerate linear constraint. In the optimization problem for $\boldsymbol{W}$ that we started with, recall that $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{U}^{\top}$. Hence, the optimality of $\boldsymbol{A} = \boldsymbol{V}$ implies $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{U}^{\top} = \boldsymbol{V}\boldsymbol{U}^{\top} = \dot{\boldsymbol{M}}^{\top}$, showing self-duality is achieved $\overrightarrow{(\text{NC3})}$. This equality becomes a proportionality in the more general case where the equal singular values of $\dot{\boldsymbol{M}}$ are not unity.

An argument similar to the one for Theorem 1 is that the classifier is behaviorally equivalent to the NCC decision rule $\overrightarrow{(\text{NC4})}$.

Much like Theorem 1, but now for cross-entropy loss, the above result again indicates that evidence of (NC1) and (NC2) as shown in Figs. 2–4 and 6 should accompany both (NC3) and (NC4), as shown in Figs. 5 and 7. In short, our results indicate an inductive bias toward NCC, which is far more total and limiting than the max-margin bias proposed by ref. 29.

6. Theoretical Derivation of Simplex ETF Emergence

We are unaware of suggestions, prior to this work, that simplex ETFs emerge as the solution of an interesting and relevant optimization problem. Prompted by the seemingly surprising nature of the above empirical results, we developed theoretical results that show that the observed end state of NC can be derived directly using standard ideas from information theory and probability theory. Roughly speaking, the simplex ETF property $\overrightarrow{(\text{NC2})}$, self-duality $\overrightarrow{(\text{NC3})}$, and behavioral simplification $\overrightarrow{(\text{NC4})}$ are derivable consequences of variability collapse $\overrightarrow{(\text{NC1})}$.

In our derivation, we consider an abstraction of feature engineering, in which an ideal feature designer chooses activations that minimize the classification error in the presence of nearly vanishing within-class variability. Our derivation shows that the ideal feature designer should choose activations whose class means form a simplex ETF.

A. Model Assumptions. Assume we are given an observation $\boldsymbol{h} = \boldsymbol{\mu}_{\gamma} + \boldsymbol{z} \in \mathbb{R}^C$, where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ and $\gamma \sim \text{Unif}\{1, \ldots, C\}$ is an unknown class index, distributed independently from $\boldsymbol{z}$. Our goal is to recover γ from $\boldsymbol{h}$, with as small an error rate as possible. We constrain ourselves to use a linear classifier, $\boldsymbol{W}\boldsymbol{h} + \boldsymbol{b}$, with weights $\boldsymbol{W} = [\boldsymbol{w}_c : c = 1, \ldots, c] \in \mathbb{R}^{C\times C}$ and biases $\boldsymbol{b} = (b_c) \in \mathbb{R}^C$; our decision rule is

$$\hat{\gamma}(\boldsymbol{h}) = \hat{\gamma}(\boldsymbol{h}; \boldsymbol{W}, \boldsymbol{b}) = \arg\max_c \langle \boldsymbol{w}_c, \boldsymbol{h}\rangle + b_c.$$

Our task is to design the classifier $\boldsymbol{W}$ and bias $\boldsymbol{b}$, as well as a matrix $\boldsymbol{M} = [\boldsymbol{\mu}_c : c = 1, \ldots, C] \in \mathbb{R}^{C\times C}$, subject to the norm constraints $\|\boldsymbol{\mu}_c\|_2 \leq 1$ for all c.

B. Information Theory Perspective. The above can be recast as an optimal coding problem in the spirit of Shannon (31). The class means $\boldsymbol{\mu}_c$ are code words, and the matrix $\boldsymbol{M}$ represents a codebook, containing C code words. A transmitter transmits a code word over a noisy channel, contaminated by white additive Gaussian noise, and then, a receiver obtains the noisy signal $\boldsymbol{h} = \boldsymbol{\mu}_c + \boldsymbol{z}$, which it then decodes using a linear decoder $\hat{\gamma}$ in an attempt to recover the transmitted γ. The norm constraint on the means captures limits imposed on signal strength due to the distance between the transmitter and receiver. Our task is to design a codebook and decoder that would allow optimal retrieval of the class identity γ from the noisy information $\boldsymbol{h}$.

C. Large Deviations Perspective. To measure success in this task, we consider the large deviations error exponent:

$$\beta(\boldsymbol{M},\boldsymbol{W},\boldsymbol{b}) = -\lim_{\sigma\to 0}\sigma^2\log P_\sigma\{\hat{\gamma}(\boldsymbol{h})\neq\gamma\}.$$

This is the right limit, as we are considering the situation where the noise is approaching zero due to variability collapse (NC1). Tools for deriving large deviations error exponents have been extensively developed in probability theory (32).

