Dynamics and Neural Collapse in Deep Classifiers trained with the Square Loss

Akshay Rangamani\textsuperscript{1}, Mengjia Xu\textsuperscript{1,2}, Andrzej Banburski\textsuperscript{1}, Qianli Liao\textsuperscript{1}, Tomaso Poggio\textsuperscript{1}

\textsuperscript{1}Center for Brains, Minds and Machines, MIT,
\textsuperscript{2}Division of Applied Mathematics, Brown University

Abstract

Recent results suggest that square loss performs on par with cross-entropy loss in classification tasks for deep networks. While the theoretical understanding of training deep networks with the cross-entropy loss has been growing, the study of square loss for classification has been lacking. Here we study the dynamics of training under Gradient Descent techniques and show that we can expect convergence to minimum norm solutions when both Weight Decay (WD) and normalization techniques, like Batch Normalization (BN), are used. We perform numerical simulations that show approximate independence on initial conditions as suggested by our analysis, while in the absence of BN+WD we find that good solutions can be achieved for small initializations. We prove that quasi-interpolating solutions obtained by gradient descent in the presence of WD are expected to show the recently discovered behavior of Neural Collapse and describe other predictions of the theory.

This is an update to CBMM Memo 112.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216.
Dynamics and Neural Collapse in Deep Classifiers trained with the Square Loss

Akshay Rangamani\textsuperscript{1}, Mengjia Xu\textsuperscript{1,2}, Andrzej Banburski\textsuperscript{1}, Qianli Liao\textsuperscript{1}, Tomaso Poggio\textsuperscript{1}

\textsuperscript{1}Center for Brains, Minds and Machines, MIT
\textsuperscript{2}Division of Applied Mathematics, Brown University

July 8, 2021

Abstract

Recent results of \cite{1} suggest that square loss performs on par with cross-entropy loss in classification tasks for deep networks. While the theoretical understanding of training deep networks with the cross-entropy loss has been growing (\cite{2} and \cite{3}), the study of square loss for classification has been lacking. Here we study the dynamics of training under Gradient Descent techniques and show that we can expect convergence to minimum norm solutions when both Weight Decay (WD) and normalization techniques, like Batch Normalization (BN), are used. We perform numerical simulations that show approximate independence on initial conditions as suggested by our analysis, while in the absence of BN+WD we find that good solutions can be achieved for small initializations. We prove that quasi-interpolating solutions obtained by gradient descent in the presence of WD are expected to show the recently discovered behavior of Neural Collapse \cite{4} and describe other predictions of the theory.

1 Introduction

A widely held belief in the last few years has been that the cross-entropy loss is superior to the square loss when training deep networks for classification problems. As such, the attempts at understanding the theory of deep learning has been largely focused on exponential-type losses \cite{2,3}, like the cross-entropy. For these losses, the predictive ability of deep networks depends on the implicit complexity control of Gradient Descent algorithms that leads to asymptotic maximization of the classification margin on the training set \cite{5,2,6}. Recently however, \cite{1} has demonstrated empirically that it is possible to achieve the same level of performance, if not better, using the square loss, paralleling older results for Support Vector Machines (SVMs) \cite{7}. Can a theoretical analysis explain when and why regression should work well for classification? This is the question we try to answer in this paper.

There has been some recent analysis of how interpolating solutions of the square loss are equivalent to max-margin SVM solutions, in the case of linear classifiers \cite{8}. We are interested in how this translates to the case of deep networks. In our analysis we assume normalization algorithms such as Batch Normalization (BN) (or Weight Normalization (WN)) together with Weight Decay (WD), since such mechanisms seem essential for reliably training deep networks using gradient descent \cite{9}, are commonly used and were used in most of the experiments by \cite{1}. Crucially, our analysis depends on these assumptions.

In deep learning, unlike the case of linear networks, we expect several global zero square loss minima corresponding to interpolating solutions (in general degenerate, see \cite{10,11} and reference therein). Although all interpolating solutions are optimal solutions of the regression problem, they will in general correspond to different margins and to different expected classification performance. In other words, zero square loss does not imply by itself either large margin or good classification on a test set. When can we expect the solutions of the regression problem obtained by GD to have large margin? We show both experimentally and analytically that obtaining large margin interpolating
solutions using the square loss depends on the scale of initialization of the weights close to zero, in the absence of weight decay. We describe the dynamics of the norm of the deep network parameters, and show that large margin solutions can be obtained using small initializations.

In the presence of weight decay, we cannot expect perfect interpolation to occur, but only a quasi-interpolation of the labels. Our experiments and analysis of the dynamics show that, depending on the weight decay parameter, there may be independence from initial conditions, as has been observed in \[1\]. In addition, we consider the case where BN and WN are used and show that weight decay helps stabilize the solutions of the normalized parameters, in addition to its role in the dynamics of the norm.

Finally, we show that these quasi-interpolating solutions satisfy the recently discovered Neural Collapse (NC) phenomenon \[4\]. According to Neural Collapse, a dramatic simplification of deep network dynamics takes place – not only do all the margins become identical, but the last layer classifiers and the penultimate layer features form the geometrical structure of a simplex equiangular tight frame (ETF). Here we prove the emergence of Neural Collapse in both the case of the square loss and of exponential-type loss functions.

**Related Work** There has been much recent work on the analysis of deep networks and linear models trained using exponential-type losses for classification. The implicit bias of Gradient Descent towards margin maximizing solutions under exponential type losses was shown for linear models with separable data in \[12\] and for deep networks in \[2, 3, 13, 14\]. Recent interest in using the square loss for classification has been spurred by the experiments in \[1\], though the practice of using the square loss is much older \[7\]. Muthukumar et al. \[8\] recently showed for linear models that interpolating solutions for the square loss are equivalent to the solutions to the hard margin SVM problem. Recent work also studied interpolating kernel machines \[15, 16\] which use the square loss for classification.

In the recent past, there have been a number of papers analyzing deep networks trained with the square loss. These include \[17, 18\] that show how to recover the parameters of a neural network by training on data sampled from it. The square loss has also been used in analyzing convergence of training in the Neural Tangent Kernel (NTK) regime \[19, 20, 21\]. Detailed analyses of two-layer neural networks such as \[22, 23, 24\] typically use the square loss as an objective function. However these papers do not specifically consider the task of classification.

Neural Collapse (NC) \[4\] is a recently discovered empirical phenomenon that occurs when training deep classifiers using the cross-entropy loss. Since its discovery, there have been a few papers analytically proving its emergence. In \[25\] Mixon et al. show NC in the regime of “unconstrained features”. Other papers have shown the emergence of NC when using the cross entropy loss \[26, 27, 28\]. While preparing this paper, we became aware of recent results by Ergen and Pilanci \[29\] (see also \[30\]) who derived neural collapse for the square loss, through a convex dual formulation of deep networks.

**Our Contributions** The main contributions of our paper are:

- We analyze the dynamics of deep network parameters, their norm, and the margins under gradient flow on the square loss. We qualitatively describe the evolution of the norm, and the role of Weight Decay and Batch/Weight Normalization in the training dynamics. We support our conclusions with experiments.

- We show that under certain assumptions, critical points of Gradient Descent with Weight Decay satisfy the conditions of Neural Collapse for both square and exponential loss functions. Our proof technique also allows us to find the relationship between the Simplex ETF and the margin of the solution.

**Outline** We structure the rest of the paper as follows. In section 2 we describe the dynamics of gradient flow training under the square loss for a binary classification problem. We use an analysis of the dynamics to obtain insights about the role of Weight Decay and Batch/Weight Normalization. In section 3 we present and describe our experiments on CIFAR10 that highlight the insights we presented in section 2. In section 4 we present our derivation of Neural Collapse when training on the square loss, while the supplementary material extends the proof to the case of exponential loss functions. We conclude in section 5 with a discussion of our results and their implications for generalization.
2 Dynamics of Gradient Flow on the Square Loss

In this section we study a supervised classification problem solved using a deep network, and derive the dynamics of its parameters when trained using the square loss. We consider a binary classification problem on a training dataset \( S = \{(x_n, y_n)\} \) where \( x_n \in \mathbb{R}^d \) are the features (normalized such that \( \|x\| \leq 1 \) and \( y_n = \pm 1 \) are the labels. We use a deep rectified network with \( L \) layers to solve this problem, and define it as \( f_W: \mathbb{R} \to \mathbb{R}, f_W(x) = W_L \sigma (W_{L-1} \ldots \sigma (W_1 x) \ldots) \), where \( x \in \mathbb{R}^d \) is the input to the network and \( \sigma: \mathbb{R} \to \mathbb{R}, \sigma(x) = \text{max}(0, x) \) is the rectified linear unit (ReLU) activation function that is applied coordinate-wise at each layer. Due to the positive homogeneity of ReLU, we can instantiate a reparameterization of \( f_W(x) \) by considering normalized weight matrices \( V_k = \frac{W_k}{||W_k||_F} \) at each layer, and pulling out the product of the layer norms as \( \rho = \prod_k ||W_k||_F \) which gives us \( f_W(x) = \rho f_V(x) = \rho V_L \sigma (V_{L-1} \ldots \sigma (V_1 x) \ldots) \)[1].

For Deep ReLU networks we also have the following structural property of the gradient (Lemma 2.1 of [32]):
\[
\sum_{i,j} \frac{\partial f_W(x)}{\partial W_k^{i,j}} W_k^{i,j} \text{ which can be rewritten as } \left( W_k, \frac{\partial f_W(x)}{\partial W_k} \right) = f_W(x). 
\]
Here \( W_k \) refers to the weight matrix of the \( k \)-th layer of the network; \( \text{vec}(W_k) \) is its vectorized form. We use \( W_k \) for both, trusting that the context disambiguates them. We define \( f_n = f_V(x) \) (the normalized network evaluated on the training sample \( x_n \)). Note that \( f_n \leq 1 \) since \( ||x|| \leq 1 \), and the weights are normalized. Separability is defined as the condition \( y_n f_n > 0, \forall n \) (all training samples are classified correctly).

We train the deep network on the dataset \( S \) by minimizing the square loss with Weight Decay:
\[
\mathcal{L} = \frac{1}{2} \sum_n (f_W(x_n) - y_n)^2 + \frac{\lambda}{2} \sum_{k=1}^L ||W_k||_F^2.
\]
Notice that if \( f_W \) is a zero loss solution of the regression problem, then \( f_W(x_n) = y_n, \forall n \). This is equivalent to \( \rho f_n = y_n \) where \( f_n \) is the margin for \( x_n \). Thus the norm \( \rho \) of a minimizer is inversely related to its average margin. While this loss is usually minimized by Gradient Descent, we will consider the dynamical system associated with a close analogue - gradient flow:
\[
\frac{\partial W_k}{\partial t} = -\frac{\partial \mathcal{L}}{\partial W_k} = \sum_n (y_n - f_W(x_n)) \frac{\partial f_W}{\partial W_k} - \lambda W_k. \tag{1}
\]
We assume that BN is used to normalize the weight matrices. Taking the normalization constraint into account, we now convert the previous dynamical system to one in \( \rho, V_k \). Using some basic vector calculus to get \( \frac{\partial \rho}{\partial t} = \frac{1}{\rho_k} \langle W_k, \frac{\partial W_k}{\partial t} \rangle \) and \( \frac{\partial V_k}{\partial t} = \frac{1}{\rho_k} (I - V_k V_k^\top) \frac{\partial W_k}{\partial t} \), we can obtain:
\[
\frac{\partial \rho}{\partial t} = \sum_n (y_n - \rho f_n) f_n - \lambda \rho \tag{2}
\]
\[
\frac{\partial V_k}{\partial t} = \frac{1}{\rho_k} \sum_n (y_n - \rho f_n) \left( \frac{\partial f_n}{\partial W_k} - V_k f_n \right) \tag{3}
\]
where in our parametrization we have \( \rho_L = \rho \) and \( \rho_k = 1 \) for \( k \neq L \). The resulting dynamics reflects weight normalization[1]. As shown in section 3 of the supplementary material, the dynamics is similar in the absence of an active normalization but with a crucial difference that is likely to be the key to the effectiveness of BN.

Dynamics of \( \rho \) While analyzing the dynamics of \( \rho \), we can start with identifying the critical points of the dynamics. When we do not have weight decay \( (\lambda = 0) \), \( \frac{\partial \rho}{\partial t} = 0 \) for interpolating solutions. However, there may be critical points that are not interpolating solutions, and are instead local minima/saddle points. In the following, we focus on interpolating or quasi-interpolating solutions (when \( \lambda > 0 \), that is on global minima of the loss.

The critical points of the \( \rho \) dynamics with Weight Decay occur when \( \frac{\partial \rho}{\partial t} = 0 \), which happens when \( \rho = \rho_{\text{eq}} \)
\[
\rho_{\text{eq}} = \frac{\sum_n y_n f_n}{\lambda + \sum_n f_n^2} \tag{4}
\]

1We choose the Frobenius norm here to simplify our calculations. While a different choice of norm (spectral, \( \ell_1, \ell_2,1 \)) may help prove tighter generalization bounds, they are functionally equivalent for the purpose of analyzing dynamics and the margin (as noted in section 3.4 of [31]).

2Batch Normalization normalizes the activity of each output unit in a layer, and thus normalizes the rows of the \( L - 1 \) layer weight matrices in addition to global normalization.
Depending upon the values of $f_n$, there are many different values of $\rho_{eq}$. Let the smallest of those be called $\rho_{\text{min}}$. From (15), if average separability holds and $\rho < \rho_{\text{min}}$, then $\frac{\partial \rho}{\partial t} > 0$. This means that if we initialize a network with small norm, its norm grows monotonically until it reaches an equilibrium value typically close to $\rho_{\text{min}}$. Since $|f_n| < 1$, if we initialize with $\rho \ll 1$, the condition for a critical point cannot be met until $\rho$ grows larger.

If we initialize a network with large norm, average separability yields $\frac{\partial \rho}{\partial t} < 0$. This means that the norm of the network decreases until an equilibrium is reached. However since $\rho \gg 1$, we expect to find an interpolating (or near interpolating) solution that is reasonably close to the initialization, since it is usually possible to find a corresponding set of normalized parameters $V_k$ such that $|\mu f_n| \approx 1$. These solutions are related to the Neural Tangent Kernel (NTK) formulation [21], where the parameters do not move too far from their initialization.

The norm $\rho$ of the solution is also related to the (normalized) margin of the network. Critical points $\rho_{eq}$ that are small in magnitude correspond to large average margin $\sum_n y_n f_n$. To sum up, starting from small initialization, GD will explore critical points with $\rho$ growing from zero. Thus quasi-interpolating solutions with small $\rho_{eq}$ (corresponding to large margin solutions) may be found before large $\rho_{eq}$ quasi-interpolating solutions which have worse margin (See Fig. 1).

**Role of Weight Decay** Equation (15) shows that weight decay performs the traditional role of promoting solutions with small norm. In the case of large initialization, we can see from (4) that, since $|f_n|^2 \ll 1$, the scale of $\rho_{eq}$ is determined by $\lambda$. Hence weight decay stabilizes the solution of gradient descent with respect to initialization (See Fig. 2).