D. Theoretical Result. As a preliminary reduction, we can assume without loss of generality that the ambient vector space, in which the code words and observations lie, is simply $\mathbb{R}^C$ (*SI Appendix*).

Theorem 3. *Under the model assumptions just given in subsections A, B, and C, the optimal error exponent is*

$$\begin{aligned}\beta^\star &= \max_{\boldsymbol{M},\boldsymbol{W},\boldsymbol{b}}\beta(\boldsymbol{M},\boldsymbol{W},\boldsymbol{b})\ \text{ s.t. }\ \|\boldsymbol{\mu}_c\|_2\le 1\ \forall c\\ &= \frac{C}{C-1}\cdot\frac{1}{4},\end{aligned}$$

where the maximum is over $C\times C$ matrices $\boldsymbol{M}$ with at most unit-norm columns, and over $C\times C$ matrices $\boldsymbol{W}$ and $C\times 1$ vectors $\boldsymbol{b}$.

Moreover, denote $\boldsymbol{M}^\star=\sqrt{\frac{C}{C-1}}\left(\boldsymbol{I}-\frac{1}{C}\mathbb{1}\mathbb{1}^\top\right)$ (i.e., $\boldsymbol{M}^\star$ is the standard simplex ETF). The optimal error exponent is precisely achieved by $\boldsymbol{M}^\star$:

$$\beta(\boldsymbol{M}^\star,\boldsymbol{M}^\star,\boldsymbol{0})=\beta^\star.$$

All matrices $\boldsymbol{M}$ achieving $\beta^\star$ are also simplex ETFs—possibly in an isometric pose—deriving from $\boldsymbol{M}^\star$ via $\boldsymbol{M}=\boldsymbol{U}\boldsymbol{M}^\star$ with $\boldsymbol{U}$ a $C\times C$ orthogonal matrix. For such matrices, an optimal linear decoder is $\boldsymbol{W}=\boldsymbol{M}^\star\boldsymbol{U}^\top$, $\boldsymbol{b}=\boldsymbol{0}$:

$$\beta(\boldsymbol{M},\boldsymbol{W},\boldsymbol{b})=\beta(\boldsymbol{M},\boldsymbol{M}^\top,\boldsymbol{0})=\beta^\star.$$

Proof: The proof is given in *SI Appendix*.

In words, if we engineer a collection of code words to optimize the (vanishingly small) theoretical misclassification probability, we obtain as our solution the standard simplex ETF, or a rotation of it.

We stress that the maximal equiangularity property of $\boldsymbol{M}^\star$ is crucial to this result: that is,

$$\langle\boldsymbol{\mu}_c^\star,\boldsymbol{\mu}_{c'}^\star\rangle=\frac{-1}{C-1},\qquad c'\neq c;$$

this property is enjoyed by every collection of class means optimizing the error exponent and is unique to simplex ETFs.

The results of this section show that simplex ETFs are the unique solution to an abstract optimal feature design problem. The fact that modern deepnet training practice has found this same simplex ETF solution suggests to us that the training paradigm—SGD, TPT, and so on—is finding the same solution as would an ideal feature engineer! Future research should seek to understand the ability of training dynamics to succeed in obtaining this solution.

7. Related Works

The prevalence of NC makes us view a number of previous empirical and theoretical observations in a new fresh perspective.

A. Theoretical Feature Engineering. Immediately prior to the modern era of purely empirical deep learning, ref. 33 proposed a theory-derived machinery building on the scattering transform that promised an understandable approach for handwritten digit recognition. The theory was particularly natural for problems involving within-class variability caused by "small" morphings of class-specific templates; in fact, the scattering transform was shown in ref. 34 to tightly limit the variability caused by template morphings. Later, refs. 35–37 complemented ref. 33 with additional theory covering a larger range of mathematically derived features, nonlinearities, and pooling operations—again designed to suppress within-class variability.

Our finding of NC, specifically (NC1), shows that feature engineering by standard empirical deepnet training achieves similar suppression of within-class variability—both on the original dataset considered by ref. 33 as well as six more challenging benchmarks. Thus, the original goal of refs. 33–37, which can be phrased as the limiting of within-class variability of activations, turns out to be possible for a range of datasets and perhaps more surprisingly, to be learnable by SGD on cross-entropy loss. Recently, Mallat and collaborators (38) were able to deliver results with scattering-transform features (combined with dictionary learning) that rival the foundational empirical results produced by AlexNet. So apparently, controlling within-class activation variability, whether this is achieved analytically or empirical, is quite powerful.