Norm regularization is however not the only contribution of weight decay. Equation (16) shows that the critical points $V_k = 0$ may not be normalized if the solution interpolates. By preventing exact interpolation, weight decay ensures that the critical points of (16) lie on the unit Frobenius norm ball. As we will discuss later, by preventing exact interpolation, gradient flow with weight decay under the square loss shows at convergence the phenomenon of Neural Collapse.

**Role of Batch Norm** As shown in section 3 of the supplementary material, the dynamics with and without normalization is the same for $\rho$ but somewhat different for $V_k$. In both cases $V_k$ is proportional to $A = 2 \sum_n [(\rho f_n - y_n) (V_k f_n - \frac{\partial f_n}{\partial V_k})]$. However, without normalization $V_k = \frac{A}{\rho}$ with normalization $V_k = A \rho$.

Thus the two dynamics have the same critical points but are different. Recall that at convergence $\rho_{eq}$ corresponds to the inverse of the margin: thus $\frac{1}{\rho_{eq}} = f^* = f_n$, since $f_n$ is the same for all $n$, when both normalization and Weight Decay are present (for the square loss as well as crossentropy – in the case of SGD). Thus $V_k$, for $k < L$ is proportional to the inverse of the margin in the normalized case and directly proportional to the margin in the un-normalized case. The factor $\frac{1}{F}$ or $f^*$ combines with the learning rate when Gradient Descent replaces gradient flow. Intuitively, the strategy to decrease the learning rate when the margin is large seems a better strategy than the opposite, since large margin corresponds to “good” minima in terms of generalization (for classification).

### 3 Experiments

We conducted a number of experiments on binary classification to support our claims from the analysis of the dynamics. We conducted our experiments on the standard CIFAR10 dataset [33]. Image samples with class label indices 1 and 2 were extracted for the binary classification task. The total training and test data sizes are 10000 and 2000, respectively. The model architecture contains 3 convolutional layers (the number of channels are 32, 64 and 128, filter size is $3 \times 3$) and one fully connected classifier layer with output number 2. Following each convolution layer, we applied a ReLU nonlinear activation function and Batch Normalization. Batch Normalization is used with learnable “affine” shifting and scaling parameters turned off (since they can always be learned by the next layer). The weight matrices of all layers are initialized with zero-mean normal distribution, scaled by a constant such that the Frobenius norm of each matrix is one of the initialization value set {0.01, 0.1, 0.5, 1, 3, 5, 10}. The network was trained using square loss and SGD with batch size 128, momentum 0.9, Weight Decay (0.01 or 0), constant learning rate 0.01 for 1000 epochs and no data augmentation. Every input to the network is scaled such that it has norm $\leq 1$. The plots in figure 2 are averaged over 10 different runs, while figures 13 and 4 were made from a single run.
In Fig. 1 we show the dynamics of $\rho$ alongside train loss and test error. We show results with and without Weight Decay in the top and bottom rows of Fig. 1 respectively. The left and right columns correspond to small (0.01) and large (5) initializations respectively. We see that without Weight Decay, with small initializations, $\rho$ grows monotonically, while with large initializations it decays monotonically. We can also see that small initializations without Weight Decay reach minima with smaller train loss. The top row plots also show that Weight Decay makes the final solutions robust to the scale of initialization, in terms of $\rho$ and of the train loss. This robustness is also seen in Fig. 2 where we plot the training margins $(y_n, f_n)$ obtained with and without Weight Decay. In the right plot, without Weight Decay, the margin distributions depend on the initialization, while in the left plot they cluster around the same values.

Finally, we would like to setup some motivating empirical evidence for our discussion of Neural Collapse [4]. Neural Collapse is the phenomenon in which within class variability disappears, and for all training samples, the last layer features collapse to their mean. This means that the outputs and margins also collapse to the same value. We can see this in the left plot of Fig. 2 where all of the margin histograms are concentrated around a single value. We visualize the evolution of the training margins over the training epochs in Fig. 3 which shows that the margin distribution concentrates over time. At the final epoch the margin distribution (colored in yellow) is much narrower than at any intermediate epochs. We also used measurements similar to those in [4] to confirm that Neural Collapse indeed occurs by the appropriate metrics. This is shown in Fig. 4 where we trained the same network as described earlier with a modified learning rate schedule for 350 epochs, and plot the conditions for NC1 and NC2. Section 6 of the supplementary material contains a longer discussion of these conditions, though one can also be found in [4].
Figure 2: Mean training margins over 10 runs for binary classification on CIFAR10 trained with Batch Normalization and Weight Decay = 0.01 (left) and without Weight Decay (right) for different initializations (init. = 0.01, 0.1, 0.5, 1, 3, 5 and 10). Weight Decay makes the final training margin robust to initialization, and concentrates the margin in a narrow band over the training set. The results without Weight Decay are dependent on initialization, and may result in a wide range of margin values.

Figure 3: Histogram of $|f_n|$ across 1000 training epochs for binary classification with batch normalization and weight decay = 0.01, learning rate 0.01, initialization 0.1. We can see that the histogram narrows as training progresses. The final histogram (in yellow) is concentrated in a narrow band, as expected for the emergence of NC1.

4 Neural Collapse

In a recent paper Papyan, Han and Donoho\cite{4} described four empirical properties of the terminal phase of training (TPT) deep networks, using the cross-entropy loss function. TPT 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 empirical measurements expose an inductive bias they call Neural Collapse (NC), involving four 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).

In this section we show that the phenomenon of neural collapse can be derived from the critical points of gradient flow under the square loss with Weight Decay. We consider a multiclass classification problem with $C$ classes with a balanced training dataset $S = \{(x_n, y_n)\}$ that has $N$ training examples
Figure 4: Neural Collapse occurs during training for binary classification. The key conditions for Neural Collapse are: (i) NC1 - Variability collapse, which is measured by $\text{Tr}(\Sigma_W\Sigma_B^{-1})$, where $\Sigma_W$, $\Sigma_B$ are the within and between class covariances, and (ii) NC2 - equinorm and equiangularity of the mean features $\{\mu_c\}$ and classifiers $\{W_c\}$. We measure the equinorm condition by the standard deviation of the norms of the means (in red) and classifiers (in blue) across classes, divided by the average of the norms, and the equiangularity condition by the standard deviation of the inner products of the normalized means (in red) and the normalized classifiers (in blue), divided by the average inner product. This network was trained on two classes of CIFAR10 with Batch Normalization and Weight Decay = 0.01, learning rate 0.01, initialization 3 for 350 epochs with a stepped learning rate decay schedule.

Theorem 1

For a ReLU deep network trained on a balanced dataset using gradient flow on the square loss with weight decay $\lambda$, critical points of Gradient Flow that satisfy Assumption 2 also satisfy the NC1-4 conditions for Neural Collapse.

Proof

Our training objective is $\mathcal{L}(W) = \frac{1}{2} \sum_{n=1}^{N_C} \sum_{i=1}^{C} \left( \frac{y_n}{i} - f_W^{(i)}(x_n) \right)^2 + \frac{1}{2} \lambda \sum_{l} ||W_l||^2_F$. We use gradient flow to train the network: $\frac{\partial W_c}{\partial t} = -\frac{\partial \mathcal{L}}{\partial W_c}$. Let us analyze the dynamics of the last layer, considering each classifier vector $W_c$ of $W_L$ separately:

$$\frac{\partial W_c}{\partial t} = \sum_{n} (y_n \delta - \langle W_c, h(x_n) \rangle) h(x_n) - \lambda W_c$$

$$= \sum_{n \in \mathcal{C}(c)} (1 - \langle W_c, h(x_n) \rangle) h(x_n) + \sum_{n \in \mathcal{C}(c'), c' \neq c} (-\langle W_c, h(x_n) \rangle) h(x_n) - \lambda W_c$$

Let us consider solutions that achieve symmetric quasi-interpolation, with $f_W^{(c)}(x_n(c)) = 1 - \epsilon$, and $f_W^{(c)}(x_n(c')) = \epsilon$. It is fairly straightforward to see that since $f_W^{(c)}$ and $W_c$ do not depend on $n$, neither does $h(x_n)$, which shows NC1. Under the conditions of NC1 we know that all feature vectors in a class...
collapse to the class mean, i.e., $h(x_{n(c)}) = \mu_c$. Let us denote the global feature mean by $\mu_G = \frac{1}{C} \sum_c \mu_c$. This means we have:

$$\frac{\partial W^c_L}{\partial t} = 0 \implies W^c_L = \frac{CN\epsilon}{\lambda(C-1)} \times (\mu_c - \mu_G) \quad (7)$$

This implies that the last layer parameters $W_L$ are a scaled version of the centered class-wise feature matrix $M = [\ldots \mu_c - \mu_G \ldots]$. Thus at equilibrium, with quasi interpolation of the training labels, we obtain $\frac{W_L}{||W_L||_F} = \frac{M}{||M||_F}$. This is the condition for NC3.

From the gradient flow equations, we can also see that at equilibrium, with quasi interpolation, all classifier vectors in the last layer ($W^c_L$, and hence $\mu_c - \mu_G$) have the same norm:

$$\langle W^c_L, \frac{\partial W^c_L}{\partial t} \rangle = \sum_n (y^c_n - f^c_W(x_n))f^c_W(x_n) - \lambda ||W^c_L||^2 = 0$$

$$\implies ||W^c_L||^2 = \frac{N}{\lambda} \left( \epsilon - \frac{C}{C-1} \epsilon^2 \right) \quad (8)$$

From the quasi-interpolation of the correct class label we have that $\langle W^c_L, \mu_c \rangle = 1 - \epsilon$ which means $\langle W^c_L, \mu_G \rangle + \langle W^c_L, \mu_c - \mu_G \rangle = 1 - \epsilon$. Now using (23)

$$\langle W^c_L, \mu_G \rangle = 1 - \epsilon - \frac{\lambda(C-1)}{CN\epsilon} ||W^c_L||^2 = 1 - \epsilon - \frac{\lambda(C-1)}{CN\epsilon} \times \frac{N}{\lambda} \left( \epsilon - \frac{C}{C-1} \epsilon^2 \right) = \frac{1}{C} \quad (9)$$

From the quasi-interpolation of the incorrect class labels, we have that $\langle W^c_L, \mu_c \rangle = \frac{\epsilon}{C-1}$, which means $\langle W^c_L, \mu_c - \mu_G \rangle = \frac{\epsilon}{C-1}$. Plugging in the previous result and using (24) yields

$$\frac{\lambda(C-1)}{CN\epsilon} \times \langle W^c_L, W^c_L \rangle = \frac{1}{C-1} - \frac{1}{C}$$

$$\implies \langle V^c_L, V^c_L \rangle = \frac{1}{||W^c_L||^2} \times \frac{CN\epsilon}{\lambda(C-1)} \times \left( \frac{\epsilon}{C-1} - \frac{1}{C} \right) = -\frac{1}{C-1} \quad (10)$$

Here $V^c_L = \frac{W^c_L}{||W^c_L||_2^2}$, and we use the fact that all the norms $||W^c_L||_2$ are equal. This completes the proof that the normalized classifier parameters form an ETF. Moreover since $W^c_L \propto \mu_c - \mu_G$ and all the proportionality constants are independent of $c$, we obtain $\sum_c W^c_L = 0$. This completes the proof of the NC2 condition. NC4 follows then from NC1-NC2, as shown by theorems in [4].

It is of interest to note here that in this quasi interpolation setting, the functional classification margin is given by $\eta = f_{y_n} - \max_{c \neq y_n} f_c = 1 - \epsilon - \frac{\epsilon}{C-1} = 1 - \frac{\epsilon}{C-1}. \epsilon$. The larger the margin, the smaller is $\epsilon$. Eq. (24) shows that the norms of the classifier weights are given by $||W^c_L||^2 = \frac{N\epsilon}{\sum_n |f_n| \eta}$. As we mentioned earlier, for a non-zero value of $\lambda$ we expect some small interpolation error $\epsilon$. In the binary case this is given by Eq. (5). Plugging this relationship into Eq. (24) we obtain $||W^c||^2 \approx \frac{N(1-\epsilon)}{\sum_n |f_n| \eta}$. This means that the lengths of the classifier simplex ETF are proportional to the margin.

**Other settings** The main assumptions in the above proof are symmetric quasi-interpolation and the use of Weight Decay (see section 5 of Supplementary Material). A similar version of the above proof can be adapted to the case of Stochastic Gradient Descent (SGD), where we can show that the NC conditions are met in expectation. We also show in section 5 of the Supplementary Material that an extension of this proof technique to the exponential loss case (a proxy for cross-entropy loss) requires small batch SGD to achieve the NC1 property.

**Predictions** We summarize here the main predictions of our analysis about Neural Collapse.

- The theoretical analysis predicts Neural Collapse not only for the case of cross-entropy, for which it was empirically found, but also for the square loss;
- SGD is required in our proof of NC1 (and hence the other NC conditions) for cross entropy while for the square loss neural collapse is predicted for SGD as well as GD (under Assumption 2);
Our proof uses Weight Decay for neural collapse (NC1 to NC4) under both the square loss and the cross entropy, and can also be adapted to the case of normalization and Weight Decay;

- The length of the vectors in the Simplex ETF that defines the classifier is proportional to the training margin;
- NC1 to NC4 should take place for any quasi-interpolating solutions (in the square loss case), including solutions that do not have large margin (that is, small $\rho$);
- in particular the analysis above predicts Neural Collapse for randomly labeled CIFAR10.

5 Discussion

An important question is whether Neural Collapse is related to good generalization of the solution of training. Our analysis suggests that this is not the case: Neural Collapse is a property of the dynamics independently of the size of the margin which provides an upper bound on the expected error – even if margin is likely to be just one of the factors determining out-of-sample performance. In fact, our prediction of Neural Collapse for randomly labeled CIFAR10, has been confirmed in preliminary experiments by our collaborators.

Despite the fact that Neural Collapse is independent of generalization, can our analysis of the square loss provide insights on generalization of the solutions of gradient flow? It is well known that large margin is usually associated with good generalization [34]; in the meantime it is also broadly recognized that margin alone does not fully account for generalization in deep nets [31, 35, 36]. Margin in fact provides an upper bound on generalization error, as shown in section 4 of the supplementary material. Larger margin gives a better upper bound on the generalization error for the same network trained on the same data. This property can be checked qualitatively by varying the margin using different degrees of random labels in a binary classification task (see Figure 1 in supplementary material). While training gives perfect classification and almost zero square loss, the margin on the training set increases and the test error also increases with the percentage of random labels as shown in the figure 1 in the supplementary material. However, the simple upper bound given in the same section does not explain the generalization behavior that we observe for different initializations (see Figure 2 in supplementary material), where small differences in margin are actually anticorrelated with small differences in test error.

Notice that the generalization bound in Section 4 of the supplementary material does not directly rely on the Weight Decay parameter $\lambda > 0$. However, robust convergence to large margins is helped by a non-zero $\lambda$, even if $\lambda$ is quite small, because of the associated greater independence from initial conditions in degenerate minima. This effect is quite different from the standard explanation that regularization is needed to force the norm to be small.