B. Observed Structure of Spectral Hessians. More recently, empirical studies of the Hessian of the deepnet training loss of image classification networks observed surprising and initially baffling deterministic structure. First observed by refs. 39 and 40, on toy models, the spectrum exhibits C outlier eigenvalues separated from a bulk, where C is the number of classes of the image classification task. Refs. 41–43 corroborated these findings at scale on modern deep networks and large datasets. Refs. 41 and 42 explained how the spectral outliers could be attributed to low-rank structure associated with class means, and the bulk could be induced by within-class variations (of logit derivatives). It was essential that the class means have greater norm than the within-class SD in order for these spectral outliers to emerge.

Under (NC1), the full matrix of last-layer activations converges to a rank-$(C-1)$ matrix, associated with class means. So under (NC1), eventually the within-class SD will be much smaller, and the outliers will emerge from the bulk. In short, the collapse of activation variability (NC1), combined with convergence of class means (NC2) to the simplex ETF limit, explains these important and highly visible observations about deepnet Hessians.

C. Stability against Random and Adversarial Noise. It is well understood classically that when solving linear systems $\boldsymbol{y}=\boldsymbol{M}\boldsymbol{x}$ by standard methods, some matrices $\boldsymbol{M}$ are prone to solution instability, blowing up small noise in $\boldsymbol{y}$ to produce large noise in $\boldsymbol{x}$; other matrices are less prone. Stability problems arise if the nonzero singular values of $\boldsymbol{M}$ are vastly different and do not arise if the nonzero singular values are all identical. The simplex ETF offers equal nonzero singular values and so, a certain resistance to noise amplification. This is a less well-known path to equal singular values, partial orthogonal matrices being of course the more well known.

In the deepnet literature, the authors of refs. 44–48 studied the stability of deepnets to adversarial examples. They proposed that stability can be obtained by making the matrices defined by the network weights close to orthogonal. However, no suggestion was offered for why trained weights, under the current standard training paradigm, would tend to become orthogonal.

In ref. 49, the authors modified the standard training paradigm, forcing linear and convolutional layers to be approximate tight frames; they showed that this leads both to better robustness to adversarial examples, as well as improved accuracy and faster training. To get these benefits, they imposed orthogonality explicitly during training.

In refs. 44, 45, and 49, the authors showed how concerns about stability can be addressed by explicit interventions in the standard training paradigm. By demonstrating a pervasive simplex ETF structure, this paper has shown that, under today's standard training paradigm, deepnets naturally achieve an implicit form of stability in the last layer. In light of the previous discussions of the benefits of equal singular values, we of course expected that the trained deep network would become more robust to adversaries, as the training progresses toward the simplex ETF. The measurements we reported here support this prediction, and evidence in ref. 50 gives further credence to this hypothesis.

8. Conclusion

This paper studied the TPT of today's canonical deepnet training protocol. It documented that during TPT a process called NC takes place, involving four fundamental and interconnected phenomena: (NC1) to (NC4).

Prior to this work, it was becoming apparent, due to ref. 29 and related work, that the last-layer classifier of a trained deepnet exhibits appreciable mathematical structure—a phenomenon called inductive bias, which was gaining ever-wider visibility. Our work exposes considerable additional fundamental and we think, surprising structure: 1) the last-layer features are not only linearly separable but actually collapsed to a C-dimensional simplex ETF, and 2) the last-layer classifier is behaviorally equivalent to the NCC decision rule. Through our thorough experimentation on seven canonical datasets and three prototypical networks, we show that these phenomena persist across the range of canonical deepnet classification problems. Furthermore, we document that convergence to this simple structure aids in the improvement of out-of-sample network performance and robustness to adversarial examples. We hypothesize that the benefits of the interpolatory regime of overparametrized networks are directly related to NC.

From a broader perspective, the standard workflow of empirical deep learning can be viewed as a series of arbitrary steps that happened to help win prediction challenge contests, which were then proliferated by their popularity among contest practitioners. Careful analysis, providing a full understanding of the effects and benefits of each workflow component, was never the point. One of the standard workflow practices is training beyond zero error to zero loss (i.e., TPT). In this work, we give a clear understanding that TPT benefits today's standard deep learning training paradigm by showing how it leads to the pervasive phenomenon of NC. Moreover, this work puts older results on a different footing, expanding our understanding of their contributions. Finally, because of the precise mathematics and geometry, the doors are open for new formal insights.

Data Availability. Experimental measurements have been deposited in the Stanford Digital Repository, https://purl.stanford.edu/ng812mz4543. An animation can be found at https://purl.stanford.edu/br193mh4244.