The main effect of $\lambda > 0$ is to eliminate degeneracy of the dynamics at the zero-loss critical points, where Equation (16) is degenerate if $\lambda = 0$. In fact, $V_k = 0$ can be satisfied even when $(V_k f_n - \partial f_n / \partial V_k) \neq 0$, implying that any interpolating solution can satisfy the equilibrium equations independently of its normalization. This degeneracy is expected, since there are infinite sets of $\rho$ and $V_k$ satisfying $\rho V_k = W_k$. Normalization thus is not effective at the critical points. Setting $\lambda > 0$ avoids this degeneracy, as shown by the critical points of Equations 15 and 16.

Limitations The theoretical analysis in this paper rests on several assumptions. First of all, we have worked here with gradient flow, rather than with the discrete case of Gradient Descent. Nothing in our analysis, however, strictly depends on this continuum limit: the results should also apply to the finite step size case. Assumption 2 on the other hand is more subtle. We showed that the assumption of a symmetric level of near interpolation implies Neural Collapse. We did not, however, formally prove that SGD on a randomly initialized network will necessarily converge to near interpolation. Our analysis only suggests that, with L2 regularization, the critical points of the gradient flow include solutions that are close to interpolation and are not biased towards specific classes in balanced datasets. Finally, we have shown that SGD with weight decay and normalization techniques (see supplementary material) is sufficient to yield Neural Collapse. The role of normalization is subtle: it is not strictly required for the proof of Neural Collapse but it facilitates convergence (especially to large margin solutions, see comments on learning rate). Importantly, the necessity of all the conditions remains an
open problem.

Acknowledgments This material is based upon work supported by the Center for Minds, Brains and Machines (CBMM), funded by NSF STC award CCF-1231216. This research was also sponsored by grants from the National Science Foundation (NSF-0640097, NSF-0827427), and AFSOR-THRL (FA8650-05-C-7262).

References


[38] T. Poggio and Y. Cooper. Loss landscape: Sgd can have a better view than gd. CBMM memo 107, 2020.


A  Degeneracies in the landscape of the loss

For large $\rho$ and very small or zero $\lambda$, we expect several solutions under GD\(^3\). The existence of several solutions is related to the existence of NTK-based solutions: intuitively the last layer is enough in an extreme case – if the last layer before the linear classifier is overparametrized wrt training data – to provide solutions for any set of random weights in the previous layers (for large $\rho$ and small $f_i$). Furthermore the intermediate layers do not need to change much under GD in the iterations immediately after initialization. The emerging picture is a landscape in which there are no zero-loss minima for $\rho < \rho_{\text{min}}$. With increasing $\rho$ from $\rho = 0$ there will be zero square-loss degenerate minima with the minimizer representing an interpolating (for $\lambda = 0$) or almost interpolating solution (for $\lambda > 0$). We expect that for $\lambda > 0$ there is a “pull” towards the minimum $\rho_{\text{eq}}$ within the local degenerate minimum of the loss and perhaps over low walls separating minima. Under certain conditions all the global minima – associated to interpolating solutions – will be connected within a unique, large valley even with $\lambda = 0$. The argument is based on Theorem 5.1 of \([37]\):

**Theorem 2 (Connected valleys of the loss [37])** If the first layer of the network has at least $2N$ neurons, where $N$ is the number of training data and if the number of neurons in each subsequent layer decreases, then every sublevel set of the loss is connected.

In particular, this implies that zero-square-loss minima with different $\rho$ are connected. A connected single valley of zero loss does not however guarantee that SGD with WD will converge to the minimum norm global minimum independently of initial conditions. The reason is that the connected valley will in general twist in the space of parameters in such a way that following it does not monotonically increase or decrease $\rho$.

B  Critical points of SGD

For SGD with minibatch of size $1$ the interpolation conditions imply $\rho f(x_n) - y_n = 0$, $\forall n$. With high probability this should hold also for minibatch sizes larger than $1$ but smaller than size of the training set $N$.

C  Dynamics with and without Normalization

The original Weight Normalization paper [39] introduces a parameterization of each layer’s parameters $W_k$ in terms of $V_k$ and $g_k$ such that $W_k = g_k \frac{V_k}{||V_k||}$. The dynamics on $g_k$ and $V_k$ is induced by the gradient dynamics of $W_k$ as follows (assuming $\frac{\partial W_k}{\partial t} = -\frac{\partial L}{\partial W_k}$)

\[
\frac{\partial g_k}{\partial t} = \left( \frac{V_k}{||V_k||} \cdot \frac{\partial W_k}{\partial t} \right)
\]

(11)

\[
\frac{\partial V_k}{\partial t} = \frac{g_k}{||V_k||} S_k \frac{\partial W_k}{\partial t}
\]

(12)

with $S_k = I - \frac{V_k V_k^T}{||V_k||^2}$.

Writing this in terms of our $\rho$, $V_k$ parameterization and using the fact that $||V_k|| = 1$, we have:

\[
\frac{\partial \rho}{\partial t} = \sum_n (y_n - \rho f_n) f_n - \lambda \rho
\]

(13)

\[
\frac{\partial}{\partial t} \frac{V_k}{\rho} = \rho \sum_n (y_n - \rho f_n) \left( \frac{\partial f_n}{\partial V_k} - V_k f_n \right)
\]

(14)

If we do not use Weight Normalization, the dynamics are given by:

\(^3\)It is interesting to recall [10] that for SGD – unlike GD – the algorithm will stop only when $\ell_n = \rho f_n - y_n = 0$ $\forall n$, which is the global minimum and corresponds to perfect interpolation. For the other critical points for which GD will stop, SGD will never stop but may fluctuate around the critical point.
\[
\frac{\partial \rho}{\partial t} = \sum_n (y_n - \rho f_n) f_n - \lambda \rho
\]  \hspace{1cm} (15)

\[
\frac{\partial V_k}{\partial t} = \frac{1}{\rho} \sum_n (y_n - \rho f_n) \left( \frac{\partial f_n}{\partial V_k} - V_k f_n \right)
\]  \hspace{1cm} (16)

The two dynamics have the same critical points but are different. Recall that at convergence \( \rho_{eq} \) corresponds to the inverse of the margin \( f^* = f_a \) which is the same for all \( n \) when both normalization and weight decay are present. Thus \( \frac{\partial V_k}{\partial t}, k < L \) is proportional to the inverse of the margin in the normalized case and directly proportional to the margin in the unnormalized case. The factor \( \frac{1}{L} \) or \( f^* \) combines with the learning rate when gradient descent replaces gradient flow. Intuitively, the strategy to decrease the learning rate when the margin is large seems a better strategy than the opposite, since large margin corresponds to “good” minima in terms of generalization (for classification).

\section{D Margin and generalization}

Assume that the square loss is exactly zero and the margins \( f_n \) are all the same. Then recall simple bounds that hold with probability at least \( (1 - \delta) \), \( \forall g \in G \) of the form \( [34] \):

\[
|L(g) - \bar{L}(g)| \leq c_1 \mathbb{R}_N(G) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}}
\]  \hspace{1cm} (17)

where \( L(g) = \mathbb{E}[\ell_{\gamma}(g(x), y)] \) is the expected loss, \( \bar{L}(g) \) is the empirical loss, \( \mathbb{R}_N(G) \) is the empirical Rademacher average of the class of functions \( G \) measuring its complexity; \( c_1, c_2 \) are constants that reflect the Lipschitz constant of the loss function and the architecture of the network. The loss function here is the ramp loss \( \ell_{\gamma}(g(x), y) \) defined as

\[
\ell_{\gamma}(y, y') = \begin{cases} 
1, & \text{if } yy' \leq 0, \\
1 - \frac{yy'}{\gamma}, & \text{if } 0 \leq yy' \leq \gamma, \\
0, & \text{if } yy' \geq \gamma.
\end{cases}
\]

We define \( \ell_{\gamma=0}(y, y') \) as the standard 0-1 classification error and observe that \( \ell_{\gamma=0}(y, y') < \ell_{\gamma>0}(y, y') \).