ACKNOWLEDGMENTS. This work was partially supported by NSF Division of Mathematical Sciences Grants 1407813, 1418362, and 1811614 and by private donors. Some of the computing for this project was performed on the Sherlock cluster at Stanford University; we thank the Stanford Research Computing Center for providing computational resources and support that enabled our research. Some of this project was also performed on Google Cloud Platform: thanks to Google Cloud Platform Education Grants Program for research credits that supplemented this work. Moreover, we thank Riccardo Murri and Hatef Monajemi for their extensive help with the Elasticluster and ClusterJob frameworks, respectively.

1. S. Ma, R. Bassily, M. Belkin, "The power of interpolation: Understanding the effectiveness of SGD in modern over-parametrized learning" in *International Conference on Machine Learning* (Proceedings of Machine Learning Research [PMLR], Cambridge, MA, 2018), pp. 3325–3334.
2. M. Belkin, A. Rakhlin, A. B. Tsybakov, "Does data interpolation contradict statistical optimality?" in *The 22nd International Conference on Artificial Intelligence and Statistics* (Proceedings of Machine Learning Research [PMLR], Cambridge, MA, 2019), pp. 1611–1619.
3. M. Belkin, D. J. Hsu, P. Mitra, "Overfitting or perfect fitting? Risk bounds for classification and regression rules that interpolate" in *Advances in Neural Information Processing Systems* (Curran, Vancouver, Canada, 2018), pp. 2300–2311.
4. M. Belkin, D. Hsu, S. Ma, S. Mandal, Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proc. Natl. Acad. Sci. U.S.A.* **116**, 15849–15854 (2019).
5. M. Belkin, *Beyond Empirical Risk Minimization: The Lessons of Deep Learning* (colloquium recording, MIT CBMM, 2019).
6. T. Strohmer, R. W. Heath, Grassmannian frames with applications to coding and communication. *Appl. Comput. Harmon. Anal.* **14**, 257–275 (2003).
7. A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images. Citeseer. https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf. (8 April 2009).
8. Y. LeCun, C. Cortes, C. Burges, MNIST handwritten digit database. AT&T Labs (2010). http://yann.lecun.com/exdb/mnist/. Accessed 1 July 2020.
9. H. Xiao, K. Rasul, R. Vollgraf, Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms arXiv:1708.07747v1 (25 August 2017).
10. J. Deng *et al.*, "Imagenet: A large-scale hierarchical image database" in *IEEE Conference on Computer Vision and Pattern Recognition, 2009* (IEEE, 2009), pp. 248–255.
11. K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 770–778.
12. G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, "Densely connected convolutional networks" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2017), pp. 4700–4708.
13. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 (4 September 2014).
14. Benchmarks.AI, Directory of AI benchmarks (2020). https://benchmarks.ai/. Accessed 1 July 2020.
15. Stanford University, Stanford DAWN (2020). https://dawn.cs.stanford.edu/. Accessed 1 July 2020.
16. GitHub, Who is best at X? (2020). https://rodrigob.github.io/are_we_there_yet/build/. Accessed 1 July 2020.
17. A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library" in *Advances in Neural Information Processing Systems* (Curran, Vancouver, Canada 2019), pp. 8026–8037.
18. G. Pleiss *et al.*, Memory-efficient implementation of DenseNets arXiv:1707.06990 (21 July 2017).
19. H. Monajemi, D. L. Donoho, ClusterJob: An automated system for painless and reproducible massive computational experiments (2015). https://github.com/monajemi/clusterjob. Accessed 1 July 2020.
20. H. Monajemi, D. L. Donoho, V. Stodden, "Making massive computational experiments painless" in *2016 IEEE International Conference on Big Data (Big Data)* (IEEE, 2016), pp. 2368–2373.
21. H. Monajemi *et al.*, Ambitious data science can be painless. *Harvard Data Sci. Rev.*, 10.1162/99608f92.02ffc552 (2019).
22. PyTorch, Forward and backward function hooks—PyTorch documentation. https://pytorch.org/tutorials/beginner/former_torchies/nnft_tutorial.html#forward-and-backward-function-hooks. Accessed 21 June 2020 .
23. S. M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2016), pp. 2574–2582.
24. R. Ekambaram, D. B. Goldgof, L. O. Hall, "Finding label noise examples in large scale datasets" in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (IEEE, 2017), pp. 2420–2424.
25. X. Zhang, An improved method of identifying mislabeled data and the mislabeled data in MNIST and CIFAR-10 appendix findings in fashion-MNIST. https://arxiv.org/pdf/1912.05283.pdf (11 December 2019).
26. N. M. Müller, K. Markert, "Identifying mislabeled instances in classification datasets" in *2019 International Joint Conference on Neural Networks (IJCNN)* (IEEE, 2019), pp. 1–8.
27. T. W. Anderson, *An Introduction to Multivariate Statistical Analysis* (Wiley, New York, NY, 1962).
28. A. R. Webb, D. Lowe, The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis. *Neural Network.* **3**, 367–375 (1990).
29. D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, N. Srebro, The implicit bias of gradient descent on separable data. *J. Mach. Learn. Res.* **19**, 2822–2878 (2018).
30. R. A. Fisher, The use of multiple measurements in taxonomic problems. *Ann. Eugen.* **7**, 179–188 (1936).
31. C. E. Shannon, Probability of error for optimal codes in a Gaussian channel. *Bell Syst. Tech. J.* **38**, 611–656 (1959).
32. A. Dembo, O. Zeitouni, *Large Deviations Techniques and Applications* (Springer-Verlag, Heidelberg, Germany, ed. 2, 1998).
33. J. Bruna, S. Mallat, Invariant scattering convolution networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1872–1886 (2013).