We now consider two solutions with zero empirical loss of the square loss regression problem obtained with the same ReLU deep network and corresponding to two different minima with two different \( \rho \)s. Let us call them \( g^a(x) = \rho_n f^a(x) \) and \( g^b(x) = \rho_b f^b(x) \). Using the notation of this paper, the functions \( f_n \) and \( f_b \) correspond to networks with normalized weight matrices at each layer.

Let us assume that \( \rho_a < \rho_b \).

We now use the observation that, because of homogeneity of the ReLU networks, the empirical Rademacher complexity satisfies the property,

\[
\mathbb{R}_N(G) = \rho \mathbb{R}_N(F),
\]  \hspace{1cm} (18)

where \( G \) is the space of functions of our unnormalized networks and \( F \) denotes the corresponding normalized networks. This observation allows us to use the bound Equation \( [17] \) and the fact that the empirical \( \bar{L}_{\gamma} \) for both functions is the same to write \( L_0(f^a) = L_0(F^a) \leq c_1 \rho_a \mathbb{R}_N(F) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \) and \( L_0(f^b) = L_0(F^b) \leq c_1 \rho_b \mathbb{R}_N(F) + c_2 \sqrt{\frac{\ln(\frac{1}{\delta})}{2N}} \). The bounds have the form

\[
L_0(f^a) \leq A \rho_a + \epsilon
\]  \hspace{1cm} (19)

and

\[
L_0(f^b) \leq A \rho_b + \epsilon
\]  \hspace{1cm} (20)

Thus the upper bound for the expected error \( L_0(f^a) \) is better than the bound for \( L_0(f^b) \). Of course this is just an upper bound. Lower bounds are not available. As a consequence this result does not guarantee that a solution with smaller \( \rho \) will always have a smaller expected error than a solution with larger \( \rho \).
D.1 Experiments on generalization

The property discussed above can be checked qualitatively by varying the margin using different degrees of random labels in CIFAR in a binary classification task. While training gives perfect classification and almost zero square loss, the margin on the training set decreases and the test error also increases with the percentage of random labels as shown in Fig. 5. See also Fig. 7.

However, the margin does not explain the behavior shown in Fig. 6 where small differences in margin are actually anticorrelated with small differences in test error. If there exist several almost-interpolating solutions with the same norm \( \rho_{eq} \), they may have similar norm and similar margin but different ranks of the weight matrices (or of the rank of the local Jacobian). In deep linear networks the GD dynamics seems to bias the solution towards small rank solutions, since large eigenvalues converge much faster the small ones [40]. It is tempting to assume that the rank may have a role in generalization and in explaining Fig. 6.

Figure 5: Mean \( 1/\rho \) and test error results over 10 runs for binary classification on CIFAR10 trained with batch normalization and different percentages of random labels (\( r = 20\%, 40\%, 60\%, 80\% \) and \( 100\% \)), initialization scale 0.1 and weight decay 0.01.

E Additional derivations of Neural Collapse

In section 4 of the main paper we showed that the phenomenon of Neural Collapse can be derived from the critical points of gradient flow under the square loss with Weight Decay. In this section we present derivations of Neural Collapse in two more situations. In section E.1 we show that the critical points of gradient flow under weight normalization also exhibits Neural Collapse. In section E.2 we show that NC occurs in the case of deep networks trained with the exponential loss as well.

E.1 Normalization case

In this subsection we stick to the setting of section 4 of the main paper and consider a multiclass classification problem with \( C \) classes with a balanced training dataset \( \mathcal{S} = \{(x_n, y_n)\} \) that has \( N \) training examples per class. We train a ReLU deep network \( f_W : \mathbb{R}^d \rightarrow \mathbb{R}^C \), \( f_W(x) = W_L \sigma(W_{L-1} \ldots W_2 \sigma(W_1 x) \ldots) \) with Gradient Descent on the square loss with Weight Decay on the parameters of the network. Under the normalized parameterization, we have \( f_W(x) = \rho f_V(x) \), where \( f_V(x) = V_L \sigma(V_{L-1} \ldots V_2 \sigma(V_1 x) \ldots) \) is the normalized network. We use the following array notation to denote the output vectors and the one-hot target vectors respectively: \( f_V(x) = [f_V^{(i)}(x)], y_n = [y_n^{(i)}] \). We will also follow the notation of [4] and use \( h(x) \) to denote the last layer features of the deep network. This means that \( f_V^{(c)}(x) = \langle V_L^c, h(x) \rangle \). Similar to section 4 of the main paper, we assume that the solution obtained by Gradient Descent satisfies the Symmetric Quasi-interpolation condition which we recall below:
Figure 6: Scatter plot for $1/\rho$ and mean test accuracy based on 10 runs for binary classification on CIFAR10. The network was trained with different initialization scales ($\text{init.} = 0.01, 0.1, 0.5, 1, 3, 5$ and 10), using batch normalization and weight decay 0.01. The horizontal and vertical error bars correspond to the standard deviations of $1/\rho$ and test accuracy computed over 10 runs for different initializations, while the square dots correspond to the mean values.

**Assumption 2 (Symmetric Quasi-interpolation)** Consider a $C$-class classification problem with inputs in a feature space $\mathcal{X}$, a classifier $f : \mathcal{X} \rightarrow \mathbb{R}^C$ symmetrically quasi-interpolates a training dataset $S = \{(x_n, y_n)\}$ if for all training examples $x_{n(c)}$ in class $c$, $f(c)(x_{n(c)}) = 1 - \epsilon$, and $f(c')(x_{n(c)}) = \epsilon C^{-1}.

This brings us to the main result of this section:

**Theorem 3** For a ReLU deep network trained on a balanced dataset using gradient flow on the square loss with Weight Normalization and Weight Decay, critical points of Gradient Flow that satisfy Assumption 2 also satisfy the NC1-4 conditions for Neural Collapse.

**Proof** Our training objective is $L(\rho, V) = \frac{1}{2} \sum_{n=1}^{NC} ||y_n - \rho f_V(x_n)||^2 + \frac{\lambda}{2} \rho^2$. We can use Gradient Flow with Weight Normalization algorithm to train the network. We can relate this algorithm to Gradient Flow on the regular network parameterization $f_w$ through the following equation: $\frac{\partial V_L}{\partial t} = -\rho_k S_k \frac{\partial \mathcal{L}(W)}{\partial W_L} = -\rho_k S_k \frac{\partial \mathcal{L}(W)}{\partial W} \frac{\partial \mathcal{L}(W)}{\partial W_L}$, where $S_k = I - V_k V_k^T$. For the parameters of the last layer, this translates to:

$$\frac{\partial V_L}{\partial t} = -\rho \left( I - [V_L^T V_L]^T \right) \frac{\partial \mathcal{L}(W)}{\partial W_L} \frac{\partial \mathcal{L}(W)}{\partial W} \frac{\partial \mathcal{L}(W)}{\partial W_L}.$$  

This means:

$$\frac{\partial V_L}{\partial t} = \rho \left[ \sum_n (y_n - \rho f_V(x_n)) h(x_n) - \sum_n (y_n - \rho f_V(x_n), f_V(x_n)) V_L \right]$$  \hspace{1cm} (21)

Now let us analyze the critical points of the dynamics of the last layer, considering each classifier vector $V_L^c$ of $V_L$ separately:

$$\sum_n (y_n - \rho f_V(x_n), f_V(x_n)) V_L^c = \sum_n (y_n^c - \rho f_V(x_n^c)) h(x_n)$$  \hspace{1cm} (22)

Let us consider solutions that achieve symmetric quasi-interpolation, with $\rho f_V^c(x_{n(c)}) = 1 - \epsilon$, and $\rho f_V^c(x_{n(c')}) = \epsilon C^{-1}$. It is fairly straightforward to see that since $f_V^c$ and $V_L^c$ do not depend on $n$, neither does $h(x_n)$, which shows NC1. Under the conditions of NC1 we know that all feature vectors in a class
collapse to the class mean, i.e., \( h(x_{n(c)}) = \mu_c \). Let us denote the global feature mean by \( \mu_G = \frac{1}{C} \sum_c \mu_c \). This means we have:

\[
\frac{CN}{\rho} \left( \epsilon(1 - \epsilon) - \frac{\epsilon^2}{C-1} \right) V_L^c = \epsilon N \mu_c - \frac{\epsilon N}{C-1} \sum_{j \neq c} \mu_j
\]

\[
= \frac{\epsilon N C}{C-1} (\mu_c - \mu_G)
\]

\[
\Rightarrow V_L^c = \frac{\rho}{1 - \frac{C}{\epsilon-1} \frac{\epsilon^2}{C-1}} (\mu_c - \mu_G)
\]

(23)

This implies that the last layer parameters \( V_L \) are a scaled version of the centered class-wise feature matrix \( M = [\cdots \mu_c - \mu_G \cdots] \). Thus at equilibrium, with quasi interpolation of the training labels, we obtain \( \frac{V_L}{||V_L||_F} = \frac{M}{||M||_F} \). This is the condition for NC3.

From the gradient flow equations, we can also see that at equilibrium, with quasi interpolation, all classifier vectors in the last layer (\( V_L^c \) and hence \( \mu_c - \mu_G \)) have the same norm:

\[
||V_L^c||_2^2 = \frac{\sum_n (\mu_n(c) - \rho f_V(x_n)) f_V(x_n)}{\sum_n (\mu_n - \rho f_V(x_n)) f_V(x_n)} = \frac{CN}{\rho} (1 - \epsilon) - \frac{\epsilon^2}{C-1}
\]

\[
= \frac{1}{C}
\]

(24)

From the quasi-interpolation of the correct class label we have that \( \langle V_L^c, \mu_c \rangle = \frac{1-\epsilon}{\rho} \) which means \( \langle V_L^c, \mu_G \rangle + \langle V_L^c, \mu_c - \mu_G \rangle = \frac{1-\epsilon}{\rho} \). Now using (23)

\[
\langle V_L^c, \mu_G \rangle = \frac{1 - \epsilon}{\rho} - \left( 1 - \frac{C}{\epsilon-1} \right) (C - 1) \frac{1}{||V_L^c||_2^2}
\]

\[
= \frac{1 - \epsilon}{\rho} - \frac{C-1}{C} = \frac{1}{\rho C}
\]

(25)

From the quasi-interpolation of the incorrect class labels, we have that \( \langle V_L^c, \mu_c \rangle = \frac{1-\epsilon}{\rho(C-1)} \), which means \( \langle V_L^c, \mu_c - \mu_G \rangle + \langle V_L^c, \mu_c \rangle = \frac{1-\epsilon}{\rho(C-1)} \). Plugging in the previous result and using (24) yields

\[
\frac{(C - 1)}{\rho} \left( 1 - \frac{C}{\epsilon-1} \right) \times \langle V_L^c, V_L^c \rangle = \frac{\epsilon}{\rho(C - 1)} - \frac{1}{\rho C}
\]

\[
\Rightarrow \langle \tilde{V}_L^c, \tilde{V}_L^c \rangle = \frac{1}{||V_L^c||_2^2} \times \frac{-1}{C(C-1)} = - \frac{1}{C-1}
\]

(26)

Here \( \tilde{V}_L^c = \frac{V_L^c}{||V_L^c||_2} \), and we use the fact that all the norms \( ||V_L^c||_2 \) are equal. This completes the proof that the normalized classifier parameters form an ETF. Moreover since \( V_L^c \propto \mu_c - \mu_G \) and all the proportionality constants are independent of \( c \), we obtain \( \sum_c V_L^c = 0 \). This completes the proof of the NC2 condition. NC4 follows then from NC1-NC2, as shown by theorems in [4].

E.2 Exponential case

Here we show that in the case of binary classification with exponential loss (a proxy for logistic loss), we require small batch SGD to derive the Neural Collapse properties. We leave an extension of this calculation to the multi-class case to future work.

For the exponential loss with normalization and weight decay the gradient flow corresponding to GD are \( \frac{\partial \mu_n}{\partial t} = \frac{1}{N} \sum_n e^{-\rho \eta_n} f_n y_n f_n - \lambda \rho \) and \( \frac{\partial \mu_k}{\partial t} = \rho \frac{1}{N} \sum_n e^{-\rho \eta_n} f_n S_k \frac{\partial f_n}{\partial \mu_k} \).
The equations at equilibrium are
\[ 0 = \frac{1}{N} \sum_n e^{-\rho y_n f_n} y_n f_n - \lambda \rho \]
\[ 0 = \frac{1}{N} \sum_n e^{-\rho y_n f_n} y_n S_k \frac{\partial f_n}{\partial V_k} \]
with \( S_k = I - V_k V_k^T \).

For GD a critical point with \( \frac{\partial V}{\partial t} = 0 \) implies \( \lambda \rho = \frac{1}{N} \sum_n e^{-\rho y_n f_n} y_n f_n \) and thus at \( \frac{\partial V}{\partial t} = 0 \) it implies
\[ \sum_n e^{-\rho y_n f_n} y_n \frac{\partial f_n}{\partial V_k} = V_k \sum_n e^{-\rho y_n f_n} y_n f_n = \lambda \rho V_k \]  
(28)

The condition above does not by itself imply that all the margins \( y_n f_n \) are the same (which is required for NC1). The situation is however different for SGD: equilibrium for SGD with minibatch size of 1 (the argument is valid also for minibatch sizes larger than 1 but smaller than \( N \), see [10]) implies that each of the terms should be vanishing independently, i.e.
\[ e^{-\rho y_n f_n} y_n f_n = \lambda \rho, \quad \forall n = 1, \cdots, N \]  
(29)
and thus \( y_1 f_1 = y_n f_n, \quad \forall n = 1, \cdots, N \). Hence, the margins \( y_1 f_1 \) are all equal. Then \( V_L h_{i,c} = f^c \) is also independent of \( i \) implying that \( h_{i,c} \) is independent of \( i \) at convergence.

With this result, the other NC properties follow in similar way as in the square loss case.

From the \( \frac{\partial V}{\partial t} = 0 \) equation, we immediately get that at the critical points \( h(x_+) \sim V_L f(x_+) \) and \( h(x-) \sim V_L f(x_-) \), giving us \( h(x_+) = -h(x_-) \). This also implies that \( \mu_+ = -\mu_- \) and \( \mu_G = 0 \). It follows then that, defining \( M = [\mu_+ - \mu_0, \mu_- - \mu_G], \) NC3 follows as \( \frac{W_L}{\|W_L\|_F} = V_L = \frac{M^T}{\|M\|_F} \).

The fact that \( \mu_+ = -\mu_- \) also immediately gives us the equinorm property, and we get that \( \langle \mu_+, \mu_- \rangle = -\frac{1}{N} = -1 \) and hence \( \langle \mu_c, \mu_{c'} \rangle = \frac{C}{C-1} \delta_{c,c'} - \frac{1}{C-1} \). Thus NC2 follows. As in the case of square loss, by the results of [4] we also get NC4, completing the proof.

The proof for the exponential loss requires SGD, unlike the square loss case. We do not know whether this is just a technicality. We conjecture that is not. In this case the prediction would be the NC1 should be found under the square loss case with or without SGD, whereas NC1 under the exponential loss requires SGD. We further conjecture that small minibatch sizes should be better than large ones for the exponential loss case.