34. S. Mallat, Group invariant scattering. *Commun. Pure Appl. Math.* **65**, 1331–1398 (2012).
35. T. Wiatowski, H. Bölcskei, A mathematical theory of deep convolutional neural networks for feature extraction. *IEEE Trans. Inf. Theory* **64**, 1845–1866 (2017).
36. T. Wiatowski, M. Tschannen, A. Stanic, P. Grohs, H. Bölcskei, "Discrete deep feature extraction: A theory and new architectures" in *International Conference on Machine Learning* (Proceedings of Machine Learning Research [PMLR], Cambridge, MA, 2016), pp. 2149–2158.
37. T. Wiatowski, H. Bölcskei, "Deep convolutional neural networks based on semi-discrete frames" in *2015 IEEE International Symposium on Information Theory (ISIT)* (IEEE, 2015), pp. 1212–1216.
38. J. Zarka, L. Thiry, T. Angles, S. Mallat, "Deep network classification by scattering and homotopy dictionary learning" in *International Conference on Learning Representations* (International Conference on Learning Representations, 2020).
39. L. Sagun, L. Bottou, Y. LeCun, *Eigenvalues of the Hessian in Deep Learning: Singularity and Beyond*. arXiv:1611.07476v2 (22 November 2016).
40. L. Sagun, U. Evci, V. U. Guney, Y. Dauphin, L. Bottou, *Empirical Analysis of the Hessian of Over-Parametrized Neural Networks*. arXiv:1706.04454v3 (14 June 2017).
41. V. Papyan, The full spectrum of deep net Hessians at scale: Dynamics with sample size. (2018).
42. V. Papyan, Measurements of three-level hierarchical structure in the outliers in the spectrum of deepnet Hessians. https://arxiv.org/pdf/1811.07062.pdf (3 June 2019).
43. B. Ghorbani, S. Krishnan, Y. Xiao, "An investigation into neural net optimization via Hessians eigenvalue density" in *Proceedings of the 36th International Conference on Machine Learning* (Proceedings of Machine Learning Research [PMLR], Cambridge, MA, 2019).
44. V. Papyan, Y. Romano, M. Elad, Convolutional neural networks analyzed via convolutional sparse coding. *J. Mach. Learn. Res.* **18**, 2887–2938 (2017).
45. Y. Romano, A. Aberdam, J. Sulam, M. Elad, Adversarial noise attacks of deep learning architectures: Stability analysis via sparse-modeled signals. *J. Math. Imag. Vis.* **62**, 313–327 (2019).
46. J. Sulam, A. Aberdam, A. Beck, M. Elad, On multi-layer basis pursuit, efficient algorithms and convolutional neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **42**, 1968–1980 (2020).
47. A. Aberdam, J. Sulam, M. Elad, Multi-layer sparse coding: The holistic way. *SIAM J. Math. Data Sci.* **1**, 46–77 (2019).
48. A. Aberdam, D. Simon, M. Elad, When and how can deep generative models be inverted? arXiv:2006.15555 (28 June 2020).
49. M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, N. Usunier, "Parseval networks: Improving robustness to adversarial examples" in *Proceedings of the 34th International Conference on Machine Learning* (JMLR.org, 2017), **vol. 70**, pp. 854–863.
50. O. Deniz, A. Pedraza, N. Vallez, J. Salido, G. Bueno, Robustness to adversarial examples can be improved with overfitting. *Int. J. Mach. Learn. Cybern.* **11**, 935–944 (2020).