## F  More details on Neural Collapse

In this section we recap the details of Neural Collapse as described in [4]. It is an empirical phenomenon that has been observed in the terminal phase of training deep networks, which we can associate with training beyond the point of separation. We listed the four conditions that are associated with Neural Collapse in section 4 of the main paper. Here we present them in a more formal manner.

We first define a deep network \( f_W(x) = W_L h(x) \), where \( h(x) \in \mathbb{R}^p \) denotes the last layer features of the deep network, and \( W_L \in \mathbb{R}^{C \times p} \) contains the parameters of the classifier. The network is trained on a \( C \)-class classification problem on a balanced dataset \( \{(x_n,y_n)\} \) with \( N \) samples per class. We can compute the per-class mean of the last layer features as:
\[ \mu_c = \frac{1}{N} \sum_{n \in N(c)} h(x_n) \]  
(30)
The global mean of all features can be computed as:
\[ \mu_G = \frac{1}{C} \sum_c \mu_c \implies \mu_G = \frac{1}{NC} \sum_{n=1}^{NC} h(x_n) \]  
(31)
The second order statistics of the last layer features can be computed as:

\[ \Sigma_W = \frac{1}{C} \sum_{c=1}^{C} \frac{1}{N} \sum_{n \in N(c)} (h(x_n) - \mu_c)(h(x_n) - \mu_c)^T \]

\[ \Sigma_B = \frac{1}{C} \sum_{c=1}^{C} (\mu_c - \mu_G)(\mu_c - \mu_G)^T \]

\[ \Sigma_T = \frac{1}{NC} \sum_{n=1}^{NC} (h(x_n) - \mu_G)(h(x_n) - \mu_G)^T \] (32)

Where \( \Sigma_W \) is the within class covariance of the features, \( \Sigma_B \) is the between class covariance, and \( \Sigma_T \) is the total covariance of the features (\( \Sigma_T = \Sigma_W + \Sigma_B \)).

We can now list the formal conditions for Neural Collapse:

### NC1 (Variability collapse)
\( \Sigma_W \to 0 \), or within-class variability of last-layer training activations collapses to zero.

### NC2 (Convergence to Simplex ETF)
\[ |||\mu_c - \mu_G|||_2 - |||\mu_{c'} - \mu_G|||_2 \to 0 \], or the centered class means of the last layer features become equinorm. Moreover, if we define \( \tilde{\mu}_c = \frac{\mu_c - \mu_G}{|||\mu_c - \mu_G|||_2} \), then we have \( \langle \tilde{\mu}_c, \tilde{\mu}_{c'} \rangle = -\frac{1}{C-1} \) for \( c \neq c' \), or the centered class means are also equiangular. The equinorm condition also implies that \( \sum_c \tilde{\mu}_c = 0 \), i.e. the centered features lie on a simplex.

### NC3 (Self-Duality)
If we collect the centered class means into a matrix \( M = [\mu_c - \mu_G] \), we have
\[ \frac{\|
abla W^2 - M\|_{\text{F}}}{\|
abla W^2\|_{\text{F}}} \to 0 \], or that the classifier \( W \) and the last layer feature means \( M \) become duals of each other.

### NC4 (Nearest Center Classification)
The classifier implemented by the deep network eventually boils down to choosing the closest mean last layer feature \( \arg\max_c \langle W^c_L, h(x) \rangle \to \arg\min_c \|h(x) - \mu_c\|^2 \)

In figure 4 of the main paper we track the convergence of a deep network to Neural Collapse by primarily observing NC1 and NC2. We see in the first panel that \( \text{Tr}(\Sigma_W \Sigma_B^{-1}) \to 0 \), which implies that NC1 is achieved, while in the second and third panel we track the equinorm and equiangular conditions. The second panel plots the ratio of standard deviation to the mean value of \( |||\mu_c - \mu_G|||_2 \), and \( |||W^c_L|||_2 \) in red and blue respectively, while the third panel plots the ratio of standard deviation to the mean of \( \frac{1}{C-1} + \cos(\mu_c - \mu_G; \mu_{c'} - \mu_G) \), and \( \frac{1}{C-1} + \cos(W^c_L, W^c_{L'}) \) in red and blue respectively. The convergence of all quantities to zero indicates that the NC2 conditions are also achieved. NC3 and NC4 follow accordingly.

### G Additional Figures

In our random label experiments, we trained a simple 4-layer ConvNet with BN and Weight Decay 0.01, initialization 0.1 using the square loss and different random label ratios (\( r = 20\%, 40\%, 60\%, 80\% \) and \( 100\% \)) for binary classification on two classes of the CIFAR10 dataset. The total train and test data size are 10000 and 2000, respectively. The training dynamics of the product norm \( \rho \) w.r.t. different random label ratios over 10 runs are shown in Fig. 7. The asymptotic product norm \( \rho \) values are increased from 80 to 120 with the increasing percentages of random labels, while the margin are decreased with the increasing percentages of random labels.
The dynamics of $\rho$ for the binary classification experiment trained with ground-truth labels using BN and Weight Decay 0.01 are shown in Fig. 8. The results in different rows are based on three different initializations (0.01, 1 and 5). The first two columns show the dynamics of $\rho$ results over 10 runs for the first 3 epochs and 30 epochs, respectively. The last column indicates the $\rho$ dynamics over 1000 epochs, where we can observe that the asymptotic $\rho$ values are mostly similar and are independent of different initializations (0.01, 1 and 5) when we applied Weight Decay during training. However, the dynamics of $\rho$ for the case without Weight Decay (see Fig. 9) are unstable across different initializations; moreover, large initializations (e.g., init = 5) shown in the third row of Fig. 9 achieved large variations even across different training runs.

Additionally, the final training and test performance for our binary classification experiments with BN and Weight Decay 0.01 are shown in Fig. 10. Two main observations can be made from the results: 1) Small initialization 0.01 achieved the highest mean test accuracy (93.71%) with small standard deviation in the first row; large initializations (1 and 5) in the second and third rows also achieved relative good mean test accuracy but with large standard deviations over 10 runs. 2) The training loss converged faster with small initialization than large initializations (1 and 5) during training over 1000 epochs. The training accuracy for small initialization also converged faster compared to using large initializations (1 and 5).

Fig. 11 shows the binary classification performance trained with BN and no Weight Decay. This generates similar trends as discussed in the previous results with Weight Decay (Fig. 10), i.e., small initialization (0.01) produced smaller training loss and better test accuracy. The training/test accuracy converged much faster compared to the two large initialization cases (1 and 5 in the second and third rows), which also resulted in smaller standard deviations over 10 runs for both training and testing. However, large initializations achieved more unstable training and testing performances with large standard deviation over 10 different training runs. Without using Weight Decay, the obtained test performance varies substantially across different initializations. Large initializations achieved much lower test accuracy.
Figure 8: Dynamics of $\rho$ for the binary classification experiments trained with square loss, batch normalization and weight decay 0.01 for 10 runs. From top to bottom correspond to initialization 0.01, 1 and 5.
Figure 9: Dynamics of $\rho$ for the binary classification experiments trained with square loss over 10 runs using batch normalization and no weight decay. From the top to bottom correspond to initialization 0.01, 1 and 5, respectively.
Figure 10: Performance of binary classification over 10 runs on two classes of CIFAR10 trained with batch normalization and weight decay 0.01. From top to bottom correspond to initialization 0.01, 1 and 5, respectively.
Figure 11: Performance of binary classification on two classes of CIFAR10 trained with batch normalization and no weight decay for 10 runs. From the top to bottom correspond to initialization 0.01, 1 and 5, respectively.